**SIG Terraform en Ansible**

We gaan in deze SIG een paar VM's uitrollen in de Oracle cloud, dit gaan we doen via Terraform. Op deze VM's wordt via de Terraform-scripts automatisch Ansible geinstalleerd. Op de VM's gaan we kijken wat Ansible kan.

We gebruiken voorbeelden zoals die in het dagelijks beheer kunnen voorkomen. Dat betekent dat we Terraform niet heel uitgebreid gaan bespreken, maar alleen kijken naar de onderdelen van een Terraform script. We gaan wel op basis van het aanpassen van een variabele de door terraform gecontroleerde omgeving aanpassen.

Bij Ansible gaan we, na wat afrondend installatiewerk, langzaam opbouwen hoe Ansible goed gebruikt kan worden vanuit een beheer perspectief. Na enkele commandline opties, gaan we playbooks bekijken en daarna de voorbeelden in deze SIG ook gebruiken ter inspiratie van andere opdrachten.

Zowel voor Terraform als voor Ansible maken we gebruik van de Oracle cloud met Oracle Unix VM's. Mocht je de opdrachten niet helemaal afkrijgen, besteedt dan de komende week om in een verloren uurtje of thuis de opdrachten alsnog af te maken. Mocht je hulp nodig hebben, spreek me dan aan of stuur een e-mail: frederique.retsema@amis.nl.

**1) Wat heb je nodig?**

1a) Een Oracle cloud account:

- Ga naar <http://www.oraclecloud.com> , maak hier evt. een nieuw account aan en log in.

1b) Een bestand met ID's. Maak een nieuwe directory (bijv. **D:\SIG**), maak in deze directory een nieuw tekstbestand met de naam **terraform.tfvars**. Zet in dit bestand de volgende tekst:

tenancy\_ocid = ""

user\_ocid = ""

fingerprint = ""

compartment\_id = ""

private\_key\_path = "D:\\SIG\\oci\_api\_key.pem"

region = "eu-frankfurt-1"

**N.B: er zijn meerdere plekken in meerdere config-files en commando's die gaan verwijzen naar D:\SIG. Als je een andere directory naam gebruikt, kijk dan extra kritisch naar wat er in commando's en configfiles in deze beschrijving staat en pas ook op die plekken de directorynaam aan.**

1c) Een tenancy ocid

In de Oracle Cloud heeft alles een Oracle Cloud ID, kortweg ocid.

- Log in in de Oracle Cloud, Zoek je tenancy-id:

- Kies in het linkermenu (streepjes in het menu links bovenin) voor

Administration > Tenancy details

- Je ziet nu de informatie over je account. Bij het eerste tabblad, onder Tenancy

Information staat OCID, met rechts daarnaast een linkje met "Copy".

Klik op Copy.

- Paste dit OCID in het eerder aangemaakte bestand terraform.tfvars achter

tenancy\_ocid, tussen de quotes.

1d) Om gegevens in de cloud te kunnen wijzigen, hebben we een SSH-key nodig die herkend wordt door de Oracle Cloud:

- Als je nog geen Git for Windows hebt, download en installeer dan Git for Windows

( <https://git-scm.com/download/win> ). Voor deze SIG zijn alle defaults goed.

- Start een **Bash for Windows**-sessie (klik op het vergrootglas linksonder in je scherm, typ

bash (+Enter)) en voer de volgende opdrachten uit:

openssl genrsa -out D:\\SIG\\oci\_api\_key.pem 2048

openssl rsa -pubout -in D:\\SIG\\oci\_api\_key.pem -out D:\\SIG\\oci\_api\_key\_public.pem

openssl rsa -pubout -outform DER -in D:\\SIG\\oci\_api\_key.pem | openssl md5 -c

- Dit laatste commando geeft als laatste regel bijvoorbeeld:

(stdin) = 1a:ff:12:ce:76:ba:fa:44:13:c2:d2:aa:09:62:74

- Het laatste deel (in dit voorbeeld dus: 1a:ff:12:ce:76:ba:fa:44:13:c2:d2:aa:09:62:74) wordt

wel de **fingerprint** van de key genoemd. Kopieer dit naar het **terraform.tfvars** bestand in

**D:\SIG**.

1e) Bovenstaande key is nodig om de configuratie in de cloud aan te kunnen passen. Je hebt ook nog een key nodig om aan te kunnen loggen naar de Virtual Machine. We zullen die ook in de directory D:\SIG neerzetten:

- Voer (in de Bash voor Windows-shell) het volgende commando uit:

ssh-keygen

Dit commando maakt de key aan. Het commando vraagt om de volgende gegevens:

Enter file in which to save the key (C:\Users\FRETSEM69/.ssh/id\_rsa):

Vul hier in: **D:\SIG\id\_rsa**

De volgende twee vragen: tip: houdt het simpel, vul geen passphrase in, dus 2x enter.

1f) Een user OCID

- ga in de Oracle Cloud naar je user:

- Kies in het linkermenu (streepjes in het menu bovenin, naast Oracle Cloud) voor

Identity > Users

- Je ziet nu een lijstje met gebruikers. Onder je e-mail adres staat OCID, met

daarnaast een linkje met "Copy". Klik op Copy.

- Past dit OCID in het eerder aangemaakte bestand terraform.tfvars achter user\_ocid

- Voeg in het Oracle Cloud-scherm je public key toe:

- Klik op de link met je e-mail adres

- Klik op de button "Add public key"

- Kopieer de inhoud van de file D:\SIG\oci\_api\_key\_public.pem naar dit scherm en

druk op button "Add"

- Controleer dat de fingerprint hetzelde is als die je eerder in stap 1d gekopieerd hebt naar

de terraform.tfvars file.

1g) De Oracle cloud is onderverdeeld in "compartments": het is voor ons voorbeeld handig om alles in een eigen compartment te installeren.

- Ga in het menu naar "Identity > Compartments" en maak een nieuw compartment aan,

bijv. met de naam "SIG" en vul bij description ook "SIG" in. Laat alle andere

waarden gelijk aan de default.

- Open de nieuwe compartment, copieer het OCID en zet dit in de terraform.tfvars.

We zijn nu klaar met de voorbereidingen.

**2) Terraform**

In deze SIG gaan we Terraform gebruiken om de omgeving op te zetten om Ansible in te oefenen.

Een terraform uitrol bestaat uit een aantal onderdelen:

- **variables** : de variabelen zoals die in de scripts gebruikt worden, zoals bijv. je OCID's voor

Oracle Cloud.

- **providers** : de bestemming van je objecten, in dit voorbeeld dus bijvoorbeeld de Oracle Cloud

- **datasources** : gegevens over de omgeving waar je in zit, bijv. datasources over de soorten VM's

zoals die in Oracle Cloud actueel zijn

**- resources :** objecten zoals je die aan wilt maken, bijvoorbeeld een virtual machine

- **outputs** : gegevens zoals de gebruiker van je terraform-script die nodig heeft, bijv. de

publieke IP-adressen van je virtual machines

Deze onderdelen kunnen in dezelfde file staan, of in meerdere files. Als je alle files extentie .tf geeft en in dezelfde directory zet, zal Terraform dit beschouwen als onderdeel van dezelfde uitrol. In de rest van dit voorbeeld ga ik uit van één bestand, **resources.tf**, waar ook variabelen, data objecten en output in staan.

Je kunt er voor kiezen om de variabelen die gebruikt worden iedere keer dat je een script draait met de hand in te vullen, of om de variabelen via een bestand te vullen. Omdat het niet fijn is om de OCID's telkens met de hand in te typen (of: te kopieren en te plakken), is het fijner om deze gegevens in een bestand op te slaan. Dit bestand heet vaak **terraform.tfvars**. We hebben in de voorbereiding zo'n terraforms.tfvars-bestand gemaakt.

2a) Download terraform van https://releases.hashicorp.com/terraform/0.11.13/ . Het is belangrijk deze versie te gebruiken: de nieuwste versie werkt niet goed samen met Oracle Cloud.

2b) Terraform is geunzipt één bestand. Unzip terraform.exe naar een lege directory, bijvoorbeeld de directory D:\SIG\terraform

2c) Download de bij deze SIG behorende bestanden vanuit Github:

git clone https://github.com/FrederiqueRetsema/AMIS-SIG-Terraform-en-Ansible.git

Ga naar de directory AMIS-sig-Terraform-en-Ansible en kopieer alle bestanden naar D:\SIG\terraform

Vanuit beheer gaan we vaak uit van een werkend voorbeeld. Dat doen we nu dus ook: in het **resources.tf** vind je een werkend voorbeeld.

Het is vanuit beheersperspectief verstandig om het bestand met OCID's niet in dezelfde directory te hebben staan als de bestanden met resources. De reden daarvoor is, dat als je wijzigingen incheckt in Github, dat je dan mogelijk ook de OCID's mee kopieert. In deze SIG gaan we er van uit dat je werkdirectory voor terraform één niveau lager ligt dan D:\SIG, bijvoorbeeld D:\SIG\terraform

2d) Kijk in **resource.tf**: het bestand begint met je variabelen. De eerste variabelen zijn niet ingevuld, deze variabelen hebben dezelfde naam als in terraform.tfvars. Daaronder staan een aantal variabelen met defaults. Je ziet, dat hier ook een tekst in staat die in een website gebruikt gaan worden. Die gaan we in een volgende stap gebruiken.

Mocht je directory structuur anders zijn dan hier in de defaults staat, pas de resources.tf file dan aan zodat de bestanden en directories in de defaults overeenkomen met jouw systeem.

2e) Het laatste deel van deze file bestaat uit output: dit zijn de gegevens die beheerders van deze file kunnen gebruiken nadat ze de uitrol van objecten hebben uitgevoerd. Kijk naar dit deel: je ziet dat je drie IP-adressen terug gaat krijgen.

2f) In deze file herken je de groepen zoals ze hierboven staan: variables, providers, datasources, resources, outputs. We gaan deze file gebruiken om de demo omgeving mee uit te rollen. Voor de "providers" moeten extra bestanden gedownload worden. Dit gebeurt vanaf de commandline met het commando:

terraform init

Ga (met een gewone CMD-sessie) naar de directory D:\SIG\terraform en voer dit commando uit. Na afloop van dit commando zie je in de directory waar je nu staat een extra directory met de naam **.terraform** . Het init commando hoef je maar 1x uit te voeren.

2g) Met **terraform plan** kun je bekijken wat de verschillen tussen de objecten in de terraform configuratie files en de werkelijke Oracle Cloud-omgeving. Deze lijst kun je vergelijken met wat je verwacht dat eruit gaat komen. De output wordt gebruikt door terraform apply (de volgende stap). Als je het eens bent met de default variabelen in het bestand **resources.tf**, dan kun je het volgende commando gebruiken. Voer het volgende command uit:

terraform plan -var-file=D:\SIG\terraform.tfvars -out=terraform.tfplan

Je ziet in de samenvatting dat er alleen objecten aangemaakt worden. In de output zie je soms "computed", dan is die waarde niet van te voren vanuit terraform scripts gedefinieerd. Je ziet ook aan de kleuren wat er gaat gebeuren: groene tekst betekent dat er objecten worden toegevoegd, oranje tekst betekent dat er objecten gewijzigd worden. Rode tekst betekent dat objecten verwijderd worden.

Als het goed is, zie je als output:

**Plan**: 10 to add, 0 to change, 0 to destroy.

2h) Met **terraform apply** kun je de wijzigingen ook echt uitvoeren:

terraform apply "terraform.tfplan"

Als alles goed gaat, zie je onderaan de output drie IP-adressen: een van de control-node en twee van andere nodes. Ga met een internet browser naar het IP-adres van node 1. Je ziet nu de tekst "Hello, World!".

2i) Bij stap 2e heb je gezien dat de variabele websitetext de waarde Hello, World! bevat. Dit blijkt ook de tekst te zijn die werkelijk op de website verschijnt. We gaan dit nu aanpassen, zodat we een andere tekst laten zien. Dat overrulen van een default kan, door de variabele op de commandline mee te geven:

terraform plan -var-file=D:\SIG\terraform.tfvars **-var "websitetext=Hello, SIG!"** -out=terraform.tfplan

Als je dit command uitvoert, zie je dat opnieuw bepaald wordt wat er al is en wat alsnog aangemaakt moet worden:

**Plan**: 3 to add, 0 to change, 3 to destroy

Misschien verbaast je dit, kijk even wat hij toevoegt en wat hij verwijdert en let op de kleuren. Je zult zien, dat hij niet alleen de twee nodes met websites verwijdert en opnieuw toevoegt, maar ook de control-node. De reden hiervoor is, dat de control-node een afhankelijkheid heeft van de twee andere nodes. Ansible ziet niet dat dit geen wijzigingen op de control-node tot gevolg heeft.

2j) Voer ook deze wijzigingen door:

terraform apply "terraform.tfplan"

Bewaar de IP-adressen van de control-node en de twee andere nodes.

Controleer via je browser dat de nieuwe tekst getoond wordt.

2l) Voor Terraform zijn we binnen deze SIG nu klaar. **Let op:** als je met deze SIG ophoudt, vergeet dan niet om je omgeving weg te gooien. Dit kan met terraform destroy:

terraform destroy -var-file=D:\SIG\terraform.tfvars

**Voer dit commando nu nog niet uit: de nodes die in Oracle Cloud draaien zijn de nodes die we gaan gebruiken voor het Ansible deel van deze SIG.**

**3) Ansible:**

Op de nodes die we net hebben uitgerold is al software uitgerold. Je kunt dit in de bestanden user\_data\_control.terug vinden in de bestanden user\_data\_control.sh en user\_data\_node.sh. Ik heb zoveel mogelijk geprobeerd voor te bereiden, maar sommige zaken zijn lastiger te scripten Dat deel gaan we nu nog even met de hand doen:

3a) Je hebt nu de publieke IP-adressen van de niet-beheernodes nodig. Je voert onderstaand commando **twee** keer uit (vervang 1.2.3.4 door het adres van achtereenvolgens node 1 en daarna node 2):

ssh -i ..\id\_rsa opc@1.2.3.4

Misschien krijg je nu een foutmelding dat de key te open staat. Oplossing: ga naar de file explorer, rechtermuis menu, properties, security, advanced. Controleer in dit scherm dat jij de eigenaar bent van deze file en klik daarna op de button "disable inheritance". Voeg via "Add" nu een regel toe dat jouw user full control over deze file hebt.

Pas, nadat je aangelogd bent, het wachtwoord van user "ansible" aan (gebruik een gemakkelijk password dat voor beide nodes gelijk is):

sudo passwd ansible

We gaan het in een latere stap ook onmogelijk maken om als root aan te loggen. Voer op deze twee nodes dus een nieuw rootwachtwoord op:

sudo passwd root

Na het aanpassen van de wachtwoorden log je uit met

exit

Controleer of je zonder ssh key met het net ingevoerde root password aan kunt loggen:

ssh root@1.2.3.4

3b) Ga naar de beheernode (vervang 1.2.3.4 door het publieke IP-adres van de demo\_control-node):

ssh -i ..\id\_rsa opc@1.2.3.4

Pas, nadat je aangelogd bent, ook hier het wachtwoord van user "ansible" aan:

sudo passwd ansible

Log op de controlnode aan als user ansible en ga naar de homedirectory van Ansible:

su ansible

cd

Richt binnen de beheernode ssh in:

ssh-keygen

Accepteer alle defaults.

Kijk in de file /etc/ansible/hosts naar de lokale IP-adressen van je nodes:

cat /etc/ansible/hosts

Voer voor alle nodes het volgende commando uit (vervang 1.2.3.4 en 1.2.3.5 door de werkelijke IP-adressen die je in de hosts-file ziet, onder [web]):

ssh-copy-id 1.2.3.4

ssh-copy-id 1.2.3.5

Er wordt nu een fingerprint getoond, met de vraag of je naar die node wilt connecten. Typ hier "yes".

Daarna wordt om een password gevraagd. Voer hier je ansible password in.

**Ad-hoc commando's:**

3c) Controleer dat ssh goed geinstalleerd is. Dat kun je doen door Ansible te gebruiken. In de hosts-file heb je al [web] gezien. Dat is de naam van de groep van de IP-adressen die er onder staan. Je kunt in 1x een commando naar al deze nodes sturen. Laten we gebruik maken van de ping-module van Ansible:

ansible web -m ping

Voorbeeld van output:

1.2.3.4 | SUCCES => {

"changed" : false,

"ping": "pong"

}

5.6.7.8 | SUCCES => {

"changed" : false,

"ping": "pong"

}

3d) Er is een handige module om informatie terug te krijgen van een of meerdere nodes. Pak willekeurig een van de twee IP-adressen uit de hosts-tabel en kijk hoeveel gegevens je terugkrijgt:

ansible 1.2.3.4 -m setup

Je ziet nu heel veel informatie van het systeem. Je kunt hier ook een filter op loslaten. Gebruik hiervoor de id's die je in de volledige lijst zag. Als je bijvoorbeeld van iedere node wilt weten hoeveel memory er in gebruik is:

ansible 1.2.3.4 -m setup -a "filter=\***ansible\_memory\_mb**\*"

Nu het lijstje hanteerbaarder is, kun je het ook weer toepassen op alle web nodes:

ansible **web** -m setup -a "filter=\*ansible\_memory\_mb\*"

3e) Ook losse ad-hoc commando's zijn mogelijk:

ansible web -m **shell** -a "**ls -al**"

3f) Hoewel het heel handig kan zijn om op meerdere nodes tegelijk een commando uit te voeren, krijgt Ansible pas echt meerwaarde als je meerdere commando's combineert. Een set van bij elkaar behorende commando's noemen we een "playbook". Hieronder zie je een voorbeeld, je vindt deze ook in de homedirectory van de Ansible user (en in de github repository in de demo directory):

---

- hosts: 1.2.3.4

tasks:

- name: touch bestand in homedir

file:

name: ~/bestand\_in\_homedir

state: touch

Wijzig in dit playbook het IP-adres in het IP-adres van een van beide nodes en voer daarna dit playbook uit:

ansible-playbook touch.yml

Tip: bij nieuwe playbooks is het verstandig je playbook maar naar één node uit te rollen. Als alles werkt, stuur je het playbook naar alle nodes in de groep.

Controleer nu, met een van de eerdere commando's, dat op één van beide nodes het bestand is aangemaakt en op de ander niet.

- Als het werkt, wijzig je playbook dan zo dat je het playbook naar web stuurt in plaats van naar 1 IP-adres.

- We hebben hier gebruik gemaakt van de file-module. Je kunt deze gebruiken voor het aanmaken, verwijderen of touchen van file. Je kunt er ook links mee aanmaken, of de owner of de group aanpassen. Meer info: [https://docs.ansible.com/ansible/latest/modules/file\_module.html#file-module](https://docs.ansible.com/ansible/latest/modules/file_module.html" \l "file-module)

3g) We gaan nu een playbook maken dat er voor zorgt dat de aliases van de ansible user goed bijgewerkt worden:

---

- hosts: 1.2.3.4

tasks:

- name: Bijwerken .bash\_profile met aliases

lineinfile:

path: /home/ansible/.bash\_profile

state: present

line: "alias ch='cat /etc/hosts'"

Voer ook dit playbook eerst uit op één node. Je ziet, net als de vorige keer, dat er eerst gegevens verzameld worden ("Gathering Facts"), hieronder staat een groen IP-adres. Dat betekent dat er niets op deze node gewijzigd is. Daaronder zie je de naam van de wijziging, bij ons dus "Bijwerken .bash-profile met aliases". Deze is oranje, hij heeft dus iets aangepast.

- Voer ditzelfde playbook nog een keer uit: je ziet dat het bijwerken van de .bash-profile geen tweede keer heeft plaatsgevonden. Dat is een belangrijk uitgangspunt van playbooks: in principe moet een playbook zo geschreven worden dat hij meerdere keren uitgevoerd kan worden, waarbij hij alleen de eerste keer iets wijzigt.

- Log aan naar de node (je kunt dit doen vanaf de control node met ssh 1.2.3.4)

- Met het commando alias kun je zien of de alias bestaat, met cat ~/.bash\_profile kun je zien dat deze regel werkelijk is toegevoegd aan de .bash\_profile.

- Je kunt via state=absent ook regels verwijderen. Zie voor alle parameters van lineinfile: <https://docs.ansible.com/ansible/latest/modules/lineinfile_module.html#lineinfile-module>

3h) Tijd om zelf een playbook te maken! De nodes in je test omgeving zijn redelijk out-of-the-box opgeleverd. Als je bijvoorbeeld met history naar de afgelopen commando's kijkt, dan zie je geen date/time van wanneer deze commando's gegeven zijn. Opdracht: maak een Ansible playbook dat er voor zorgt dat bij de history een datum en tijd meegegeven wordt. Zie ook deze link: <https://www.cyberciti.biz/faq/unix-linux-bash-history-display-date-time/>

3i) Je kunt ook meerdere taken na elkaar bijwerken. In het volgende playbook gaan we er voor zorgen dat mensen niet meer met root kunnen aanloggen. Er worden drie zaken geregeld:

- Er wordt een groep sudoers aangemaakt

- De sudoers groep moet toegevoegd worden aan de sudoers file

- De rootuser wordt disabled

Al deze acties dienen op de andere node uitgevoerd te worden als root. Daarvoor is een sudo nodig, in ansible-playbook heet dit become:

---

- hosts: 1.2.3.4

**become: yes**

tasks:

- name: Maak groep sudoers aan

group:

name: sudoers

state: present

- name: Voeg regel toe aan de sudoers file, alle commando's toegestaan

lineinfile:

path: /home/ansible/test

state: present

create: yes

line: "%sudoers\tALL=(ALL)\tALL"

- name: Disable de root user

user:

name: root

state: present

password\_lock: yes

**Belangrijk:** je ziet dat de extra regel in de test file in de homedirectory van Ansible aangemaakt is.Als dit gelukt is, ga dan met ssh naar deze node toe en controleer of het toevoegen van deze regel aan de sudoers file geen problemen oplevert. Dit kun je doen door:

- sudo visudo

- lees de regel in uit /home/ansible/test en voeg deze toe aan de sudoers file

- Schrijf de sudoers file weg

- Als je hier geen foutmelding op krijgt, dan is de nieuwe regel geldig

- Als je hier wel een foutmelding op krijgt, pas de regel dan aan (eerst binnen sudoers,

daarna in het playbook) totdat dit wel werkt.

- Als het playbook op deze manier werkt, pas (pas) dan het playbook aan zodat deze

/etc/sudoers aanpast ipv /home/ansible/test.

- Test het aangepaste playbook op één node

- Log opnieuw aan naar deze node, test met sudo ls of sudo nog steeds werkt.

Doe je dit niet en pas je direct de /etc/sudoers file aan, dan zul je merken dat als je een foutje gemaakt hebt in de regel die toegevoegd moet worden, dat je geen sudo meer kunt doen. Als je root dan ook gedisabled hebt, dan kun je in feite niets anders dan je node weggooien en een backup terug te zetten...

Voer bovenstaand playboek en de beschreven werkwijze voor het testen van de sudoers-file uit. Mocht je onverhoopt toch een sudoers-file onbruikbaar maken, maak dan gebruik van de tweede web node. Mocht je ook die kwijt raken, voer dan de terraform commando's voor het weggooien van je omgeving uit en maak via terraform je omgeving daarna opnieuw aan.

Documentatie: module group: [https://docs.ansible.com/ansible/latest/modules/group\_module.html](https://docs.ansible.com/ansible/latest/modules/group_module.html%23group-module)

Documentatie: module user: <https://docs.ansible.com/ansible/latest/modules/user_module.html>

**N.B**: onder Oracle Unix zou het ook voldoende kunnen zijn om gebruikers aan de (al bestaande) groep wheel toe te voegen. In dat geval gebruik je alleen de taak om de root user te disablen. Ik vind zelf sudoers als groep duidelijker weergeven wat het doel is van de groep. Het maakt het eventueel ook mogelijk om wheel te gebruiken voor beheerders en sudoers voor eindgebruikers en sudoers minder rechten te geven.

3j) Met de users module kun je ook gebruikers toevoegen en in groepen zetten. Opdracht: voeg gebruiker marieke via het playbook van de vorige opdracht toe aan een web node en geef Marieke rechten voor de sudoers groep. Controleer de wijzigingen door in te loggen op je web node, wijzig het wachtwoord van Marieke en log aan als Marieke. Probeer als Marieke een sudo te doen en controleer dat je oplossing werkt.

3k) Je hebt nu een aantal playbooks gezien en ook de documentatie al een paar keer bekeken. In deze opdracht willen we de yum module gebruiken om SNMP uit te rollen op de web nodes. Tips:   
 - gebruik **yum search** **snmp** om achter de naam van het SNMP package te komen.

- gebruik het zoekvak linksboven in de Ansible documentatie om de pagina van de yum

module te zoeken. Maak het niet te moeilijk: je hebt maar twee parameters van deze

module nodig om dit te laten werken.

- Controleer na het uitvoeren van het playbook op de betreffende node met

**yum list installed | grep snmp** of het installeren ook gelukt is.

- Voer hetzelfde commando ook uit op de andere node en constateer dat je hier andere

output hebt.

3l) Soms is het handig om een scriptje te schrijven wat je op meerdere nodes uit wilt voeren. We zullen in deze opdracht een scriptje maken op de control node en dit uit gaan voeren op beide web nodes:

- Maak een script aan op de control node met de volgende inhoud:

#! /bin/bash

echo "Naam van deze machine: $HOSTNAME"

echo "SSH connectie: $SSH\_CONNECTION"

echo "Tijdstip: `date`"

- Voer dit script (met de hand, zonder Ansible te gebruiken) op de control node uit,

controleer dat de variabelen netjes worden ingevuld

Maak daarna een playbook dat met behulp van de copy module dit script overzet. Voer het script

(remote) uit via de shell module. Gebruik het command "script.sh > output.txt"

Controleer je script door na het uitvoeren van je playbook op je node de inhoud van output.txt te

bekijken.

**Belangrijk:** als je script op de node waarop je het script uitvoert bestanden aanpast, zorg er dan

voor dat je commando maar éénmalig iets aanpast, ook als je script meer dan eens uitgevoerd

wordt. Controleer dit eerst op een testnode door je playbook meer dan 1x uit te voeren. Het

resultaat zou op je testnode nog steeds goed moeten zijn.

**Controle:** Voer het playbook voor de tweede keer uit. Je ziet dat het schellscript ook de tweede

keer uitgevoerd wordt, controleer dat de inhoud maar eenmalig in output.txt staat.

**Belangrijk: Voer na afloop van de SIG het destroy commando van terraform uit, zie de laatste opdracht van hoofdstuk 2.**

**4) Nuttige links**

Lijst met alle modules: <https://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html>

Modules per categorie: <https://docs.ansible.com/ansible/latest/modules/modules_by_category.html>