

# Documentation - Maquette technique

## Introduction

La maquette produite et déployée lors de l'oral de CCNP, portant sur le thème "Virtualization, Automation and Programmability", repose sur l'utilisation conjointe d'Ansible et de Docker pour développer une infrastructure informatique minimaliste.

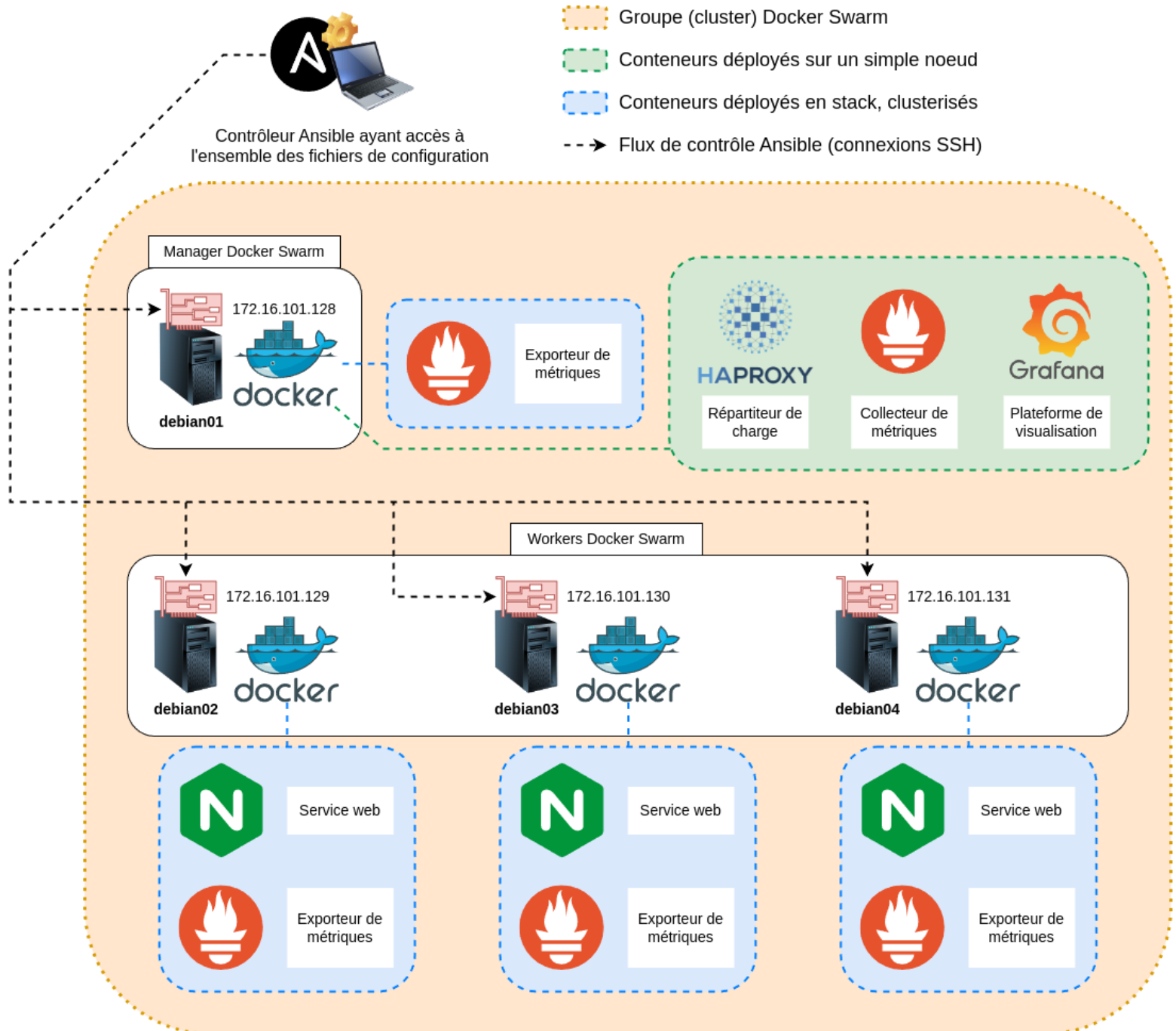
### 1. Description de l'infrastructure

#### 1.1. Systèmes, services et rôles

Système	Rôles
Machine physique	Contrôleur Ansible. Dispose d'un accès SSH à chaque VM pour l'exécution d'instructions par Ansible. Héberge : <ul style="list-style-type: none"><li>- Hyperviseur de type 2 pour gérer les VMs.</li><li>- Fichiers de configuration des services déployés, fichiers Docker Compose/Stack et playbooks Ansible.</li></ul>
VM1	Manager Docker Swarm, orchestrateur central des services déployés par Docker. Héberge : <ul style="list-style-type: none"><li>- HAProxy, répartiteur de charge pour des services web.</li><li>- Prometheus, pour remonter des métriques d'état des ressources sur chaque système.</li><li>- Grafana, pour la visualisation des métriques remontées par Prometheus.</li><li>- Node Exporter Prometheus, pour la remontée de métriques.</li></ul>
VM2, VM3, VM4	Worker Docker Swarm, nœud équilibrant la charge des services applicatifs. Héberge : <ul style="list-style-type: none"><li>- Nginx, instance de service web en backend du service HAProxy.</li><li>- Node Exporter Prometheus pour la remontée de métriques.</li></ul>

L'orchestrateur (VM1, Manager Docker Swarm) a pour rôle principal de gérer et superviser le cluster Docker, en prenant des décisions liées au déploiement des conteneurs, à la planification des tâches, et à la gestion de la haute disponibilité.

## 1.2. Schéma de l'infrastructure



### 1.3. Connectivité réseau nécessaire

Afin de faire fonctionner Docker Swarm, les machines virtuelles devront pouvoir communiquer entre elles sur les ports suivants :

1. TCP 2377, pour des opérations de gestion du cluster.
2. UDP 4789, pour la communication des réseaux overlay créés et gérés par Docker.
3. UDP 7946 et TCP 7946, pour la communication entre nœuds du cluster.

## 2. Etapes de développement de l'infrastructure

Le développement de l'infrastructure à déployer sera décomposé en plusieurs étapes, afin de bien observer la contribution de chaque technologie de virtualisation, d'automatisation ou de programmabilité exploitée au cours d'un cycle de déploiement de l'infrastructure.

Ce déploiement sera alors rythmé par divers jeux d'instructions Ansible (playbooks) :

1. Installation de Docker sur chaque système, configuration initiale de chaque système.
2. Récupération des images nécessaires aux conteneurs déployés sur chaque système.
3. Création du cluster Docker Swarm : liaison des instances de Docker hébergées sur chaque système et répartition des rôles Manager et Worker dans le cluster.
4. Transmission des fichiers Docker Compose/Stack et des fichiers de configuration des services déployés au Manager du cluster Docker.
5. Déploiement de l'ensemble des services (clusterisés ou non) par l'intermédiaire du Manager.
6. Création d'un compte de service privilégié Grafana avec clé d'API pour importer/exporter de manière automatisée des sources de données et graphes dans Grafana.

Un script Bash, exécuté depuis le contrôleur Ansible, viendra achever le déploiement de la maquette, en chargeant une source de données (correspondant au service Prometheus déployé) et un graphe préconfiguré exploitant cette même source de données dans Grafana.

### 3. Fichiers nécessaires

Les fichiers nécessaires au déploiement de la maquette sont fournis dans un fichier ZIP, contenant l'arborescence suivante :

- Dossier **docker** : contient l'ensemble des fichiers YAML exploités par Docker Compose/Stack, permettant de déployer des services avec Docker depuis le Manager.
- Dossier **ansible** : contient l'ensemble des jeux d'instruction exploités par Ansible, exécutés par le contrôleur Ansible.
- Dossier **services** : contient l'ensemble des fichiers et scripts de configuration liés aux services déployés avec Docker.
- Dossier **keys** : contient les clés publiques et privées utilisées pour communiquer par SSH entre le contrôleur Ansible et les VMs. Sur chaque VM, la clé publique correspondante sera enregistrée dans le répertoire de l'utilisateur chargé d'exécuter des instructions reçues par Ansible (fichier *authorized\_keys* sous Linux).

Les disques durs des VMs ne seront pas fournis, en raison du volume conséquent de ces fichiers et de leur capacité à être reproduits simplement par la création de VMs et la réinstallation de systèmes d'exploitation.

Pour être certain que le déploiement de la maquette soit correctement réalisé, il sera nécessaire de conserver l'arborescence initiale du fichier ZIP.

## Paramétrage initial

### 1. Hyperviseur

Les étapes de paramétrage initial effectuées dans l'hyperviseur, consistant à créer les VMs et leur permettre de communiquer entre elles, différeront légèrement selon l'hyperviseur choisi pour déployer la maquette. Nous avons choisi d'utiliser l'hyperviseur VMware Workstation dans cette documentation.

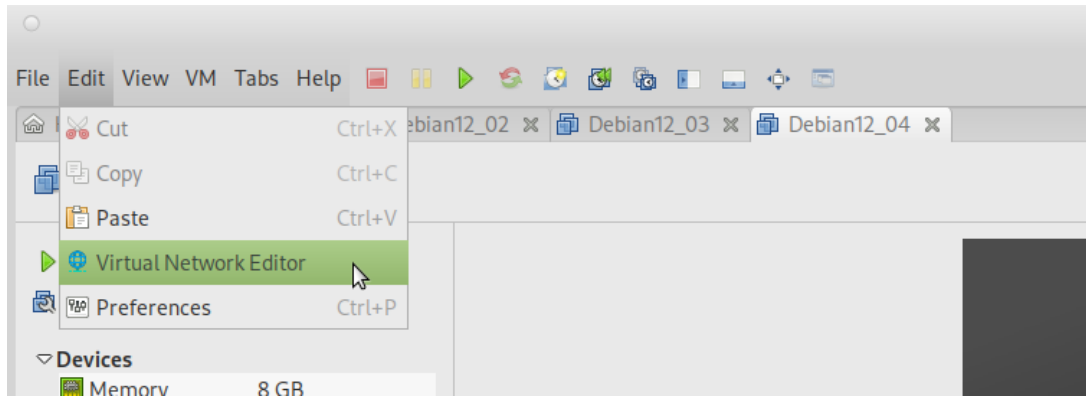
#### 1.1. Création d'un réseau virtuel avec NAT

Il nous faut d'abord disposer d'un réseau virtuel permettant aux hôtes y étant rattachés d'aller sur Internet, par un mécanisme de NAT. On ajoutera sur ce réseau les systèmes virtuels déployés, afin que ceux-ci puissent télécharger les éléments logiciels nécessaires

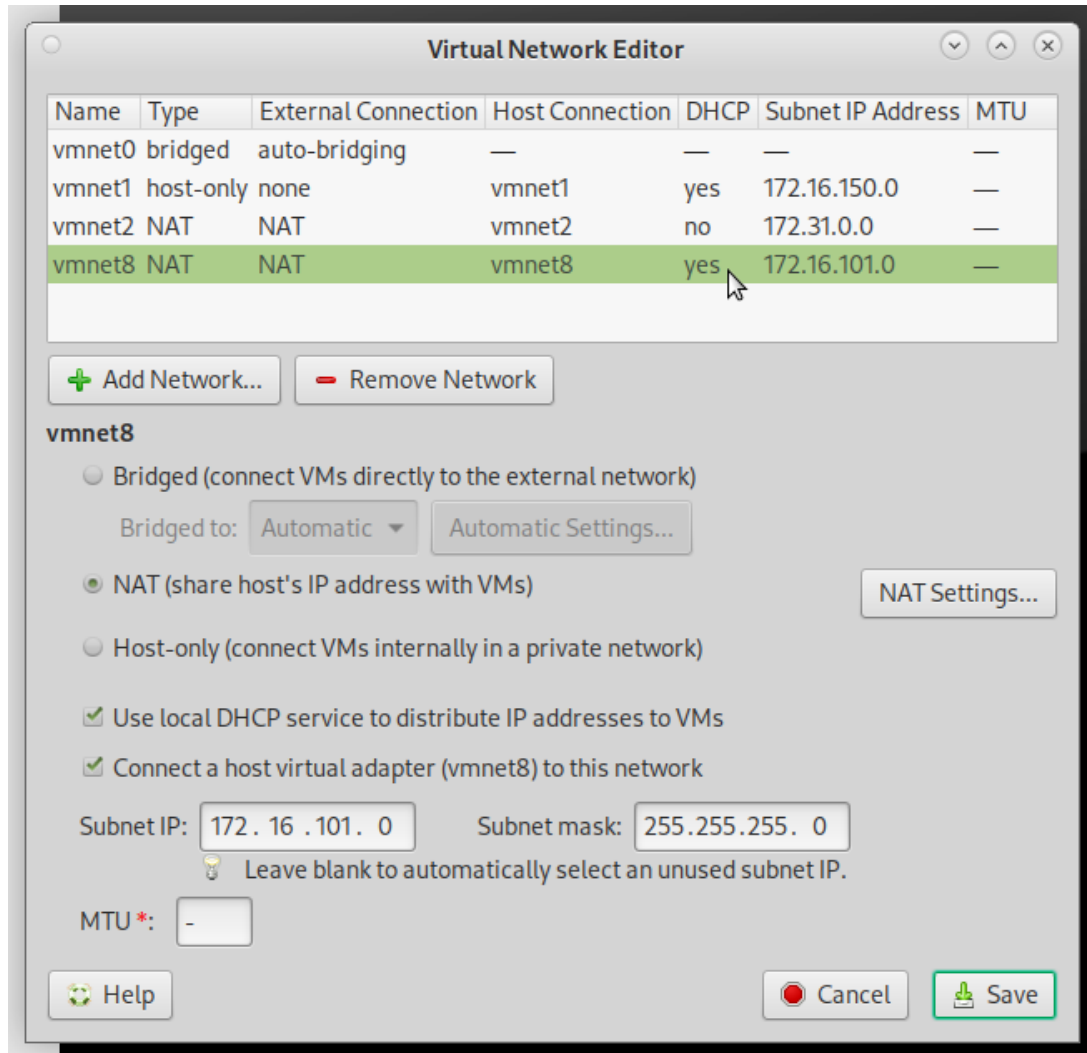
au déploiement de la maquette tout en étant capables de communiquer simplement entre eux.

Depuis VMware Workstation :

1. Aller dans le Virtual Network Editor :



2. Créer ou identifier un réseau virtuel NAT qui sera utilisé pour le déploiement de la maquette :

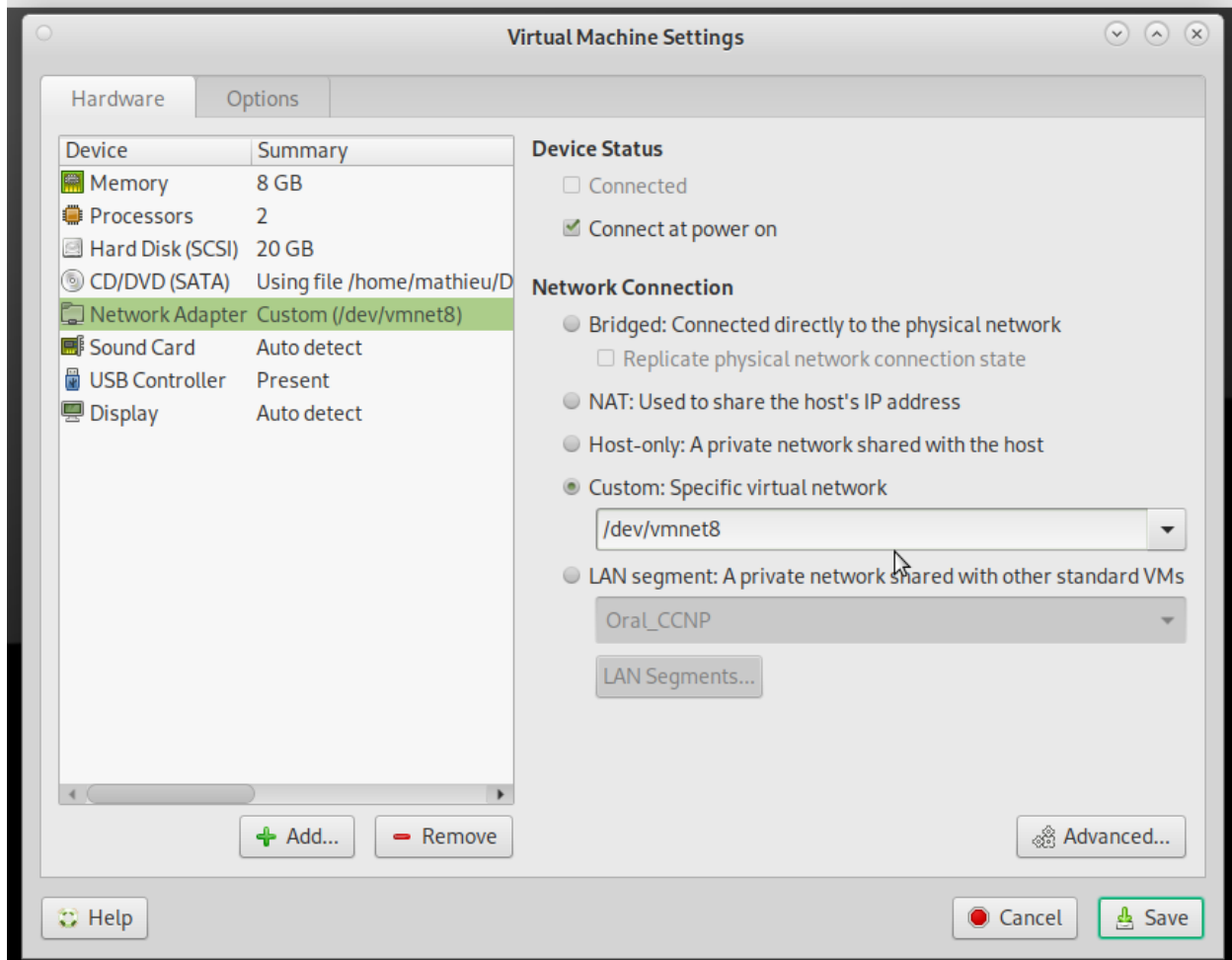


## 1.2. Création des VMs

Des systèmes Debian 12, stables, sont totalement adaptés. L'essentiel sera de choisir un système d'exploitation Unix/Linux, incluant le noyau nécessaire aux services déployés par la suite. Ce système devra alors être capable d'installer et d'utiliser Docker, et plus particulièrement Docker Engine, qui constitue les fonctionnalités logicielles essentielles de Docker, exploitables directement en lignes de commandes.

Voir les [instructions d'installation](#) pour chaque système supporté par Docker Engine.

A la création des systèmes, on ajoutera dans leurs paramètres une de leurs interfaces réseau dans le réseau virtuel configuré/identifié précédemment, pour qu'ils disposent d'un accès à Internet :



A l'installation des systèmes, on spécifiera :

- Comme utilisateur : *ansible*
- Selon la présence d'un serveur DHCP ou non dans le réseau virtuel utilisé, un adressage statique ou dynamique par DHCP.
- L'installation d'un serveur SSH, qui sera utilisé par Ansible pour communiquer avec le système virtualisé.

## 2. Paramétrage initial des systèmes virtualisés

Une fois l'ensemble des systèmes virtualisés installés et leurs paramètres réseau ayant été configuré, il suffira ensuite de procéder aux étapes suivantes.

### 2.1. Gestion de clés pour les connexions SSH (Ansible)

Copier la clé publique, selon chaque système virtualisé, associée à l'utilisateur *ansible* sur chaque machine virtuelle, dans le fichier *authorized\_keys* du répertoire SSH de l'utilisateur *ansible*.

Cela permettra à l'utilisateur opérant depuis le poste de contrôle Ansible d'exécuter des instructions sur chaque système virtualisé sans avoir besoin de renseigner de mots de passe lors des exécutions.

### 2.2. Droits sudo de l'utilisateur ansible

Faire en sorte que l'utilisateur *ansible* sur chaque système virtualisé puisse exécuter des commandes privilégiées sans mot de passe, ou en fournissant un mot de passe de manière sécurisée, pour l'exécution d'instructions privilégiées par Ansible.

Par souci de simplicité, on pourra se résoudre alors (à ne jamais utiliser dans un environnement de production) à accorder à l'utilisateur *ansible* des droits sudo (élévation de privilèges) sans mot de passe nécessaire, et ce pour l'exécution de toute commande :

1. Avec un utilisateur disposant de droits suffisants pour le faire (l'utilisateur *root* par exemple), éditer le fichier de droits sudo du système :  
**# visudo**
2. Ajouter les droits d'élévation de privilèges lors de l'exécution de toute commande, sans demande de mot de passe, pour l'utilisateur *ansible* :  
**ansible ALL=(ALL) NOPASSWD:ALL**
3. Enregistrer les modifications.

## 3. Ajustement de l'inventaire utilisé par Ansible

Selon les paramètres réseau attribués à chacun des systèmes virtualisés, il faudra adapter le contenu du fichier *hosts* fourni dans le fichier ZIP. Ce fichier correspond à un fichier de définition d'hôtes utilisé par Ansible, appelé inventaire, définissant les paramètres avec lesquels Ansible va tenter de se connecter aux différents systèmes sur lesquels exécuter des jeux d'instructions.

Voir une [section introductive](#) de la documentation officielle d'Ansible portant sur les inventaires.



Il est vivement recommandé d'utiliser des paramètres réseau identiques à ceux fournis dans cette documentation. Autrement, d'autres adaptations devront être réalisées dans le reste des fichiers fournis, spécifiquement dans ceux des **services**.

#### 4. Installation d'Ansible sur le contrôleur Ansible

Etant désormais en possession de l'ensemble des fichiers nécessaires et de quelques systèmes virtuels préconfigurés pour interagir avec un contrôleur Ansible, il reste à installer Ansible sur la machine qui sera utilisée avec l'arborescence de fichiers fournie pour contrôler l'exécution d'instructions sur les systèmes virtuels.

Voir la section de la documentation officielle d'Ansible détaillant les étapes et éléments nécessaires pour [installer Ansible](#).

## Déploiement de la maquette

On peut désormais passer au déploiement de la maquette.

Le déploiement de la maquette consistera essentiellement en l'exécution de jeux d'instructions successifs (playbooks Ansible), situés dans le dossier **ansible**, depuis un terminal en lignes de commandes à l'aide des commandes suivantes :

```
ansible-playbook -i hosts playbook_1.yml
```

```
ansible-playbook -i hosts playbook_2.yml
```

```
ansible-playbook -i hosts playbook_3.yml
```

```
ansible-playbook -i hosts playbook_4.yml
```

```
ansible-playbook -i hosts playbook_5.yml
```

```
ansible-playbook -i hosts playbook_6.yml
```

L'exécution du dernier jeu d'instructions créera une clé d'API Grafana dans le dossier **services**, qui pourra par la suite être exploitée par le script Bash *grafana\_imports.sh* pour importer une source de données Prometheus ainsi qu'un graphe permettant de visualiser les réponses HTTP aux requêtes sur les services web déployés, au travers du service HAProxy.