Installer Docker et Docker Compose

Installer le dépôt Docker

- Méthode installation pour Debian : https://docs.docker.com/engine/install/debian/
- Installer le dépôt Docker :
  - Mettre à jour les dépôts : `apt update`
  - Installer les paquets suivant : `apt install ca-certificates curl gnupg`
  - Ajouter la clé du dépôt Docker :

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg
--dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

  - Ajouter le fichier apt contenant le dépôt Docker :

```
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-
by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Installer Docker

- Installer les paquets nécessaire : `apt install apt-transport-https gnupg2 software-properties-common`
- Installer Docker : `apt install docker-ce docker-ce-cli containerd.io`
- Activer le service Docker au démarrage : `systemctl enable docker`
- Ajouter l'utilisateur *admin* au groupe *docker* : `usermod -aG docker admin`
  - Se connecter sur une console avec l'utilisateur *admin* et tester Docker : `docker run hello-world`
  - Si le message d'erreur `docker: Got permission denied while trying to connect to the Docker daemon socket` apparait :
    - Essayer de sourcer à nouveau le fichier `.bashrc` : `source ~/.bashrc`
    - Essayer de redémarrer la machine pour activer la prise en compte de Docker
  - Tester à nouveau le bon fonctionnement de Docker : `docker run hello-world`

Limiter la mémoire et le CPU des containers Docker

Afin de pouvoir limiter la consommation de mémoire et de CPU des containers Docker, il est nécessaire de vérifier que le noyau Linux est compilé avec les bonnes options :

Si le noyau est correctement compiler, vous pouvez passer à l'étape suivante :

- Vérifier si la configuration de l'hôte est nécessaire : '' docker info ''
    - Si la commande précédente afficher le message suivant il est nécessaire de suivre les étapes suivantes : **WARNING: No swap limit support**
- Éditer le fichier de configuration de Grub et y ajouter/compléter le contenu suivant : '' vi /etc/default/grub ''

```
GRUB_CMDLINE_LINUX="cdgroup_enable=memory swapaccount=1"
```

- Valider la prise en compte de la modification précédente avec la commande : '' update-grub ''
- Redémarrer la machine : `reboot`
- Vérifier que le message **WARNING: No swap limit support** a disparu : '' docker info ''

Déplacer le dossier contenant les données de Docker

- Par défaut, les données de Docker sont stockées dans un unique dossier : `/var/lib/docker/`
- Passer en root : `sudo -i`
- Commencer par arrêter le service : `systemctl stop docker`
    - Vérifier l'arrêt : `systemctl status docker`
    - Vérifier qu'il n'y ait plus de processus Docker : `ps faux | grep -i docker`
- Créer un fichier `/etc/docker/daemon.json` dans lequel vous indiquerez le nouveau dossier : `vi /etc/docker/daemon.json`
    - Contenu du fichier `/etc/docker/daemon.json` :

```
{
  "data-root": "/data/docker"
}
```

- Créer le dossier de destination des données de Docker : `sudo mkdir /data/docker`
- Copier les données dans le nouveau dossier : `rsync -avxP /var/lib/docker/ /data/docker`
    - Vérifier que les tailles des 2 dossiers correspondent : `du -hs /var/lib/docker/; du -hs /data/docker`
- Renommer l'ancien dossier : `mv /var/lib/docker /var/lib/docker.old`
- Redémarrer le service Docker : `systemctl start docker`
    - Vérifier le statut : `systemctl status docker`
- Accéder aux interfaces web des outils utilisant Docker, si Portainer est installé vérifier le paramètre *Root directory* de l'hôte.
- Si tout est ok, supprimer l'ancien dossier : `rm -rf /var/lib/docker.old`

Installer Docker Compose

- Dans les versions récentes de Docker, Compose est directement intégré à la commande `docker`. Si la commande suivante afficher la version de Compose, inutile de suivre les étapes suivantes : `docker compose version`
- Installer *docker-compose* v2 : `apt install docker-compose-plugin`
- Vérifier la version installée : `docker --help |grep compose`
- NOTES : supprimer l'ancienne version v1 télécharger manuellement : `rm -f /usr/local/bin/docker-compose ; rm -f /etc/bash_completion.d/docker-compose`

Activer IPv6 pour Docker

- Éditer le fichier */etc/docker/daemon.json* : `vi /etc/docker/daemon.json`
- Ajouter les paramètres suivant :

```
{
    "ipv6": true,
    "fixed-cidr-v6": "c01d:c01a:cafe::/64",
    "experimental": true,
    "ip6tables": true
}
```

- Redémarrer Docker : `systemctl restart docker`
- Vérifier la prise en compte d'IPv6 dans le réseau par défaut : `docker network inspect bridge|grep EnableIPv6`
- Si vous avez déjà créer votre réseau local Docker (Ex. *nginx-proxy*) :
  - éteindre tous les containers via Portainer par exemple.
  - supprimer votre réseau : `docker network rm nginx-proxy`
  - créer à nouveau le réseau externe (nginx-proxy) avec l'activation d'IPv6
  - Redémarrer toutes les stacks avec : `docker compose down; docker compose up -d`
- Si vous utiliser le container nginxproxy/nginx-proxy activer l'écoute sur les IPv6 en plus d'IPv4:

```
    ports:
      - "0.0.0.0:80:80"
      - "0.0.0.0:443:443"
      - "[::]:80:80"
      - "[::]:443:443"
```

  - Vous pouvez vérifier la bonne écoute sur les ports 80 et 443 de l'IPv6 locale avec : `ss -6ltpn`
- Si vous utiliser nginxproxy/docker-gen, activer le support d'IPv6 dans le fichier *docker-compose.yml* :

```
    environment:
      ENABLE_IPV6: true
```

- Tester la connectivité IPv6 vers votre site web avec : https://ready.chair6.net/

Création d'un réseau local Docker

- Créer un réseau local de type bridge "utilisateur" compatible IPv4 et IPv6 avec : '' docker network create --driver=bridge --ipv6 --subnet=172.18.5.0/24 --ip-range=172.18.5.0/24 --subnet a1f:ea75:ca75::/48 --gateway=172.18.5.1 nginx-proxy ''
  - Ainsi, depuis un container associé à ce réseau, il est possible d'adresser l'hôte avec l'IP de la gateway, soit : 172.18.5.1
  - De limiter les accès sur un outil de l'hôte aux IP des container de ce réseau qui seront

créé parmi (--ip-range) : 172.18.5.0/24

Activer l'API Docker sur l'instance

Voir la documentation d'activation de l'API Docker.

Mise à jour de Docker et Docker Compose

- La mise à jour se fait via le système de paquet de Debian à l'aide d'Apt.
- Suite à une mise à jour de Docker sur l'instance "db", il est nécessaire de mettre à jour le fichier systemd s'il n'a pas encore été surchargé.

Copier les fichiers docker-compose.yml depuis le dépôt Github sinp-<region>-srv

Les fichiers *docker-compose.yml* des différents outils de suivi sont hébergés dans un dépôt Github. Il est nécessaire de copier ces fichiers sur les différentes instances afin de pouvoir les installer.

- Les dépôts :
  - SINP PACA : https://github.com/cbn-alpin/sinp-paca-srv
  - SINP AURA : https://github.com/cbn-alpin/sinp-aura-srv
- Ces depôts contiennent les dossiers de base suivant :
  - *bkp-srv* : les fichiers pour l'instance "bkp-srv"
  - *db-srv* : les fichiers pour l'instance "db-srv"
  - *shared* : les fichiers communs aux différentes instances.
  - *web-srv* : les fichiers pour l'instance "web-srv"
- Chacun de ces dossiers de base contient ensuite une arborescence de dossiers comparable à celle du serveur correspondant.
- Les fichiers *docker-compose.yml* sont hébergés par le dossier de l'utilisateur *admin*.
- Pour chaque serveur, synchroniser ces fichiers avec le serveur correspondant en utilisant `rsync` :
  - Se placer à la racine du dépôt
  - Lancer la commande `rsync` suivante, ici pour *web-srv* et le dossier */web-srv/home/admin/docker* avec l'option `--dry-run` (à supprimer quand tout est ok) :

```
 rsync -av ./web-srv/home/admin/docker/ admin@web-<region>-
sinp:/home/admin/docker/ --dry-run
```

Renommer une stack générer via docker-compose

Lorsqu'on lancer `docker-compose up`, Docker-Compose se charge de créer les volumes nommés. Lors de leur création, il utilise le nom du dossier dans lequel se trouve le fichier `docker-compose.yml`. Ainsi, si l'on change ultérieurement ce nom de dossier, Docker-Compose recréer de

nouveaux volumes nommés au prochaine lancement. Il est donc nécessaire de suivre la procédure suivante pour renommer le dossier d'une stack créé par Docker-Compose :

- Se placer dans le dossier de la stack et arrêter les services : `docker-compose down`
- Renommer le dossier de la stack. Ex. : `mv monitor.silene.eu monitor`
- Lister tous les volumes utilisés par la stack via Portainer via le menu "Volumes" puis utiliser le filtre en tapant le premier mot composant le nom du dossier. Ex. : pour le dossier "monitor.silene.eu", taper "monitor". La stack correspondante est nommée "monitorsileneeu".
- Pour chaque volume listé, il est nécessaire de :
  - créer un nouveau volume avec le nouveau nom :

```
docker volume create \
    --label com.docker.compose.project=<futur-nom-de-la-stack> \
    --label com.docker.compose.version=<version-de-docker-compose>
\
    --label com.docker.compose.volume=<nom-du-volume> \
    --name <futur-nom-de-la-stack>_<nom-du-volume>
```

    - Ex. :

```
docker volume create \
    --label com.docker.compose.project=monitor \
    --label com.docker.compose.version=1.24.1 \
    --label com.docker.compose.volume=influxdb-storage \
    --name monitor_influxdb-storage
```

  - S'assurer via Portainer que le nouveau volume a été créé correctement (nom et labels).
  - Copier les données de l'ancien volume vers le nouveau :

```
docker run --rm -it \
    -v <ancien-nom-de-la-stack>_<nom-du-volume>:/from:ro \
    -v <futur-nom-de-la-stack>_<nom-du-volume>:/to \
    alpine ash -c "cd /from ; cp -av . /to"
```

    - Ex. :

```
docker run --rm -it \
    -v monitorsileneeu_influxdb-storage:/from:ro \
    -v monitor_influxdb-storage:/to \
    alpine ash -c "cd /from ; cp -av . /to"
```

- Se placer dans le nouveau dossier de la stack et lancer les services : `docker-compose up -d`
- Vérifier que tout fonctionne correctement

Déplacer les données (volumes nommés) de Docker d'un serveur à un autre

Il est possible de déplacer les volumes nommés d'un serveur à un autre :

- Mettre à jour Docker sur les 2 instances pour avoir la même version installée.

- Arrêter les services utilisant les volumes sur les serveur d'origine : `docker compose down`
  - Si possible arrêter également Docker : `systemctl stop docker`
- Rechercher les dossiers des volumes nommées concernés dans : `/var/lib/docker/volumes/`
- Une fois repéré, copier chaque dossier sur le serveur de destination. Il est neccessaire de passer par l'utilisateur *admin* car il n'y a pas d'accès direct à root sur le serveur de destination. Ex. avec le volume nommé *matomo-storage* dans la stack *analytics* : `rsync -e 'ssh -p <bkp-srv-ssh-port>' -avxP analytics_matomo-storage/ admin@bkp-paca-sinp:~/dwl/analytics_matomo-storage/`
- Sur le serveur de destination, déplacer le dossier dans l'emplacement (non standard) des données de docker (sur `/data/docker/`) : `mv analytics_matomo-storage /data/docker/volumes/`
- Modifier le propriétaire qui a été défini à *admin* par le propriétaire initial, généralement *root* (vérifier sur le serveur d'origine) : `chown –from=admin root -R /data/docker/volumes/analytics_matomo-storage '' * Modifier le DNS pour faire pointer le nom de domaine sur le nouveau serveur. * Vous pouvez maintenant ré-installer les fichiers Docker Compose sur le nouveau serveur. * Modifier les éventuels fichiers de configuration ou ''.env` pour adapter la configuration au nouveau serveur.
- Relancer la stack : `docker compose up -d`
- Vérifier que tout fonctionne à nouveau.

Commandes utiles

- Accéder aux logs du service Docker : '' journalctl -x -u docker.service ''

Docker

- Lancer une image : `docker run <image>`
- Lister toutes les images : `docker image ls -a`
- Supprimer une image : `docker image rm <image id>`
- Lister tous les containers : `docker container ls --all`
- Supprimer un container : `docker container rm <hash>`
- Construire une image à partir d'un fichier *Dockerfile* présent dans le dossier courant : `docker build -t <nom>:<tag> .`
- Se connecter à un container : `docker exec -it <container name> /bin/bash`
  - Depuis un container (si `ip` est installé - paquet *iproute2*), afficher l'IP de l'hôte : `ip route|awk '/default/ { print $3 }'`
- Informations sur la place occupée par Docker :
  - Résumé de la place occupée :

    ```
    docker system df
    ```

  - Infos détaillées dont la place occupée par chaque volume :

    ```
    docker system df --verbose
    ```

- Nettoyage :
  - **ATTENTION: s'assurer au préalable que tous les containers utiles sont bien actif**

sur l'hôte !

- Nettoyer tout (images, containers et réseaux) inactifs :

```
docker system prune
```

- Nettoyer tout (images, containers et réseaux) et les volumes contenant des données (!!) inactifs :

```
docker system prune --volumes
```

Docker Compose

- **Source** : https://opensharing.fr/docker-compose-commandes-utiles
- ** build **: * Construire ou reconstruire les images des services dont le Dockerfile a été modifié : `docker-compose build [SERVICE...]` * Construire ou reconstruire les images avec les dernières versions des images de base : `docker-compose build --pull [SERVICE...]` * Construire ou reconstruire les images sans utiliser le cache : `docker-compose build --no-cache` * ** config **: * Vérifier la configuration et la syntaxe du fichier Docker Compose : `docker-compose config` * Lister les services définis dans le fichier Docker Compose : `docker-compose config --services` * Lister les volumes définis dans le fichier Docker Compose : `docker-compose config --volumes` * ** down ** : * Arrêter les services, supprimer les containers et les réseaux de l'application : `docker-compose down` * Arrêter les services, supprimer les containers, les réseaux et les volumes : `docker-compose down --volumes` * Arrêter les services, supprimer les containers, les réseaux, les volumes et les images construites et téléchargées (**ATTENTION** : la suppression des volumes entraine la perte de toutes les données associées aux containers) : `docker-compose down --volumes --rmi all`
- ** up **:
  - Reconstruire et redémarrer un service seulement : `docker-compose up -d --no-deps --build <service-name>`
- ** Divers ** :
  - Exécuter une commande à l'intérieur d'un container : `docker-compose exec [options] [-e KEY=VAL...] SERVICE COMMAND [ARGS...]`
  - Lister les images utilisées (téléchargées et construites) par les containers de services : `docker-compose images [SERVICE...]`
  - Afficher la sortie produite par les services spécifiés : `docker-compose logs [options] [SERVICE...]` Ex. `docker-compose logs -f --tail=3`

From:
https://sinp-wiki.cbn-alpin.fr/ - **CBNA SINP**

Permanent link:
https://sinp-wiki.cbn-alpin.fr/serveurs/installation/docker?rev=1714402046

Last update: **2024/04/29 14:47**