

Workshop Vagrant + Ansible



Contents

Let op!	3
Software voorbereiden	3
Opzetten van de Vagrant Omgeving	3
Vagrantfile	4
Vagrant Box	5
Ansible	8
Playbooks	9
Standaard Playbook	9
Playbooks Runnen	11
Opdrachten	12
Opdracht 1 (Node)	12
Opdracht 2 (Apache)	12

Let op!

Omdat ansible niet natively kan draaien op Windows machines en een deel van de klas Windows machines gebruikt zal Ansible in deze tutorial draaien op de vagrant box zelf in plaats van op een control machine.

Software voorbereiden

Voordat we beginnen aan de workshop is het belangrijk dat op de machine waar je mee gaat werken zowel de nieuwe versie van Vagrant en Virtual Box zijn geïnstalleerd.

Je kan de laatste versies voor Windows / OSX en Linux hieronder vinden:

Vagrant: <https://www.vagrantup.com/downloads.html>

Virtual Box: <https://www.virtualbox.org/wiki/Downloads>

Opzetten van de Vagrant Omgeving

We gaan nu beginnen met het opzetten van de Vagrant box. Het eerste deel van de workshop zal zijn opgesplitst in OSX en Windows aangezien de installatie van een Vagrant box net iets anders werkt op ieder platform.

Maak een nieuwe map aan op je Mac of Windows machine en noem deze iets als “vagrant-tut”.

Windows

Open je “Powershell” als administrator en navigeer naar de locatie waar je je nieuwe map aan wilt maken, in dit geval doen we dit in de root van de C schijf. (Of met de hand mocht je dit makkelijker vinden)

```
• cd c:/  
• mkdir vagrant-tut  
• cd vagrant-tut
```

OSX

Open je “Terminal” en navigeer naar de locatie waar je je nieuwe map aan wilt maken, in dit geval doen we dit in de root van de home directory. (Of met de hand mocht je dit makkelijker vinden)

```
• cd ~  
• mkdir vagrant-tut  
• cd vagrant-tut
```

Vagrantfile

Nu is het tijd om de vagrant box op te gaan zetten op je lokale machine. Net als bij de vorige stap open je de command line interface die hoort bij je besturingssysteem. (Terminal of Powershell)

```
PS C:\vagrant-tut> vagrant init ubuntu/trusty64
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

Ga naar de map die we hierboven hebben aangemaakt en initieer een nieuwe vagrant box met het volgende commando:

- `vagrant init ubuntu/trusty64`

Het stukje “ubuntu/trusty64” is een variabele optie die je kan wijzigen door een andere template / box te kiezen op: <https://app.vagrantup.com/boxes/search> maar voor deze tutorial is het aangeraden om de standaard box te gebruiken.

Je zal nu zien dat er een nieuw bestand is aangemaakt in je “vagrant-tut” map genaamd “Vagrantfile”. Open dit bestand in je favoriete IDE (Atom / PhpStorm / Notepad++) en voeg het volgende stukje toe:

- `config.vm.provision :shell do |sh|`
- `sh.path = "provision.sh"`
- `sh.args = [".ansible provisioning/setup.yml provisioning/hosts/dev_hosts"]`
- `end`



Pastebin: <https://pastebin.com/X0rb04jy>

Onder:

- `config.vm.box = "ubuntu/trusty64"`

Mocht er iets fout gaan kan je de hele Vagrantfile hier vinden: <https://pastebin.com/hsvdxfbD>

Het stukje hierboven vertelt vagrant om tijdens het installeren het shell script “provision.sh” te runnen, dat shell script installeert Ansible op de virtuele machine zodat we hier tijdens deze workshop mee kunnen werken.

 .vagrant	1/9/2018 11:01 PM	File folder	
 provision.sh	1/9/2018 11:31 PM	Shell Script	1 KB
 Vagrantfile	1/9/2018 11:17 PM	File	4 KB

Maak vervolgens een nieuw bestand aan in de map waar je “Vagrantfile” staat genaamd “provision.sh” en voeg het stukje hieronder eraan toe:

```
• if [ ! -f /usr/bin/ansible-playbook ]  
• then  
• apt-get install software-properties-common  
• apt-add-repository ppa:ansible/ansible  
• apt-get update  
• apt-get install -y ansible  
• fi  
•  
• ansible-playbook --inventory="localhost," -c local /vagrant/provision/playbook.yml
```

Pastebin: <https://pastebin.com/KhLk7Ngt>

Vagrant Box

```
PS C:\vagrant-tut> vagrant up  
Bringing machine 'default' up with 'virtualbox' provider...  
==> default: Importing base box 'ubuntu/trusty64'...  
==> default: Matching MAC address for NAT networking...  
==> default: Checking if box 'ubuntu/trusty64' is up to date...  
==> default: A newer version of the box 'ubuntu/trusty64' is available! You currently  
==> default: have version '20170811.0.1'. The latest is version '20171213.0.3'. Run  
==> default: `vagrant box update` to update.  
==> default: Setting the name of the VM: vagrant-tut_default_1515537097987_43991  
==> default: Clearing any previously set forwarded ports...  
==> default: Clearing any previously set network interfaces...  
==> default: Preparing network interfaces based on configuration...  
default: Adapter 1: nat  
==> default: Forwarding ports...  
default: 22 (guest) => 2222 (host) (adapter 1)  
==> default: Booting VM...  
==> default: Waiting for machine to boot. This may take a few minutes...  
default: SSH address: 127.0.0.1:2222  
default: SSH username: vagrant  
default: SSH auth method: private key  
default:
```

Nu is het tijd om de vagrant box te gaan starten, je doet dit door het volgende commando te runnen vanuit je command line interface in de map waar je Vagrantfile staat:

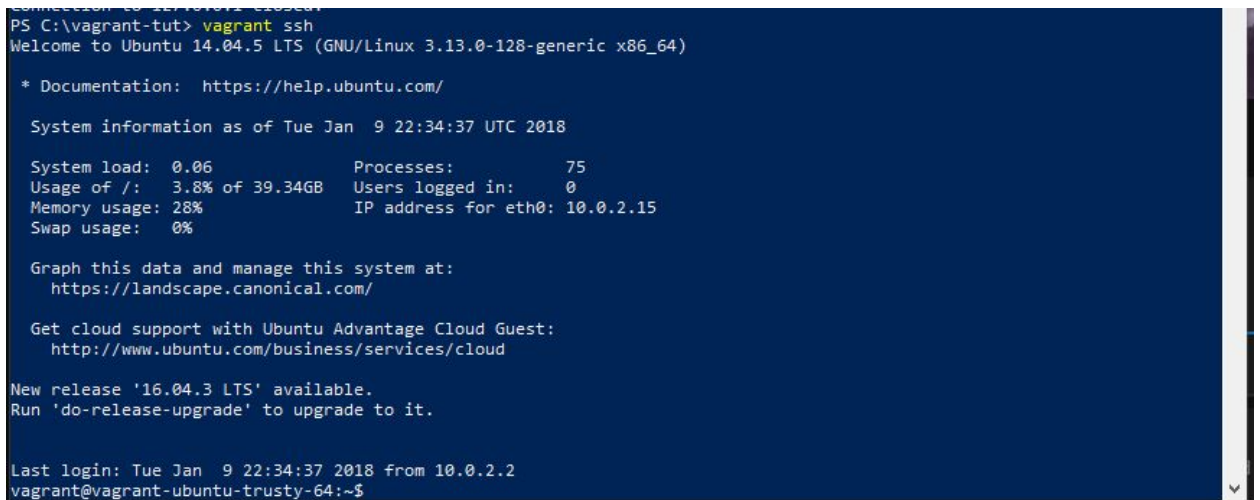
```
• vagrant up
```

Dit zal een tijdje duren aangezien Vagrant de Vagrant box moet downloaden en installeren op je lokale machine.

Nu de Vagrant box is opgestart is het tijd om in te gaan loggen op de virtual machine, gelukkig is dit door vagrant heel erg simpel gemaakt en kan je dit doen met het volgende simpele commando:

- `vagrant ssh`

Als je bent ingelogd zou je een bericht moeten zien dat er ongeveer zo uitziet:



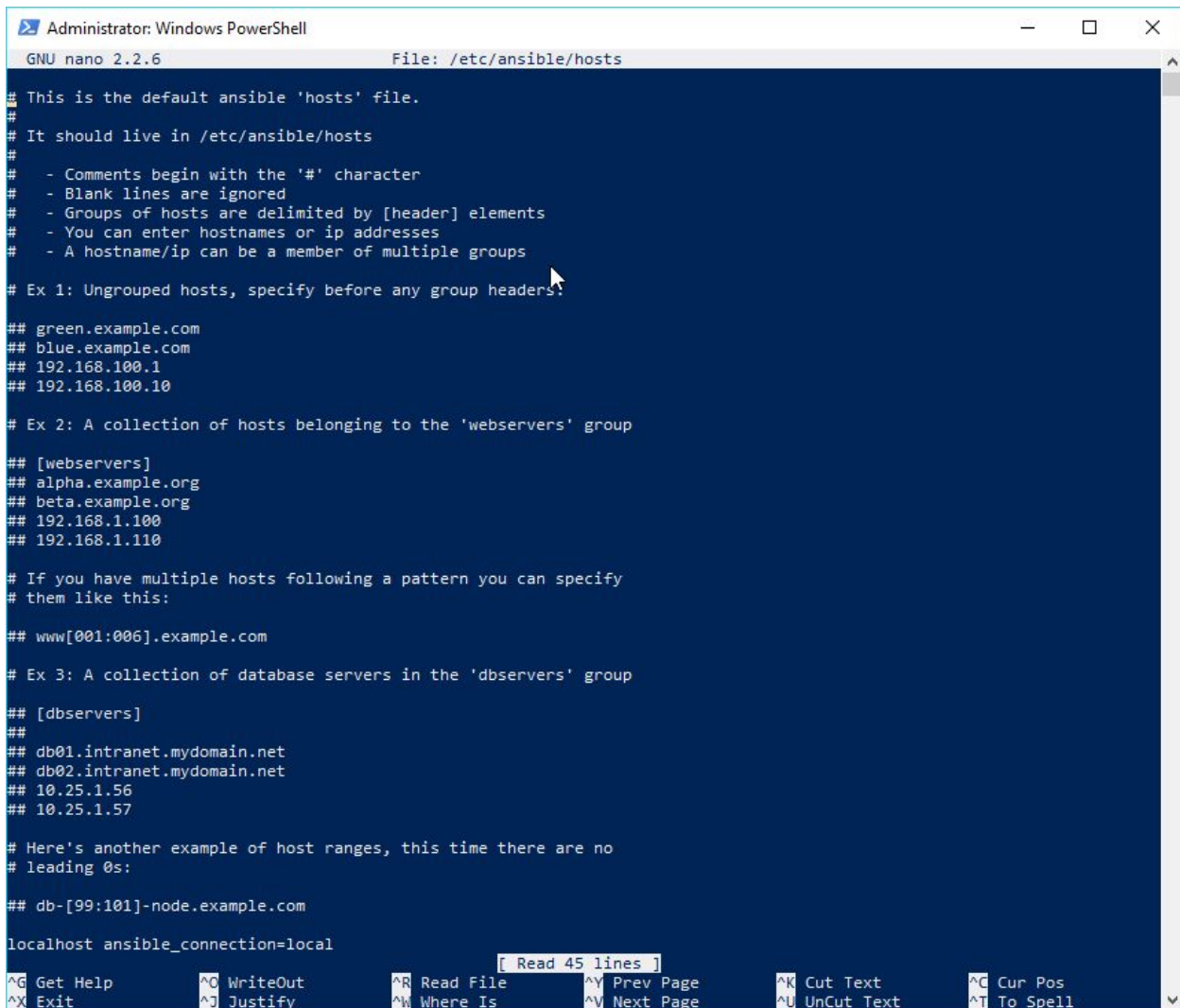
```
connection to 10.0.2.15 closed.  
PS C:\vagrant-tut> vagrant ssh  
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 3.13.0-128-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com/  
  
System information as of Tue Jan  9 22:34:37 UTC 2018  
  
System load:  0.06               Processes:            75  
Usage of /:   3.8% of 39.34GB     Users logged in:     0  
Memory usage: 28%               IP address for eth0: 10.0.2.15  
Swap usage:   0%  
  
Graph this data and manage this system at:  
https://landscape.canonical.com/  
  
Get cloud support with Ubuntu Advantage Cloud Guest:  
http://www.ubuntu.com/business/services/cloud  
  
New release '16.04.3 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Tue Jan  9 22:34:37 2018 from 10.0.2.2  
vagrant@vagrant-ubuntu-trusty-64:~$
```

Vervolgens gaan we een paar kleine dingen instellen zodat we ansible goed kunnen gebruiken. Ga met het volgende commando naar de map waar “hosts” file van ansible staat:

- `cd /etc/ansible/`

en open vervolgens het bestand in nano:

- `sudo nano hosts`



```
Administrator: Windows PowerShell
GNU nano 2.2.6      File: /etc/ansible/hosts

# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
# - Comments begin with the '#' character
# - Blank lines are ignored
# - Groups of hosts are delimited by [header] elements
# - You can enter hostnames or ip addresses
# - A hostname/ip can be a member of multiple groups
#
# Ex 1: Ungrouped hosts, specify before any group headers!
## green.example.com
## blue.example.com
## 192.168.100.1
## 192.168.100.10
#
# Ex 2: A collection of hosts belonging to the 'webservers' group
## [webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
#
# If you have multiple hosts following a pattern you can specify
# them like this:
## www[001:006].example.com
#
# Ex 3: A collection of database servers in the 'dbservers' group
## [dbservers]
##
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57
#
# Here's another example of host ranges, this time there are no
# leading 0s:
## db-[99:101]-node.example.com

localhost ansible_connection=local

[ Read 45 lines ]
^G Get Help      ^O WriteOut      ^R Read File     ^Y Prev Page     ^K Cut Text       ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is      ^V Next Page     ^U UnCut Text    ^T To Spell
```

voeg vervolgens onderstaande toe onderaan in het bestand en sluit af met “Ctrl / CMD + X” en sla het bestand op als hierom wordt gevraagd.

- localhost **ansible_connection=local**

Keer vervolgens terug naar de home directory van de virtual machine en ga daarna door naar het Ansible gedeelte van deze workshop.

- cd /home/vagrant

Ansible

Nu de vagrant box goed is geconfigureerd is het eerst even verstandig om te kijken om Ansible goed is geïnstalleerd. Als je het commando “ansible-playbook” intypt zou je iets moeten krijgen als:

```
• ansible-playbook
vagrant@vagrant-ubuntu-trusty-64:~/tutorial/playbooks$ ansible-playbook
Usage: ansible-playbook [options] playbook.yml [playbook2 ...]

Runs Ansible playbooks, executing the defined tasks on the targeted hosts.

Options:
  --ask-vault-pass      ask for vault password
  -C, --check           don't make any changes; instead, try to predict some
                        of the changes that may occur
  -D, --diff           when changing (small) files and templates, show the
                        differences in those files; works great with --check
  -e EXTRA_VARS, --extra-vars=EXTRA_VARS
                        set additional variables as key=value or YAML/JSON, if
                        filename prepend with @
  --flush-cache         clear the fact cache
  --force-handlers      run handlers even if a task fails
  -f FORKS, --forks=FORKS
                        specify number of parallel processes to use
                        (default=5)
  -h, --help           show this help message and exit
  -i INVENTORY, --inventory=INVENTORY, --inventory-file=INVENTORY
                        specify inventory host path or comma separated host
                        list. --inventory-file is deprecated
  -l SUBSET, --limit=SUBSET
                        further limit selected hosts to an additional pattern
  --list-hosts         outputs a list of matching hosts; does not execute
                        anything else
  --list-tags          list all available tags
  --list-tasks         list all tasks that would be executed
  -M MODULE_PATH, --module-path=MODULE_PATH
                        prepend colon-separated path(s) to module library
                        (default=[u'/home/vagrant/.ansible/plugins/modules',
                        u'/usr/share/ansible/plugins/modules'])
  --new-vault-id=NEW_VAULT_ID
                        the new vault identity to use for rekey
  --new-vault-password-file=NEW_VAULT_PASSWORD_FILES
                        new vault password file for rekey
```

Als je het bericht hierboven ziet staat in je Powershell of Terminal is je Ansible correct geconfigureerd en kunnen we beginnen!

Playbooks

Ansible playbooks zijn een manier om commando's te verzenden naar een remote server of computer op een geordende manier. In plaats van op elke machine losse commando's uit te voeren stap voor stap kan je nu complexe installaties in een script uitwerken en deze vervolgens op een remote machine laten uitvoeren.

Ansible playbooks zijn geschreven in het YAML data serialization format. Mocht je niet weten wat dit is, denk aan bijvoorbeeld dictionaries, arrays of lists maar dan in een makkelijk op te slaan format in een bestand op de harddisk. Het lijkt er op JSON maar is een stuk makkelijker te lezen voor mensen dan JSON files en is daarom ook makkelijker om mee te werken.

Elk playbook bestaan uit 1 of meerdere acties welke de machine vertellen wat ze moeten doen. In Ansible heet elke actie een "Task".

Standaard Playbook

Laten we is kijken naar een heel simpel playbook:

```
• ---
• - hosts: droplets
•   become: true
•   tasks:
•     - name: Installs nginx web server
•       apt: pkg=nginx state=installed update_cache=true
•       notify:
•         - start nginx
•
•   handlers:
•     - name: start nginx
•       service: name=nginx state=started
•
```

Wat je hierboven ziet is een extreem basale versie van een ansible playbook, laten we is kijken wat er werkelijk gebeurt.

Het bestand begint met:

```
• ---
```

Dit is een requirement voor YAML om herkent te worden als een goedwerkend bestand. YAML bestanden staan het toe om verschillende "documenten" in 1 bestand neer te zetten en deze te splitsen door "---" echter wordt dit niet ondersteund door Ansible.

YAML bestanden zijn erg gevoelig voor white-spaces om verschillende data te groeperen. Daarom is het verstandig om alleen maar spaties te gebruiken en geen tabs.

Items die beginnen met een "-" worden gezien als list items, items die gebruik maken van "key: value" operators worden gezien als hashes of dictionaries. Dit is hoe een basis YAML bestand is opgebouwd.

De volgende regel die we tegenkomen is:

- - hosts: droplets

Hosts zijn de servers of machines waar dit playbook op wordt uitgevoerd, deze machines zijn ingesteld in de hosts file van Ansible die je kan vinden in: `"/etc/ansible/hosts"`.

Mocht je meer informatie willen over hosts van Ansible kijk dan hier:

http://docs.ansible.com/ansible/latest/intro_inventory.html

Het stukje `"hosts"` is ook meteen het begin van de `"play"`. Dit houdt in: een combinatie van verschillende taken die zijn gegroepeerd voor deze host. Dat brengt ons bij het volgende stukje in de YAML file namelijk de tasks:

- tasks:
- - name: Installs nginx web server
- apt: pkg=nginx state=installed update_cache=true
- notify:
- - start nginx

Bovenaan hebben we een top level namelijk `"tasks"` dit vertelt Ansible dat je vanaf nu de taken gaat beschrijven die je in deze `"play"` wilt uitvoeren. Elke taak is te definiëren door middel van de `"key: value"` annotatie. Waar de `"name"` de naam aangeeft van de taak en deze kan je alles noemen wat je wilt. Vervolgens krijg je het stukje `"apt:"` dit is een referentie naar de Ansible module die je aanspreekt voor het commando.

Let er op: In de uitvoering van het commando zal Ansible eerst de `"apt-get update"` doen voordat hij het package installeert. Ook al staat `"update_cache=true"` achteraan de regel.

Wat hier eigenlijk staat is dat je eerst vraagt aan Linux een `"apt-get update"` uit te voeren met `"update_cache=true"`. Hierna zal Linux `"apt-get install nginx"` uitgevoerd worden door het `"pkg=nginx"` gedeelte. Uiteindelijk wordt er gezegd verzend de volgende notificatie uit, namelijk `"notify: - start nginx"`. De notify functionaliteit van een playbook is eigenlijk een directe referentie naar de `"handlers"` die gebruikt worden om bepaalde events aan te sturen.

- handlers:
- - name: start nginx
- service: name=nginx state=started

De handler `"start nginx"` wordt niet aangeroepen behalve als de task `"Installs nginx web server"` deze specifiek aanroept. Nu hebben we alleen een handler die nginx start. Maar wie weet heb je bij een andere task ook een handler nodig die nginx herstarten. Je kan aangeven welke status je wilt dat een service uitkomt na het uitvoeren van de handler met het `"state"` gedeelte. Voor het herstarten van nginx zou je dus de state `"restarted"` gebruiken.

Meer informatie over de states van een service kan je hier vinden:

http://docs.ansible.com/ansible/latest/service_module.html#status

Playbooks Runnen

Wanneer je een nieuw playbook hebt aangemaakt heet deze meestal iets als “nginx.yml” of “php.yml”. Wanneer je playbook af is wordt het tijd om deze te laten runnen door Ansible. Doordat we tijdens het opzetten van de Vagrant box Ansible hebben verteld om playbooks lokaal te runnen kunnen we dit heel simpel doen, namelijk met:

- `ansible-playbook nginx.yml`

Wanneer je nu dit playbook uitvoert zal je zien dat Ansible iedere task van het playbook gaat afwerken en je het resultaat zal vertellen van de task.

```
vagrant@vagrant-ubuntu-trusty-64:~/tutorial/playbooks/apache$ ansible-playbook apache.yml
PLAY [all] *****

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

TASK [Update apt] *****
changed: [localhost]

TASK [Install Apache] *****
ok: [localhost]

TASK [Create custom document root] *****
ok: [localhost]

TASK [Set up HTML file] *****
ok: [localhost]

TASK [Set up Apache virtual host file] *****
ok: [localhost]

PLAY RECAP *****
localhost                : ok=6    changed=1    unreachable=0    failed=0
```

Mocht er nou iets niet helemaal goed gaan met het laten runnen van je playbook kan je gebruik maken van de “-vvv” flag om het playbook in verbose te laten runnen. Dit zorgt ervoor dat iedere stap een melding geeft van de handelingen die worden uitgevoerd.

Opdrachten

Voor de opdrachten hieronder kan je gebruik maken van de documentatie die beschikbaar is gesteld door Ansible zelf: http://docs.ansible.com/ansible/latest/playbooks_intro.html#basics

Mocht je ergens niet uitkomen vraag het dan gerust.

Opdracht 1 (Node)

Probeer aan de hand van het voorbeeld hierboven en informatie die je kan vinden in de documentatie een playbook te schrijven waar je zowel Node als NPM installeert.

Tips:

- Maak gebruik van “apt” om verschillende dingen te installeren.
- Zorg dat je eerst een update uitvoert van de repo voordat je dingen installeert.

Opdracht 2 (Apache)

Maak voordat je begint met deze opdracht een “index.html” bestand aan in de map waar je je playbook aanmaakt. De inhoud van dit bestand mag leeg zijn aangezien je deze alleen zal gebruiken om te kopiëren door middel van Ansible.

Probeer een playbook te schrijven wat de volgende dingen doet:

- Het updaten van de repo voordat je apache installeert.
- Apache installeren.
- Het aanmaken van een custom root voor een website in “/var/www/tutorial”
- Kopieer het “index.html” bestand naar de nieuwe custom root die je hebt aangemaakt.
- Een handler die Apache herstart nadat je het bestand hebt gekopieerd.

Tips:

- Maak gebruik van het “vars” keyword om gebruik te kunnen maken van variables.
- Maak gebruik van “copy” keyword om het .html bestand te kunnen kopiëren.
- Zoek in de Ansible documentatie extra opties op om je Ansible playbook te kunnen customizen.