

DUPONT
Jade

Rapport de Projet 4ème année

P29 - Conteneurs pour site web

Sommaire :

- > [Introduction](#)
- > [I\) Recherches préalables sur rkt et son fonctionnement](#)
- > [II\) Mise en place manuelle d'un conteneur avec une application simple](#)
- > [Conclusion](#)
- > [Wiki Projet P29](#)

Introduction:

Dans le cadre d'une amélioration constante des services proposés sur le web et d'accélérer les systèmes existants ou de diminuer la consommation en ressources physiques ou énergétiques il s'agira dans notre cas de l'hébergement de site webs. Jusqu'à présent pour l'hébergement web on utilisait des technologies comme les Machines Virtuelles ou des conteneurs (exemple: Docker) pour mettre en place des sites webs.

Cependant ces moyens sont, très consommateur en espace pour les Machines Virtuelles ou en gaspillent dans le cas des conteneurs Docker.

Le but du projet est donc de réaliser un système de création/suppression de sites web utilisateurs au travers d'une interface web en utilisant un système de conteneur comme rkt ou runC (exemples proposés) afin que les sites webs soient isolés les uns des autres mais également du système hébergeur.

Il faudra également une architecture réseau pour les conteneurs, ainsi que les scripts de créations/suppression/mise à jour des conteneurs

On rendra les sites webs accessible d'internet au travers d'une adresse IPv4 unique ou d'un réseau IPv6 pour cela un système de redirection web sera nécessaire. Il sera également possible de mettre en place un utilitaire de réservation de sous-domaines DNS.

*** Choix techniques du projet :**

- type de serveur web : Apache2
- module de scripts pour les pages web : PHP
- Type de base de données : MySQL
- type de conteneur : rkt (rocket)

* Choix de l'architecture :

```
Hardware (PC portable)
|
|->Host OS (Ubuntu 16.04) -> super-super-user (root)
|
|   |->rkt (conteneur 1)
|   |   |
|   |   |-> Application (site web 1) -> super-user1
|   |
|   |->rkt (conteneur 2)
|   |   |
|   |   |-> Application (site web 2) -> super-user2
|   |
|   |->rkt (conteneur 3)
|   |   |
|   |   |-> Application (site web 3) -> super-user3
|   |
|   |->rkt (....)
|   |   |
|   |   |-> Application (....)
|   |
|
```

Bien évidemment, on s'assurera que le super-userN n'a pas accès aux données de l'application du super-userK (N différent de K). En revanche, root a accès à toutes les applications (lecture et écriture), peut en ajouter ou en supprimer.

* Choix du matériel :

L'un des avantages que j'ai eu pour ce projet est la non-nécessité de commander du matériel, j'ai donc pu formater mon PC portable afin d'y installer Ubuntu et travailler directement dessus.

* Planning Théorique du déroulement du projet

1) Recherches préalables sur rkt et son fonctionnement

Dans un premier temps il était nécessaire de comprendre comment fonctionnent les conteneurs en général et comment les utiliser.

J'ai effectué quelques recherches sur runC mais j'ai trouvé plus de contenu concernant rkt. J'ai trouvé ce document [\[Container mechanics in rkt and Linux.pdf\]](#) où se trouvent des explications pour comprendre comment fonctionne rkt au niveau système.

Dans la partie Network isolation [p38] et Network tooling [p39] on remarque que rkt réalise la communication réseau via un plugin [\[GitHub - Container Network Interface\]](#)

Pour l'installation de rkt, n'existant pas de version pour ubuntu il est mentionné sur [\[rkt - Ubuntu\]](#) que l'on peut installer la version debian.

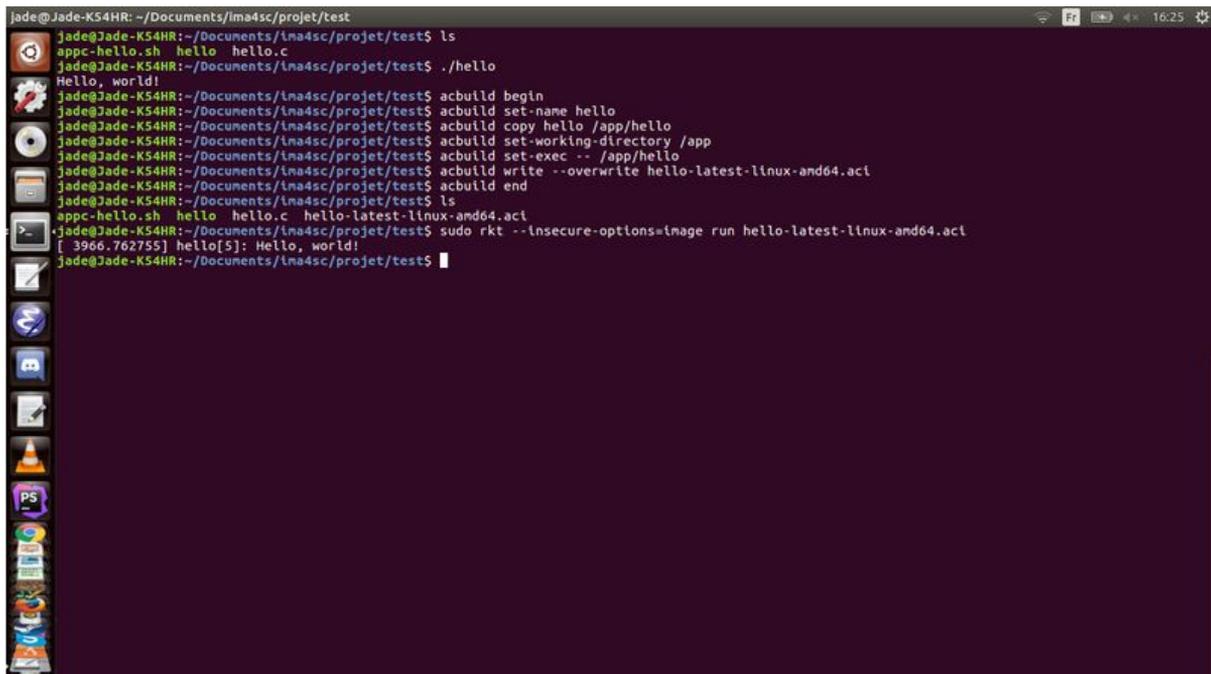
Remarque personnelle: Lors des séances de projet à Polytech, je n'ai pas pu télécharger et/ou installer d'éléments via le wifi en raison de blocage (même avec les paramètres proxy & EXPORT dans le .bashrc)

II) Mise en place manuelle d'un conteneur avec une application simple

Une fois rkt et CNI installés, j'ai consulté un tutoriel sur comment utiliser rkt pour mettre une application dans un conteneur [\[Setup rkt containers\]](#), le lien est mentionné pour fedora mais le principe est le même, on nous indique qu'il faut utiliser un utilitaire: Acbuild [\[GitHug - Acbuild\]](#) afin de créer une image .aci de notre application pour utilisable par rkt pour créer notre conteneur.

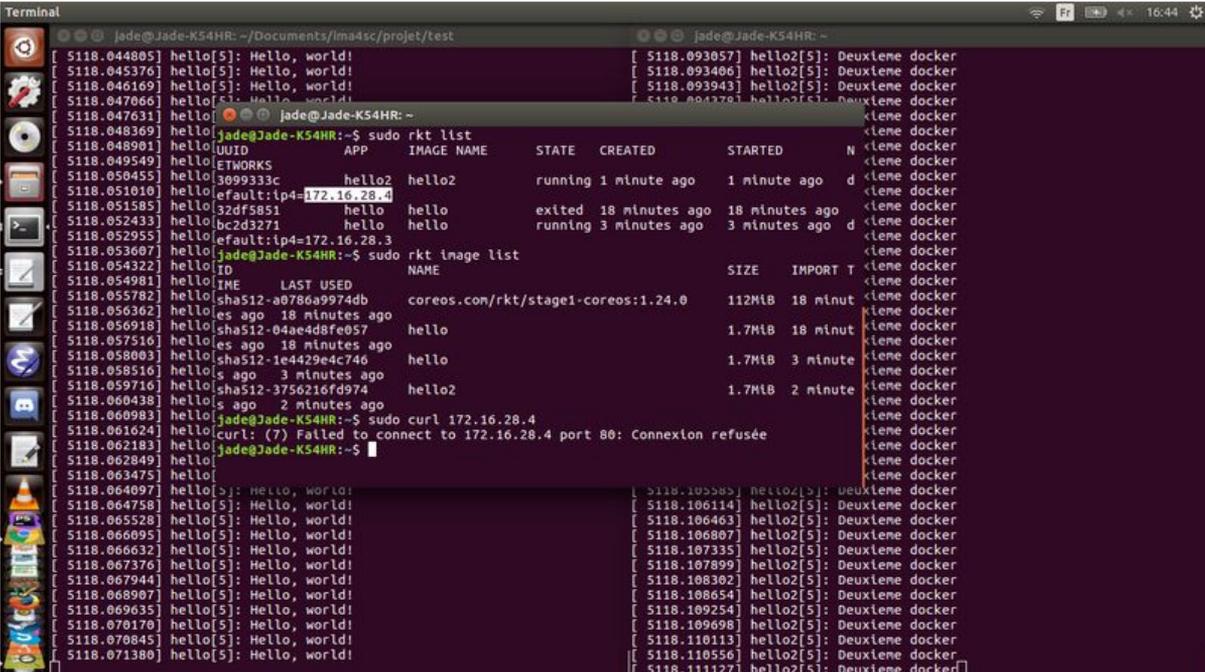
En suivant les instructions d'installation, le système d'exploitation de ma machine étant neuf, il manquait une commande : "Golang-go" permettant d'installer AcBuild et encore une fois le proxy Polytech bloquait l'installation de mon "sudo apt-get install golang-go"

En suivant le tutoriel mentionné précédemment, je suis parvenue à créer manuellement une image .ACI d'un programme C ayant pour unique but de vérifier le bon fonctionnement des utilitaires.



```
jade@Jade-K54HR: ~/Documents/lna4sc/projet/test
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ ls
appc-hello.sh  hello  hello.c
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ ./hello
Hello, world!
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild begin
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild set-name hello
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild copy hello /app/hello
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild set-working-directory /app
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild set-exec -- /app/hello
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild write --overwrite hello-latest-linux-amd64.aci
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ acbuild end
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ ls
appc-hello.sh  hello  hello.c  hello-latest-linux-amd64.aci
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$ sudo rkt --insecure-options=image run hello-latest-linux-amd64.aci
[ 3966.762755] hello[5]: Hello, world!
jade@Jade-K54HR:~/Documents/lna4sc/projet/test$
```

on remarque que le conteneur test s'est correctement lancé puis en modifiant légèrement le code du programme C (ajout d'une boucle) afin d'obtenir une exécution continue dans le conteneur et de maintenir ce dernier



The terminal screenshot shows a user named 'jade' at 'jade@Jade-K54HR' working with Docker containers. The user runs 'sudo rkt list' which shows a table of containers. One container named 'hello2' is in a 'running' state with IP address '172.16.28.4'. The user then runs 'sudo rkt image list' showing a list of images. Finally, the user runs 'sudo curl 172.16.28.4' which fails with the message 'Failed to connect to 172.16.28.4 port 80: Connexion refusée'. The terminal also shows a continuous stream of 'hello[5]: Hello, world!' messages from the 'hello2' container.

APP	IMAGE NAME	STATE	CREATED	STARTED	N
hello	hello2	running	1 minute ago	1 minute ago	d

NAME	SIZE	IMPORT T
coreos.com/rkt/stage1-coreos:1.24.0	112MiB	18 minut
sha512-a0786a9974db		
es ago 18 minutes ago		
sha512-04ae4d8fe057	1.7MiB	18 minut
es ago 18 minutes ago		
sha512-1e4429e4c746	1.7MiB	3 minute
s ago 3 minutes ago		
sha512-3756216fd974	1.7MiB	2 minute
s ago 2 minutes ago		

Maintenant que le programme s'effectue en boucle (deux versions différentes avec chacune un conteneur respectif) on remarque dans la liste des applications en cours d'exécution que chaque conteneur encore actif possède une adresse IPv4 d'accès (pouvant normalement être utilisé via la commande "curl") mais faute de configuration revokit un refus en cas de tentative de connexion.

Pour ce qui est de la suite, la mise en place d'un serveur apache2 dans un conteneur, je n'ai pas pu poursuivre en raison de manque de temps personnel et/ou de connaissances en système et réseau en raison de multiples absences maladie ou autres problèmes personnels rencontrés tout au long de l'année.

Conclusion :

Au cours de ce projet j'ai pu voir la difficulté à mener un projet de niveau ingénieur, complet et en solitaire.

Bien que je n'ai pu terminer la partie principale du projet (création automatisée de conteneur, mise en réseau et accès par internet), je suis convaincue que cela aurait été l'occasion de mettre en place les connaissances (théoriquement) acquises cette année en Système et Réseau pour la partie isolation, mise en place des scripts et la mise en réseau ou alors en Base De Données ainsi que les connaissances en développement web que j'ai acquises au cours de mon stage 2016 pour la partie site web (HTML, PHP, SQL).