Configuration centralisée de serveurs et de services

TP1

COLLATY Srikanth
KOBEISSI Bessel

Fabieng <fabieng@flexcorp.fr>

Objet : Bienvenue chez FlexCorp !

Bonjour *Bessel et Srikanth,*

Bienvenue chez FlexCorp !

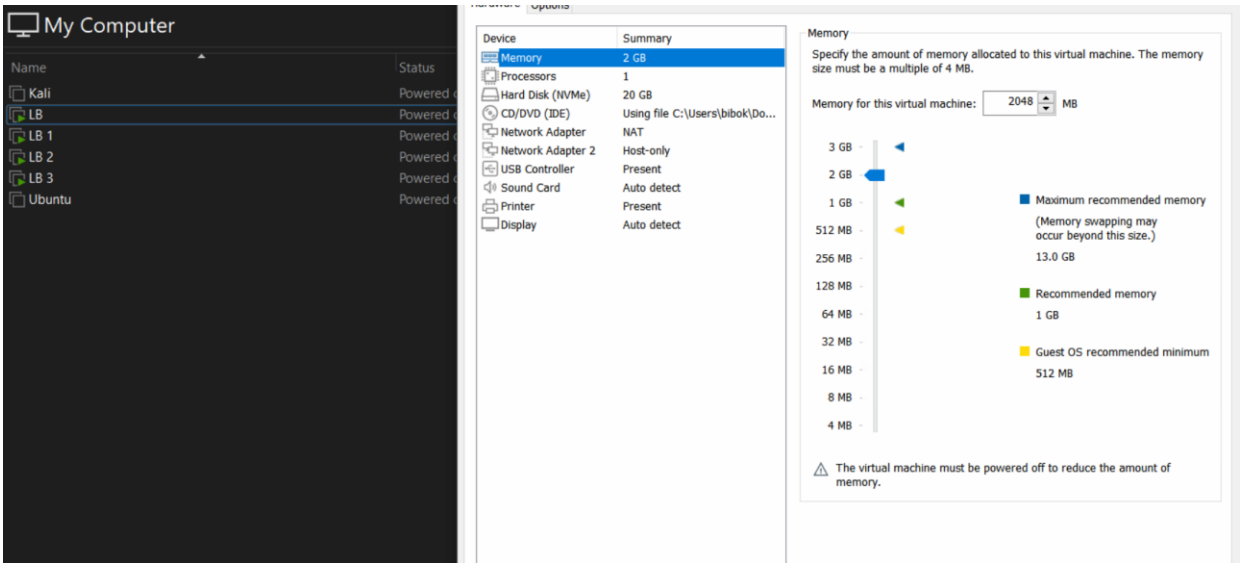
Comme évoqué tout à l'heure, j'ai urgemment besoin que vous me déployiez des serveurs pour une démo client demain matin. Le client va surement essayer de se connecter en masse, donc il me faut un loadbalancer et tout le tintouin habituel. Je compte sur vous.

A très bientôt au babyfoot !

Fabieng
Chief Chef Manager

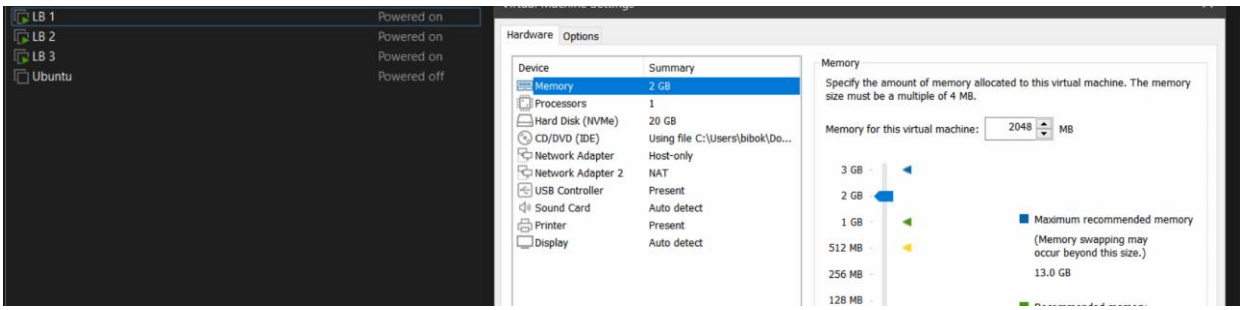
Travail à effectuer

1. Créez une machine virtuelle CentOS que vous nommerez « LB » (load balancer) avec 2 cartes réseau : une capable de joindre internet et l'autre dédiée à un réseau privé interne.



Nous avons créé la machine virtuelle avec 2 cartes réseau (NAT et Host-only)

2. Créez 3 machines virtuelles CentOS connectées uniquement au réseau privé interne. Elles doivent pouvoir utiliser la machine « LB » comme passerelle internet.



Nous avons cloné « LB » en 3 machines virtuelles « LB 1 », « LB 2 » et LB 3 »

3. Sur votre machine « LB », installez NGINX et vérifiez qu'il soit bien accessible (depuis votre ordinateur)

Serveur NGINX bien installé et accessible depuis notre ordinateur

4. Sur votre machine « LB », installez Ansible et ajoutez à l'inventaire par défaut vos 3 machines dans un groupe que vous pourrez nommer « webservers » par exemple

```
[bessel@localhost setup]$ ls
hosts
[bessel@localhost setup]$ nano webservices.yml
[bessel@localhost setup]$ cat webservices.yml
[webservices]
127.0.0.1
192.168.59.132
192.168.48.128
192.168.48.129
192.168.48.130
```

Ajout de l'inventaire webservices

5. Configurez l'accès SSH pour qu'Ansible puisse se connecter à vos 3 machines. Vous pouvez simplement écrire en dur vos mots de passe dans votre configuration, nous nous attarderons plus tard sur la sécurisation des accès

```
[bessel@localhost ~]$ sudo yum install openssh-clients
[sudo] Mot de passe de bessel :
Dernière vérification de l'expiration des métadonnées effectuée il y a 0:57:00 le lun. 14 nov. 2022 10:47:09 CET.
Le paquet openssh-clients-8.0p1-16.el8.x86_64 est déjà installé.
Dépendances résolues.
Rien à faire.
Terminé !
[bessel@localhost ~]$ sudo yum install openssh-server
Dernière vérification de l'expiration des métadonnées effectuée il y a 0:57:21 le lun. 14 nov. 2022 10:47:09 CET.
Le paquet openssh-server-8.0p1-16.el8.x86_64 est déjà installé.
Dépendances résolues.
Rien à faire.
Terminé !
[bessel@localhost ~]$ sudo systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2022-11-14 10:19:46 CET; 1h 25min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1118 (sshd)
    Tasks: 1 (limit: 11047)
   Memory: 1.2M
   CGroup: /system.slice/sshd.service
           └─1118 /usr/sbin/sshd -D -oCiphers=aes256-gcm@openssh.com,chacha20-poly1305@openssh.com,aes256-ctr,aes256-cbc,aes128-gcm@openssh.com,aes128-ctr,aes12

nov. 14 10:19:46 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
nov. 14 10:19:46 localhost.localdomain sshd[1118]: Server listening on 0.0.0.0 port 22.
nov. 14 10:19:46 localhost.localdomain sshd[1118]: Server listening on :: port 22.
nov. 14 10:19:46 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
lines 1-15/15 (END)
```

Installation de SSH et vérification de son état

```
[bessel@localhost ~]$ sudo systemctl start sshd
[bessel@localhost ~]$ sudo systemctl enable sshd
[bessel@localhost ~]$ sudo systemctl disable sshd
Removed /etc/systemd/system/multi-user.target.wants/sshd.service.
[bessel@localhost ~]$ sudo systemctl stop sshd
[bessel@localhost ~]$ sudo systemctl restart sshd
[bessel@localhost ~]$ ssh bessel@192.168.48.128
The authenticity of host '192.168.48.128 (192.168.48.128)' can't be established.
ECDSA key fingerprint is SHA256:KyEInuefANsJMmINyUJENSxg090vBPbw+KMB2MDfA+I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.48.128' (ECDSA) to the list of known hosts.
bessel@192.168.48.128's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Mon Nov 14 10:59:45 2022
[bessel@localhost ~]$ ls
Bureau  Documents  Images  Modèles  Musique  Public  Téléchargements  Vidéos
[bessel@localhost ~]$ exit
déconnexion
Connection to 192.168.48.128 closed.
[bessel@localhost ~]$
```

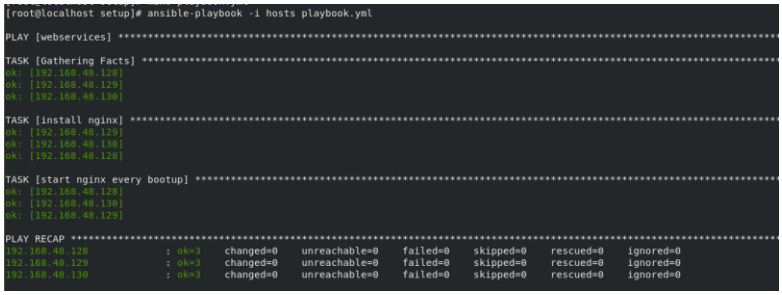
Connexion en SSH entre « LB » et « LB 1 » par exemple

```
- hosts: webservices

  user: root

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes
```

Playbook.yml

```
[webservices]
192.168.48.128
192.168.48.129
192.168.48.130

[all:vars]
ansible_connection=ssh
ansible_user=root
ansible_ssh_pass=admin
```

Webservices.yml

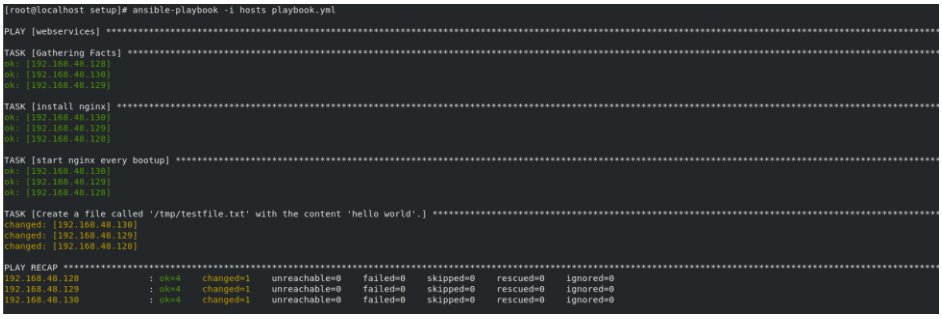


Nous pouvons apercevoir que l'exécution de playbook fonctionne bien

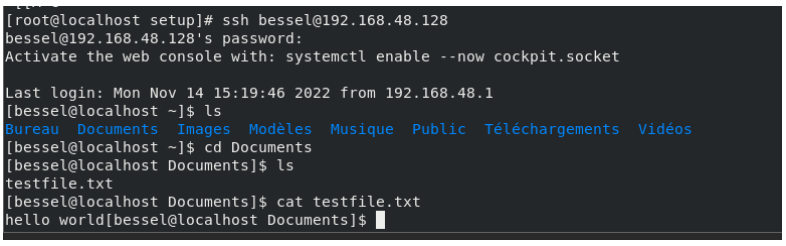
6. Concevez et exécutez un playbook Ansible de test ayant pour but d'écrire « Hello World ! » dans un fichier texte quelconque sur l'ensemble des 3 machines

```
- hosts: webservices

  user: root

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes

    - name: Create a file called '/tmp/testfile.txt' with the content 'hello world'.
      copy:
        content: hello world
        dest: /home/bessel/Documents/testfile.txt
```

Playbook.yml mis à jour avec l'ajout d'hello world dans un document texte et deployé sur chaque machine



On peut apercevoir que des changements ont été effectués

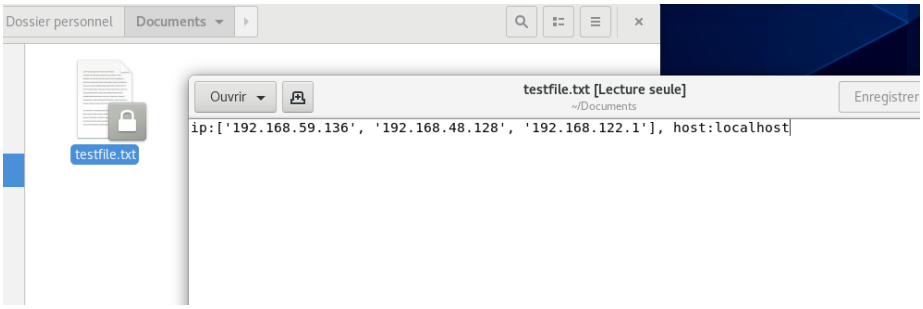


En lisant le contenu de testfile.txt, on peut remarquer « hello world »

7. Concevez et exécutez un playbook Ansible ayant pour but d'écrire des informations à propos des machines (par exemple leur nom, leur adresse IP, etc) dans un fichier texte quelconque sur l'ensemble des 3 machines

```
- hosts: webservices

  user: root

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes


    - name: store info
      copy:
          content: "ip:{{ansible_all_ipv4_addresses}}, host:{{ansible_hostname}}"
          dest: /home/bessel/Documents/testfile.txt
```

Playbook.yml mis à jour avec la récupération des adresses ip et hostname des machines virtuelles



Aperçu du fichier testfile.txt avec les modifications attendues

8. Concevez et exécutez un playbook Ansible ayant pour but de mettre à jour tous les paquets yum de l'ensemble des 3 machines

```
- hosts: webservices

  user: root

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes


    - name: upgrade
      ansible.builtin.yum:
        name: '*'
        state: latest
```

Mise à jour du playbook.yml en mettant à jour les paquets yum sur les 3 machines

9. Mettez en place un moyen sécurisé de stocker les informations d'accès à vos serveurs pour Ansible. Vous êtes libre de trouver le moyen qui vous semble le plus adapté. Vous devez sur votre rapport justifier ce choix (et pourquoi pas les autres).

```
[root@localhost setup]# nano secret_api_key.yml
[root@localhost setup]# ansible-vault encrypt secret_api_key.yml
New Vault password:
Confirm New Vault password:
Encryption successful
[root@localhost setup]# cat secret_api_key.yml
$ANSIBLE_VAULT;1.1;AES256
343930363261313061306531646334396538343335643164346431363133643232313032363136 63
383863376134643938343561366463633034623861643333 0a303764336432396461646661363665
613934353630656564366533663064373661373134663739366465316164383530613766306263 65
366336653239326664 0a323931326231666233376563376664646666333662643439663665353031
396231323565353030636231303039376233323861313830326339363733339663733
[root@localhost setup]# ansible-vault view secret_api_key.yml
Vault password:
api_key: dEZSY!3P22Pc#s
[root@localhost setup]#
```

Nous pouvons voir qu'en voulant voir le contenu de la clé avec cat, son contenu est crypté

➔ Nous avons fait le choix d'utiliser Ansible Vault car il utilise l'algorithme AES256 et qui fournit un cryptage symétrique qui est connu pour avoir une mise en place assez simple et simple à utiliser. Il est également développé par Ansible ce qui est intéressant vu le TP sur lequel nous travaillons.

10. Concevez et exécutez un playbook Ansible ayant pour but d'installer NGINX et d'activer le service lié

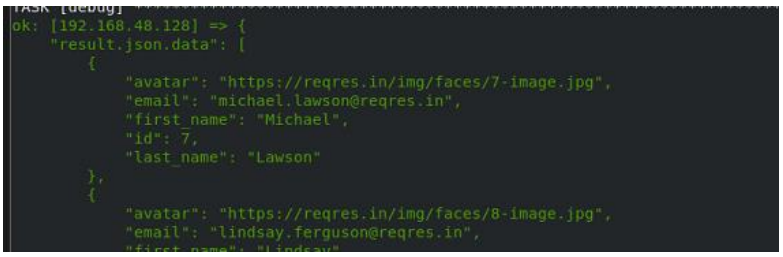
```
- hosts: webservices

  user: root

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes
```

Configuration pour que NGINX s'installe et s'active sur toutes nos machines

```
[root@localhost setup]# ansible-playbook -i hosts playbook.yml

PLAY [webservices] *************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************
ok: [192.168.48.130]
ok: [192.168.48.129]
ok: [192.168.48.128]

TASK [install nginx] ***********************************************************************************************************************************
ok: [192.168.48.130]
ok: [192.168.48.128]
ok: [192.168.48.129]

TASK [start nginx every bootup] ************************************************************************************************************************
ok: [192.168.48.130]
ok: [192.168.48.129]
ok: [192.168.48.128]

PLAY RECAP *********************************************************************************************************************************************
192.168.48.128             : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.48.129             : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.48.130             : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@localhost setup]#
```

Vérification

11. Concevez et exécutez un playbook Ansible ayant pour but d'ajouter les règles de sécurité nécessaires sur vos 3 machines afin que la machine « LB » soit autorisée à leur envoyer des requêtes HTTP et qu'elles puissent répondre.



Réponse de « LB 1 »

Réponse de « LB 2 »

Réponse de « LB 3 »

```
- hosts: webservices

  user: root

  become: false
  vars:
    server: "https://reqres.in"
    endpoint: "/api/users?page=2"

  tasks:
    - name: install nginx
      yum: pkg=nginx state=present

    - name: start nginx every bootup
      service: name=nginx state=started enabled=yes

    - name: HTTP query
      ansible.builtin.uri:
        url: "{{ server }}{{ endpoint }}"
        method: GET
        status_code: 200
        timeout: 30
      register: result

    - name: debug
      ansible.builtin.debug:
        var: result.json.data
```

Playbook mis à jour pour permettre d'envoyer des requêtes http et ainsi recevoir une réponse des 3 machines

Dans notre cas : récupérer la liste des users avec des informations comme le prénom, mail, etc..

Source : https://www.ansiblepilot.com/articles/submit-a-get-request-to-a-rest-api-endpoint-interact-with-web-services-ansible-module-uri/

12. Sur votre machine « LB », configurez NGINX en mode reverse proxy avec load balancer ayant comme destination vos 3 machines

    Source trouvée pour mettre en place le serveur proxy mais faute de temps, cela n'a pas pu être possible : https://rdr-it.com/en/nginx-configuration-as-reverse-proxy/

13. Concevez et exécutez un playbook Ansible ayant pour but de remplacer sur la page web de test NGINX « Red Hat Enterprise Linux » par l'adresse IP (ou le nom) du serveur qui traite la réponse sur vos 3 machines. Vérifiez le bon fonctionnement en tentant d'accéder plusieurs fois de suite à votre site avec votre ordinateur.

14. BONUS : Créez une machine virtuelle CentOS 8 et installez et configurez dessus AWX pour gérer Ansible.