Ansible

op Ubuntu

Inhoudsopgave

Table of Contents

1 INLEIDING OEFENING....4
1.1 Alle nodige pakketten....4
1.2 Hostnames in /etc/hosts....4
1.3 Gebruiker aanmaken voor Ansible....5
1.4 IP adres bij login scherm....5
2 PASWOORDLOZE SSH....6
2.1 Sleutel genereren....6
2.2 Uittesten SSH verbinding....7
3 ANSIBLE COMMANDO'S....8
3.1 Shell commando....8
3.2 Ansible hosts....8
3.3 Root rechten....8
3.4 Installeren met apt....9
3.5 Ansible copy....9
3.6 templates....10
3.7 synchronize....10
3.8 get_url....10
3.9 git....11
4 ANSIBLE WEBSERVER PLAYBOOK....12
4.1 YAML bestanden....12
4.2 Ansible-lint....12
4.3 Webserver deploy....13
4.4 Webserver met een template....14
4.5 Handlers....15
5 VARIABLES EN FACTS....17
5.1 Variabele declareren....17
5.2 Variabelen in een bestand....17
5.3 Facts....18
5.4 Output van een commando....18
6 ANSIBLE BEST PRACTICES....21
6.1 Roles....21
6.2 Ansible-galaxy....21
7 ANSIBLE MODULES EN PLUGINS....24
7.1 Module Plugin....24
7.2 Custom Module....25
7.3 Verschil Modules en Plugins....27
7.4 Plugins....27
8 ANSIBLE COLLECTION....29
9 OEFENING....31
10 BIJLAGE ANSIBLE CONTAINER....32
11 REFERENTIES....35

1 INLEIDING OEFENING

DOEL: Configuratie Ansible

NODIG: Twee Ubuntu >=22.04 LTS systemen (grafische interface optioneel).
Kies minstens 2 processoren !

OPZET:

master
host-only 192.168.57.100/24
|
|
host-only 192.168.57.101/24
node

Een alternatieve opzet kan via een docker container als node. De mogelijke configuratie hiervan vind je in de bijlage.

1.1 Alle nodige pakketten

master:

```
root@master:~# sudo apt install ansible openssh-server
```

Dit kan op linux systemen ook via python bv:

```
root@master:~# pip3 install --user ansible>=2.10
```

1.2 Hostnames in /etc/hosts

Indien je geen ip adres gekregen hebt kan dit met het commando dhclient *interfacenaam*.
Pas /etc/hosts aan door de naam en het ip adres van de master en de node(s) toe te voegen:

```
127.0.0.1           localhost
192.168.57.100      master
192.168.57.101      node
```

Opgelet, verwijder eventueel de lijn met 127.0.1.1. Deze zorgt er anders voor dat nodes proberen verbinding te maken met localhost in plaats van het echte IP adres.

```
127.0.0.1           localhost
~~127.0.1.1         master~~
192.168.57.100      master
192.168.57.101      node
```

De naam van je systeem zelf kan je als root aanpassen met het commando:

```
root@localhost:~# hostnamectl set-hostname master
```

Let er op dat je even terug een nieuwe shell/terminal of sessie moet starten eer de naam wordt aangepast:

```
root@localhost:~# exit
user@localhost:~$ sudo su
root@master:~#
```

1.3 Gebruiker aanmaken voor Ansible

Op ALLE nodes (inclusief de master) maken we een gebruiker "ansible" aan. De home directory van deze gebruiker wordt de /home/ansible directory.
Voor het aanmaken van de gebruiker, kan je volgend commando gebruiken.

```
root@master:~# groupadd ansible
root@master:~# useradd -m -u 1001 -g ansible ansible
root@master:~# echo ansible:supersecret| chpasswd -c SHA512
```

1.4 IP adres bij login scherm

```
#!/bin/bash
echo '\S' > /etc/issue
echo 'Kernel \r on an \m'>> /etc/issue
ip a | grep -o '192.*/' >> /etc/issue
cp /etc/issue /etc/issue.net
```

2 PASWOORDLOZE SSH

Dit commando doe je op de nodes en de master om ssh te gebruiken

.

```
root@master:~# sudo apt install openssh-server
root@master:~# systemctl start sshd
```

2.1 Sleutel genereren

Voor de communicatie naar de nodes maken we gebruik van passwoordloze ssh.
We loggen in met de nieuwe gebruiker op de master en genereren daar een RSA sleutelpaar. Je mag de default locatie behouden en GEEN paswoord ingeven.

```
root@master:~#  su - ansible
ansible@master:~$  ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansible/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansible/.ssh/id_rsa.
Your public key has been saved in /home/ansible/.ssh/id_rsa.pub.
The key fingerprint is:
16:ea:35:48:f1:56:4a:7e:0d:e1:6c:2c:1e:fe:60:4d ansible@master
The key's randomart image is:
+--[ RSA 2048]----+
|      . . +.     |
|       = * o     |
|      ._ 0  .    |
|     /   \       |
|     |0v0|.      |
|  ...++ ++..     |
|      \./   .    |
|                 |
|                 |
+-----------------+
ansible@master:~$  ssh-copy-id node
The authenticity of host 'node (192.168.57.101)' can't be established.
ECDSA key fingerprint is 60:16:dd:2a:9a:53:c4:96:ff:54:e0:10:cf:99:6c:f1.
Are you sure you want to continue connecting (yes/no)? yes
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted
now it is to install the new keys
ansible@node's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'node'"
and check to make sure that only the key(s) you wanted were added.
```

2.2 Uittesten SSH verbinding

Er mag NIET naar een paswoord gevraagd worden:

```
ansible@master:~$ ssh node
Last login: Tue Feb 29 23:46:08 2024
ansible@node:~$  exit
logout
Connection to node closed.
```

3 ANSIBLE COMMANDO'S

3.1 Shell commando

Test een shell commando uit op de node

```
ansible@master:~$ ansible all --inventory "node," -m shell -a 'echo Ansible '
node | CHANGED | rc=0 >>
Ansible
```

3.2 Ansible hosts

Schrijf de node in het hosts bestand /etc/ansible/hosts.

```
[nodes]
node
```

Je kan nu zonder --inventory of -i een commando uitvoeren op de shell op een host:

```
ansible@master:~$ ansible all -m shell -a 'echo Ansible op ;hostname'
node | CHANGED | rc=0 >>
Ansible op
node
```

Je kan nu met ansible een ping uitvoeren naar je node.

```
ansible@master:~$ ansible all -m ping
node | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

De uptime van de nodes opvragen

```
ansible@master:~$ ansible all -m shell -a "uptime"
node | CHANGED | rc=0 >>
 18:40:42 up 32 min,  2 users,  load average: 0,01, 0,00, 0,00
```

3.3 Root rechten

Standaard heeft de ansible gebruiker geen rechten om als root iets aan te passen.

Voer als root op de node volgend commando uit

```
root@node:~# echo "ansible ALL=(ALL) NOPASSWD: ALL"> /etc/sudoers.d/ansi
```

Hierdoor kan gebruiker ansible zonder paswoord een sudo commando uitvoeren.

Normaal gebruik je geen sudo commando om aan te geven dat je dingen als root gaar doen. Optie --become of -b is om root te worden.

```
ansible@master:~$ ansible all -m shell -a 'sudo whoami'
[WARNING]: Consider using 'become', 'become_method', and 'become_user' rather
than running sudo node | CHANGED | rc=0 >>
root
ansible@master:~$ ansible --become all -m shell -a 'whoami'
node | CHANGED | rc=0 >>
root
```

3.4 Installeren met apt

Van zodra je root rechten hebt, kan je ansible gebruiken om pakketten te installeren of desinstalleren.

Package stress installeren met apt:

```
ansible@master:~$ ansible all -b -m apt -a "name=stress state=present"
```

Package htop installeren met apt:

```
ansible@master:~$ ansible all -b -m apt -a "name=htop state=present"
```

Zorg dat je een extra terminal opendoet naar de node. Start daar het programma htop op. Start dan 30 seconden het programma stress op:

```
ansible@master:~$ ansible all -m shell -a 'stress --cpu 2 --timeout 30s'
```

Package htop verwijderen kan met de status absent:

```
ansible@master:~$ ansible all -b -m apt -a "name=htop state=absent"
```

3.5 Ansible copy

Voer volgende commando's uit op de master in het tekstbestand cmd.sh. Rechten worden overgenomen van de source. Je kan ze ook instellen met de mode parameter.

```
ansible@master:~$ echo '#!/bin/bash' > cmd.sh
ansible@master:~$ echo 'echo Hello world' >> cmd.sh
ansible@master:~$ echo 'ip a|grep -o 172.*/|cut -d/ -f1' >> cmd.sh
ansible@master:~$ ansible all -m copy -a 'src=cmd.sh dest=cmd.sh'
ansible@master:~$ ansible all -m shell -a './cmd.sh'
node | FAILED | rc=126 >>
/bin/sh: 1: ./cmd.sh: Permission deniednon-zero return code
ansible@master:~$ ansible all -m copy -a 'src=cmd.sh dest=cmd.sh mode=755'
ansible@master:~$ ansible all -m shell -a './cmd.sh'
node | CHANGED | rc=0 >>
Hello world
node
```

3.6 templates

Soms willen we in bestanden variabelen at runtime aanpassen. Om dat te doen hebben we templates nodig.
Templates worden in jinja2 geschreven. Variabelen worden ingesloten in dubbele accolades. Maak het bestand `cmd-template.sh.j2` met volgende inhoud:

```
#!/bin/bash
echo {{ bericht }}
ip a|grep -o 172.*/|cut -d/ -f1
```

Je kan met de module template de variabelen in de template invullen bij het "copieren". Bij een shellscript zoals hier, geef je best executable rechten mee, zodat het script kan opgestart worden.

```
ansible@master:~$ ansible all -m template -a "src=cmd-template.sh.j2 dest=cmd-
template.sh mode=755" -e "bericht=Hello"
ansible@master:~$ ansible all -m shell -a './cmd-template.sh'
```

3.7 synchronize

Ansible kan gebruik maken van rsync om bestanden te synchroniseren tussen source en destination.

Package rsync installeren met apt:

```
ansible@master:~$ ansible all -b -m apt -a "name=rsync state=present"
```

Je kan ook de mode push of pull gebruiken.

```
ansible@master:~$ ansible all -m synchronize -a 'src=cmd.sh dest=cmd.sh'
node | CHANGED => {
    "changed": true,
    "cmd": "/usr/bin/rsync --delay-updates -F --compress --archive
--rsh=/usr/bin/ssh -S none -o StrictHostKeyChecking=no -o
UserKnownHostsFile=/dev/null --out-format=<<CHANGED>>%i %n%L
/home/ansible/cmd.sh node:cmd.sh",
    "msg": "<f..tp..... cmd.sh\n",
    "rc": 0,
    "stdout_lines": [
        "<f..tp..... cmd.sh"
    ]
}
```

3.8 get_url

Bestanden kunnen ook vanuit een url binnengehaald worden

```
ansible@master:~$ ansible all -m get_url -a
"url='https://github.com/luckylittle/ansible-cheatsheet/blob/master/ansible-
cheatsheet.txt' dest=."
ansible@master:~$ ansible all -m shell -a 'ls -al *.txt'
node | CHANGED | rc=0 >>
-rw-rw-r-- 1 ansible ansible 184014 Feb 29 23:55 ansible-cheatsheet.txt
```

3.9 git

Bestanden kunnen ook vanuit een git repo binnengehaald worden
Package git en tree (optioneel) installeren met apt:

```
ansible@master:~$ ansible all -b -m apt -a "name=git,tree state=present"
```

De git repo binnenhalen:

```
ansible@master:~$ ansible all -m git -a
"repo=https://github.com/luckylittle/ansible-cheatsheet.git dest=./cheatsheet"
ansible@master:~$ ansible all -m shell -a 'tree cheat*'
node | CHANGED | rc=0 >>
cheatsheet
|-- DO407-exam-notes.txt
|-- README.md
|-- ansible-cheatsheet.txt
|-- ec2.ini
`-- ec2.py
0 directories, 5 files
```

4 ANSIBLE WEBSERVER PLAYBOOK

4.1 YAML bestanden

Een ansible playbook definieert specifieke taken.
Een YAML bestand start met drie dashes --- om aan te geven dat het een YAML bestand is.

- Inspringen moet met spaties en mag niet met een tab.
- Opsommingen maak je met een streepje
- Elke task krijgt een naam en een actie.

Maak het bestand playbook-apt.yml met volgende inhoud:

```
---
- name:hosts: all
  tasks:
  - name: Install apache2 package
    apt: name=apache2 state=present update_cache=true
  - name: Start apache2 server
    service: name=apache2 state=started
```

Hierbij geeft "hosts" weer op welke hosts het script moet draaien. Bij "hosts:all" kijkt ansible in het bestand /etc/ansible/hosts.
De module apt installeert apt packages (state=present). Wanneer apt de optie update_cache=true meekrijgt zal er eerst een apt-get update plaatsvinden (de pakketlijsten krijgen dan een update zodat de laatste versies bekend zijn).

Opstarten met:

```
ansible@master:~$ ansible-playbook --become playbook-apt.yml
```

Command line kan dit ook

```
ansible@master:~$ ansible all -b -m apt -a "name=apache2 state=present
update_cache=true"
ansible@master:~$ ansible all -b -m service -a "name=apache2 state=started"
```

Surf nu naar de node. De website moet draaien op poort 80.

4.2 Ansible-lint

Nakijken van een playbook kan je met het pakket ansible-lint.

```
ansible@master:~$ apt-cache search ansible | grep lint
ansible-lint - lint tool for Ansible playbooks
root@master:~# sudo apt-get install ansible-lint
```

Als alles in orde is geeft ansible-lint geen output.

```
ansible@master:~$ ansible-lint playbook-apt.yml
ansible@master:~$
```

4.3 Webserver deploy

Maak het bestand webserver-deploy.yml aan met volgende inhoud

```
---
- name: Webserver Playbook - Deploy
  hosts: all
  tasks:
    - name: Install apache2 package
      apt:
        name: apache2
        state: present
    - name: Start and enable apache2 service
      service:
        name: apache2
        enabled: true
        state: started
    - name: Create a custom index.html file
      copy:
        mode: "644"
        dest: /var/www/html/index.html
        content: |
          Ansible Works
          This is my webpage
```

Installeren kan met:

```
ansible@master:~$ ansible-playbook -b webserver-deploy.yml
```

Kijk na met je browser op adres http://node

Maak het bestand met de naam webserver-remove.yml.
Hierin plaats je het volgende bestand:

```
---
- name: Webserver playbook - Remove
  hosts: all
  tasks:
    - name: Stop en disable apache2 service
      service:
        name: apache2
        enabled: false
```

```
        state: stopped
    - name: Remove apache2 package
      apt:
        name: apache2
        state: absent
```

Verwijderen doe je met:

```
ansible@master:~$ ansible-playbook -b webserver-remove.yml
```

4.4 Webserver met een template

Voer opnieuw de deploy uit met:

```
ansible@master:~$ ansible-playbook -b webserver-deploy.yml
```

Maak webserver-fact.yml:

```
---
- name: Update index.html with host IP
  hosts: all
  become: true
  gather_facts: true
  tasks:
    - name: Get host IP address
      set_fact:
        host_ip: "{{ ansible_default_ipv4.address }}"
    - name: Update index.html with host IP
      template:
        mode: "644"
        src: index-ip.html.j2
        dest: /var/www/html/index.html
```

Maak de template index-ip.html.j2 aan met volgende inhoud:

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to My Ansible Server</title>
</head>
<body>
    <h1>Hello World!</h1>
    <p>My IP is: {{ host_ip }}</p>
</body>
</html>
```

```
ansible@master:~$ ansible-playbook webserver-fact.yml
```

Surfen naar de site geeft het IP adres van de webserver weer.

4.5 Handlers

Handlers zijn zoals tasks in een playbook. In tegenstelling tot een task, zal een handler enkel opgestart worden wanneer ze door een notify getriggerd worden bij een statusverandering.

Maak het bestand webserver-handler.yml:

```
---
- name: Restart Apache when website changes
  hosts: all
  become: true
  gather_facts: true
  tasks:
    - name: Install apache2 package
      apt:
        name: apache2
        state: present
    - name: Start and enable apache2 service
      service:
        name: apache2
        enabled: true
        state: started
    - name: Create a custom index.html file
      copy:
        mode: "644"
        dest: /var/www/html/index.html
        content: |
          Ansible time {{ ansible_date_time.iso8601 }}
          on my webpage

      notify:
       - Restart Apache Handler

  handlers:
     - name: Restart Apache Handler
       service:
         name: apache2
         state: restarted
```

De webpagina zal bij elke uitvoer een update krijgen van de tijd. De status komt hierdoor op Changed. Notify zal dus elke keer de handler triggeren.

```
ansible@master:~$ ansible-playbook webserver-handler.yml

PLAY [Restart Apache when website changes]
***********************************

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

TASK [Install apache2 package]
************************************************
ok: [node]

TASK [Start and enable apache2 service]
***************************************
ok: [node]

TASK [Create a custom index.html file]
****************************************
changed: [node]

RUNNING HANDLER [Restart Apache Handler]
*****************************************
changed: [node]

PLAY RECAP
*********************************************************************
node                 : ok=5    changed=2    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0

ansible@master:~$ curl http://node
Ansible time 2023-02-29T11:55:16Z
on my webpage
```

5 VARIABLES EN FACTS

5.1 Variabele declareren

In de sectie vars definieer je de variabelen die je wilt gebruiken.

```
---
- name: Variabele gebruiken
  hosts: all
  vars:
    - bericht1: "hello"
    - bericht2: "world"
  tasks:
    - name: Show var
      debug: msg="Bericht {{ bericht1 }} {{ bericht2 }}"
```

5.2 Variabelen in een bestand

Maak het bestand mysecrets.yml met volgende inhoud:

```
secret1: "hello"
secret2: "secretworld"
```

Gebruik volgende playbook:

```
---
- name: Variabele file gebruiken
  hosts: all
  vars_files:
    - mysecrets.yml
  tasks:
    - name: Show var
      debug: msg="Bericht {{ secret1 }} {{ secret2 }}"
```

5.3 Facts

Maak het bestand myfacts.yml met volgende inhoud:

```
---
- name: Facts gebruiken
  hosts: all
  gather_facts: true
  tasks:
   - name: Show a fact var
     ansible.builtin.debug:
      var: ansible_facts['nodename']
   - name: Show a fact debug message
     debug: msg="Hostname {{ ansible_facts['nodename'] }} "
   - name: Echo a fact in the shell put it in myvar
     shell: "echo 'Host {{ansible_nodename}}'"
     register: myvar
   - name: Show a variable as debug message
     debug: msg=" {{ myvar.stdout }} "
```

```
ansible@master:~$ ansible-playbook myfacts.yml
```

5.4 Output van een commando

Schrijf het bestand date-time.yml:

```
---
- name: Uses ansible_date_time fact and date cmd
  hosts: all
  tasks:
  - name: Ansible fact - ansible_date_time
    debug:
     msg: "Date: {{ ansible_date_time.date }} Time
{{ ansible_date_time.time }}"

  - name: Get timestamp from the system
    shell: "date +%Y-%m-%d%H-%M-%S"
    register: datetime

  - name: Set variables
    set_fact:
     cur_date: "{{ datetime.stdout[0:10]}}"
     cur_time: "{{ datetime.stdout[10:]}}"
```

```
  - name: System timestamp - date time
    debug:
     msg:  "Date: {{ cur_date }} Time: {{ cur_time }}"
```

```
ansible@master:~$ ansible-playbook date-time.yml

PLAY [Uses ansible_date_time fact and date cmd]
*******************************

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

TASK [Ansible fact - ansible_date_time]
*****************************************
ok: [node] => {
    "msg": "Date: 2023-12-08 Time 12:14:48"
}

TASK [Get timestamp from the system]
******************************************
changed: [node]

TASK [Set variables]
*********************************************************
ok: [node]

TASK [System timestamp - date time]
*******************************************
ok: [node] => {
    "msg": "Date: 2023-12-08 Time: 12-14-48"
}

PLAY RECAP
*********************************************************************
node                 : ok=5    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Met volgende ansible playbook maak je een directory aan, schrijf je een shell script, start je het script op en vang je de output op in een variabele.

```
---
- name: Create and Write Shell Script
  hosts: all
  become: true
  vars:
    script_directory: /home/ansible/scripts
    script_name: my_script.sh
    script_path: "{{ script_directory }}/{{ script_name }}"
  tasks:
    - name: Create a directory for the script
      file:
```

```
      path: "{{ script_directory }}"
      state: directory
      mode: 0755  # Make the directory executable

  - name: Write a shell script
    copy:
      content: |
        #!/bin/sh
        echo "Hello from the generated script!"
      dest: "{{ script_path }}"
      mode: 0755  # Make the script executable

    - name: Run a shell script
      command: "{{ script_path }}"
      register: script_output
      changed_when: script_output.rc != 0
      # Uses the return code to define
      # when the task has changed.

    - name: Print the script output
      debug:
        var: script_output.stdout
```

6 ANSIBLE BEST PRACTICES

Best practices bij ansible is werken met een directory structuur waarin op een vaste manier alle elementen van ansible kunnen vastgelegd worden. Die elementen groepeer je in roles.
Verder is het aan te raden om ansible-lint te gebruiken om een vaste opmaak, gebruik van commentaar, gebruik van lege lijnen, gebruik van inspringingen en gebruik van namen te standaardiseren.

6.1 Roles

In Ansible is een "role" een manier om playbooks en taken te structureren en te modulariseren. Het biedt een mechanisme om taken, variabelen, handlers , files en templates te organiseren, zodat ze gemakkelijk kunnen worden hergebruikt in verschillende projecten of omgevingen.

6.2 Ansible-galaxy

Ansible galaxy is een tooltje dat gebruik maakt van de site https://galaxy.ansible.com om roles in te laden die door andere gebruikers zijn samengesteld. Je kan er ook Collections gebruiken (dat zijn combinaties van roles)

Hier gebruiken we ansible-galaxy om zelf een directorystructuur aan te maken en te gebruiken. Ansible is gemaakt om deze structuur vanuit een git-repository in te laden.
Met het commando ansible-galaxy init maken we de directorystructuur aan:

```
ansible@master:~$ mkdir -p roles && cd roles
ansible@master:~/roles$ ansible-galaxy init nginx
- Role nginx was created successfully
ansible@master:~/roles$ tree nginx
nginx
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files

ansible@master:~/roles$ cd ..
```

roles/nginx/vars/main.yml:

```
---
nginx_port: 80
nginx_server_name: www.mijnserver.com
```

roles/nginx/tasks/main.yml:

```
---
- name: Role task update cache for Ubuntu
  apt:
    update_cache: true
  when: ansible_os_family == 'Debian'

- name: Role task install Nginx
  package:
    name: nginx
    state: present

- name: Role task Copy website files
  copy:
    mode: "644"
    src: "{{ item }}"
    dest: /var/www/html/
  with_fileglob: "*.html"

- name: Role task copy Nginx configuration
  template:
    mode: "644"
    src: default.j2
    dest: /etc/nginx/sites-enabled/default
  notify: Handler Restart Nginx
```

roles/nginx/templates/default.j2:

```
server {
    listen {{ nginx_port }} default_server;
    server_name {{ nginx_server_name }};
    root /var/www/html;
    index index.html index.htm ;
}
```

roles/nginx/handlers/main.yml:

```
---
- name: Handler Restart Nginx
  service:
    name: nginx
    state: restarted
```

roles/nginx/meta/main.yml:

```
---
galaxy_info:
  author: Jan Celis
  description: Install and configure Nginx webserver
  license: GPL-3.0
  min_ansible_version: 2.8
  platforms:
    - name: Ubuntu
```

We maken een playbook aan. Deze maakt gebruik van de nginx role. Deze playbook MOET in de directory hoger dan "roles" staan!

playbooknginx.yml:

```
---
- name: Install nginx from roles directory
  hosts: all
  gather_facts: yes
  become: yes

  tasks:
    - name: Install nginx using a role
      import_role:
        name: nginx
```

Schrijf in de files directory een eigen index.html bestand, zodat je de server kan testen:

```
ansible@master:~/roles$ echo "Mijn nginx website gemaakt met een role" >
roles/nginx/files/index.html
```

Start playbooknginx.yml op:

```
ansible@master:~/roles$ ansible-playbook playbooknginx.yml
```

Test uit met een browser naar de node.

7 ANSIBLE MODULES EN PLUGINS

7.1 Module Plugin

In Ansible is een module een op zichzelf staand, herbruikbaar stuk code dat een specifieke taak uitvoert op een target machine. Ansible-modules kunnen in verschillende programmeertalen worden geschreven.

Voorbeeld: De ingebouwde module file kan bestanden en directories aanmaken/verwijderen en rechten instellen

```
- name: Create "/etc/test/" and set permissions
  ansible.builtin.file:
    path: /etc/test
    state: directory
    mode: '0750'
```

De module wait_for bevriest de uitvoering van een playbook totdat er aan een bepaalde voorwaarde is voldaan.
Onderstaand voorbeeld wacht tot er een string in een bestand staat, alvorens verder te gaan met de taken:

```
- name: Wait_for a string is in a file
  ansible.builtin.wait_for:
    path: /tmp/example_file
    search_regex: "String exists, continue"
```

Deze volledige playbook wacht op een pid bestand, dat pas zal aangemaakt zijn wanneer de server gestart is:

```
---
- name: Wait_for a pid file exists
  hosts: all
  tasks:
    - name: Wait for the existence of /var/run/nginx.pid
      wait_for:
        path: /var/run/nginx.pid
```

7.2 Custom Module

Een eigen module kan je in elke taal schrijven. Ansible heeft een bibliotheek in python waarmee je eenvoudig je eigen module kan maken.
Maak de directory modules aan:

```
ansible@master:~$ mkdir -p modules && cd modules
ansible@master:~/modules$
```

Als voorbeeld maken we het bestand /home/ansible/modules/iwashere.py aan.

```
#!/usr/bin/python
__author__    = "Jan Celis <jan.celis@kdg.be>"
__copyright__ = "copyright 2023 GPL v3.0"
__license__   = "GPL"
__version__   = "0.1"
__status__    = "Prototype"

import json
import sys
from ansible.module_utils.basic import AnsibleModule

def main():
  module = AnsibleModule(
    argument_spec = dict(
      name  = dict(required=True, type='str'),
      email = dict(required=True, type='str'),
    )
  )
  name  = module.params['name']
  email = module.params['email']
  data  = dict(
    output="Your tagfile was stored successfully",
  )
  try:
    file = open("/tmp/iwashere.txt","w")
    file.write(name+ "," + email + "\n")
    module.exit_json(changed=True, success=data,msg=data)
  except Exception as e:
    module.fail_json(msg='Something went wrong')

if __name__ == '__main__':
  main()
```

Dit pythonscript zal als parameter de naam en het email adres gebruiken en wegschrijven in het bestand /tmp/iwashere.txt. Moest het wegschrijven niet lukken dan stopt het script met een exception en foutmelding.

Deze module kunnen we in playbook playbook-iwashere.yml oproepen met "iwashere".

```
---
- name: Test Module iwashere
  hosts: all
  become: true
  tasks:
  - name: Create file /tmp/iwashere.txt
    iwashere: name="Jan Celis" email="jan.celis@kdg.be"
```

Om de module te gebruiken stel je de variable ANSIBLE_LIBRARY in met het PATH van de module:

```
ansible@master:~/modules$ export ANSIBLE_LIBRARY=/home/ansible/modules
ansible@master:~/modules$ ansible-playbook playbook-iwashere.yml
```

7.3 Verschil Modules en Plugins

In onderstaande playbook zijn apt, copy en service **module plugins** die op de target uitgevoerd worden. De **become plugin** zorgt ervoor dat je iets als root gebruiker zal kunnen uitvoeren en de **notify plugin** zal een taak starten na een trigger.

```
---
- name: Install and Configure Nginx
  hosts: all
  become: yes

  tasks:
    - name: Install Nginx
      apt:
        name: nginx
        state: present

    - name: Start Nginx service
      service:
        name: nginx
        state: started

    - name: Copy Nginx configuration
      copy:
        src: /etc/nginx/nginx.conf
        dest: /etc/nginx/nginx.conf.bak
      notify:
        - Restart Nginx

  handlers:
    - name: Restart Nginx
      service:
        name: nginx
        state: restarted
```

7.4 Plugins

Plugins voeren functies uit voor een module. In principe zijn modules plugins die taken uitvoeren op de target computer.
Er zijn verschillende soorten plugins. Enkele voorbeelden:

Inventory Plugins:
Doel: Inventory-plugins bepalen hoe Ansible hosts ontdekt en groepeert voor het uitvoeren van taken. Dat kan bv vanuit een bestand, vanuit de cloud of vanuit een databank.
Voorbeeld: ini, yaml, ec2

Connection Plugins:
Doel: Connection-plugins definiëren hoe Ansible verbinding maakt met hosts en taken uitvoert. Connecties gebeuren standaard over een SSH verbinding, maar er zijn ook ingebouwde plugins voor Windows Remote Management (winrm) en Windows PowerShell Remote Protocol (psrp).
Voorbeeld: ssh, paramiko, local, winrm, psrp

Module Plugins:
Doel: Module-plugins vertegenwoordigen de uitvoerbare eenheden van werk in Ansible. Ze implementeren specifieke taken.
Voorbeeld: shell, copy, apt

Lookup Plugins:
Doel: Lookup-plugins halen gegevens op tijdens de uitvoering van een playbook.
Voorbeeld: file, env, template

Filter Plugins:
Doel: Filter-plugins veranderen de weergave van variabelen of manipuleren de gegevens tijdens de uitvoering.
Voorbeeld: default, regex_replace, json_query

Callback Plugins:
Doel: Callback-plugins worden aangeroepen na specifieke gebeurtenissen in de uitvoering, bijvoorbeeld na het voltooien van een taak of playbook, opvang van shell output
Voorbeeld: default, json, yaml

Strategy Plugins:
Doel: Strategy-plugins bepalen hoe taken parallel of sequentieel worden uitgevoerd.
Voorbeeld: linear, free, debug

Action Plugins:
Doel: Action-plugins worden gebruikt om specifieke taken uit te voeren binnen een playbook.
Voorbeeld: add_host, set_fact

Vars Plugins:
Doel: Vars-plugins bieden dynamische variabelewaarden tijdens de uitvoering.
Voorbeeld: host_vars, group_vars

Test Plugins:
Doel: Test-plugins worden gebruikt om bepaalde voorwaarden te controleren
Voorbeeld: bool, equals, failed_when

8 ANSIBLE COLLECTION

Een ansible collection is een bundeling van playbooks, roles, modules en plugins die je kan ge/herbruiken. Collections haal je vanuit de Ansible Galaxy website (https://galaxy.ansible.com).

We installeren eerst een nginx webserver vanuit een role die we terug vinden op de Ansible Galaxy site:

```
ansible@master:~$ ansible-galaxy role search nginxinc

Found 21 roles matching your search:

 Name                                  Description
 ----                                  -----------
 nginxinc.nginx                        Official Ansible role for installing>
 nginxinc.nginx_app_protect            Official Ansible role for installing>
 nginxinc.nginx_config                 Official Ansible role for configurin>
 nginxinc.nginx_controller_agent       A role to install, configure, and up>
 ...

ansible@master:~$ ansible-galaxy role install nginxinc.nginx
```

We maken het bestand galaxy-role.yml

```
---
- name: Install nginx from ansible galaxy role
  hosts: all
  become: true
  tasks:
    - name: Install NGINX
      ansible.builtin.include_role:
        name: nginxinc.nginx
      vars:
        nginx_branch: stable
```

```
ansible@master:~$ ansible-playbook galaxyrole.yml
```

Daarna bekijk je online de collection devsec.hardening. Deze bevat de roles:
msql_hardening
nginx_hardening
os_hardening
ssh_hardening

Bij de install staat er een requirement voor ansible versie >=2.9.10. Mogelijk moet je updaten naar een hogere/de laatste versie van ansible. Dat kan als volgt:

```
root@master:~# sudo apt-get remove --purge ansible
root@master:~# sudo apt-add-repository ppa:ansible/ansible
root@master:~# sudo apt update
root@master:~# sudo apt-get install ansible
```

We installeren de collectie devsec.hardening:

```
ansible@master:~$ ansible-galaxy collection install devsec.hardening
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Downloading
https://galaxy.ansible.com/api/v3/plugin/ansible/content/published/
collections/artifacts/devsec-hardening-9.0.0.tar.gz to
/home/ansible/.ansible/tmp/ansible-local-30585hzsse5a0/tmpf8s9ng1w/devsec-
hardening-9.0.0-xds5a8xy
Installing 'devsec.hardening:9.0.0' to
'/home/ansible/.ansible/collections/ansible_collections/devsec/hardening'
devsec.hardening:9.0.0 was installed successfully
'ansible.posix:1.5.4' is already installed, skipping.
'community.mysql:3.8.0' is already installed, skipping.
'community.crypto:2.16.1' is already installed, skipping.
'community.general:7.5.2' is already installed, skipping.
```

We schrijven een playbook. Deze maakt gebruik van de nginx_hardening role in de collection devsec.hardening:

```
---
- name: Harden Nginx server
  hosts: all
  become: true
  roles:
    - name: devsec.hardening.nginx_hardening
```

```
ansible@master:~$ ansible-playbook galaxyhardening.yml
```

9 OEFENING

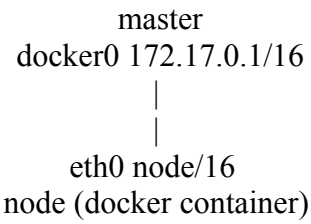
Schrijf een ansible playbook dat gebruik maakt van een role. Voorzie hiervoor de juiste directorystructuur.

Volgende taken worden uitgevoerd op een node met standaard ubuntu 22.04:
- Installeren van nginx, opstarten als service en ook na reboot
- Verzekeren dat er geen apache2 op staat of kan opstarten
- Aanpassen van de standaard website met een custom html pagina vanuit een template met volgende facts: hostname, ip adres, processor en RAM
- Installeren van mogelijk extra software die je nodig hebt
- Schrijven en draaien van een script op de node dat nakijkt of de website draait op poort 80
- Draait de website, dan toon je "Running", anders toon je "Not Running"
- Zorg dat alle yml files voldoen aan ansible-lint

Maak ook een playbook waarin alles gestopt en verwijderd wordt.

10 BIJLAGE ANSIBLE CONTAINER

Containeropstelling
OPZET:



Voor deze configuratie heb je docker nodig. De installatie en configuratie van docker valt buiten de scope van deze cursus.

Installeer ansible en openssh op de master (ook docker uiteraard)

```
root@master:~# sudo apt install ansible openssh-server
```

Zorg er voor dat de hostname van de master klopt.

```
root@master:~# hostnamectl set-hostname master
root@master:~# nslookup 172.17.0.1
1.0.17.172.in-addr.arpa  name = master.
```

Kijk het ip adres van de master na. Normaal heeft deze bij de installatie van docker het adres 172.17.0.1 gekregen:

```
root@master:~# ip -4 a show dev docker0
5: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
group default
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
      valid_lft forever preferred_lft forever
```

Maak de user ansible aan op de master

```
root@master:~# groupadd ansible
root@master:~# useradd -m -u 1001 -g ansible ansible
root@master:~# echo "ansible:supersecret"|chpasswd
```

Log in als user ansible en maak met ssh-keygen een sleutel om automatisch in te loggen in de node
Gebruik de default ssh-keygen opties en GEEN paswoord.

```
root@master:~# su - ansible
ansible@master:~$ mkdir /home/ansible/.ssh
ansible@master:~$ ssh-keygen -t rsa
```

Dit laatste commando maakt in /home/ansible/.ssh het bestand id_rsa.pub aan. Dat bestand heb je nodig op de node. Het bestand krijgt daar de naam authorized_keys. De Dockerfile zet dit bestand in de image klaar.

Dockerfile:

```
FROM ubuntu:22.04

LABEL author="Jan Celis <jan.celis@kdg.be>"
LABEL description="Start ansible container met autologin"
LABEL requires="Publieke sleutel van 'master' id_rsa.pub"

# Software installeren
RUN apt-get update && apt-get -y install iproute2
RUN apt-get -y install ansible
RUN apt-get -y install sudo
RUN apt-get -y install openssh-server
RUN mkdir /var/run/sshd

# Gebruiker ansible aanmaken
RUN groupadd ansible
RUN useradd -m -u 1001 -g ansible ansible
RUN echo "ansible:supersecret"|chpasswd
RUN mkdir /home/ansible/.ssh

# Rechten voor ansible om root te worden zonder paswoord
RUN mkdir -p /etc/sudoers.d/
RUN echo 'ansible ALL=(root)
NOPASSWD:ALL'>/etc/sudoers.d/ansible

# Rechten voor ansible om automatisch in te loggen
COPY id_rsa.pub /home/ansible/.ssh/authorized_keys

# Blijven draaien met ssh daemon
CMD ["/usr/sbin/sshd","-D"]

# Poorten openzetten
EXPOSE 21 22 80 443 8080
```

Met deze dockerfile EN het bestand id_rsa.pub kan je een image aanmaken.
De hostname van de master MOET je toevoegen opdat er automatisch met de gebruiker ansible via ssh kan ingelogd worden. Deze hostname en IP adres komt in /etc/hosts van de container.

```
user@master:~$ docker build -t ansibleimg --add-host=master:172.17.0.1 .
```

Als alles succesvol verloopt, kan je nu een container opstarten met deze image

Eerst verwijderen we een mogelijke vorige container met dezelfde naam:

```
user@master:~$ docker container stop ansiblecontainer 2>/dev/null
user@master:~$ docker container rm -f ansiblecontainer 2>/dev/null
```

Dan starten we de container op. We laten deze als daemon draaien (-d) en we laten de poorten automatisch doormappen naar de container (-P):

```
user@master:~$ docker run -d -P --name ansiblecontainer ansibleimg
user@master:~$ docker exec ansiblecontainer ip a | grep -o 172.*/
node/
```

Je kan nu met de gebruiker ansible vanuit de master een automatische ssh login doen bij de container

```
user@master:~$ su - ansible
ansible@master:~$ ssh node
The authenticity of host 'node (node)' can't be established.
ED25519 key fingerprint is SHA256:G7mPHaha9JCtApk5w5MxHJW/fHz8qEXZoTeKehSKI.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'node' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-36-generic x86_64)

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
$
```

De container stoppen en verwijderen kan met volgende commando's:

```
user@master:~$ docker container stop ansiblecontainer 2>/dev/null
user@master:~$ docker container rm ansiblecontainer 2>/dev/null
```

De image kan je verwijderen met:

```
user@master:~$ docker image rm ansibleimg
```

11 REFERENTIES

1. SENSO V., Practical Ansible, Apress 2021,146 p.

2. MEIJER B. e.a, Ansible Up and Running, 3rd Edition, O Reilly 2023, 590 p.

3. SSH login without password
   http://www.linuxproblem.org/art_9.html

4. Deploy and remove a webserver
   https://www.redhat.com/sysadmin/ansible-callback-plugins

5. Ansible copy files
   https://adamtheautomator.com/ansible-copy/

6. Ansible variables
   https://spacelift.io/blog/ansible-variables

7. Ansible Date and Timestamp
   https://ttl255.com/ansible-getting-date-and-timestamp/

8. Ansible Best Practices
   https://docs.ansible.com/ansible/2.8/user_guide/playbooks_best_practices.html

9. Developing Ansible Roles
   https://www.redhat.com/sysadmin/developing-ansible-role

10. Custom Ansible Module
    https://learning-ocean.com/tutorials/ansible/ansible-custom-module

11. Difference between Modules and Plugins
    https://thecloudops.org/difference-between-modules-and-plugins/

12. Ansible Collection Tutorial
    https://ostechnix.com/ansible-collections/