TP RT0704

Mise en place de l'environnement de développement

génération d'une clé RSA depuis l'hôte

- ssh-keygen -t rsa

transfert de la clé publique de l'hôte sur l'invité

- ssh-copy-id -i ~/.ssh/id_rsa.pub user@172.18.10.20

vérif de python3, pip3 et virtualenv

- sudo apt-get install python3 -y && sudo apt-get install python3-pip -y && sudo pip install virtualenv

création du script hello-world

- vi hello-world.py

```
#!/bin/python3
print ("hello-world")
```

exécution du script

- python3 hello-world.py

Installation de docker

Installez Docker sur la machine Virtuelle

- sudo apt install docker.io -y

ajout de user au groupe docker (pas besoin de root pour docker)

- sudo usermod -aG docker user

Testez l'installation de docker avec le conteneur **hello world**

- docker run hello-world

Gestionnaire de file

Téléchargez et exécutez un conteneur RabbitMQ

Activation du pluggin management (http) et forward du port 15672 du conteneur rabbitmq au port 8080 de l'ubuntu server (VM)

utilisateur par défaut:

- id=guest

- mdp=guest

commande:

- docker run -d --hostname my-rabbit --name some-rabbit -p 8080:15672 rabbitmq:3-management

Flask

Téléchargez et exécutez un conteneur Flask

Lancer un conteneur flash avec une redirection du port 8000 de l'invité sur le port 8081 de l'hote, le fichier app.py génère automatiquement le hello-world

- docker run -d --privileged --hostname conteneur-flask --name conteneur-flask -p 8081:8000 altoning/flask3

accès OK via http://172.18.10.X:8081

Modification du fichier app

Création d'un conteneur flask qui exécutera un shell /bin/bash

- docker run -a stdin -a stdout -a stderr -it --privileged --hostname conteneur-flask --name conteneur-flask -p 8081:8000 altoning/flask3 /bin/bash

- vi app.py

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
        return "Hello World!"

@app.route("/guerrier")
def guerrier():
        return "Je suis un guerrier"

@app.route("/mage")
def mage():
        return "Je suis un mage"
```

Lancement du script boot.sh

- /bin/bash boot.sh

Créez une page web avec un template JINJA

Jinja 2 est déja installé

- mkdir templates
- vi templates/template1.html

```
bash-4.4# cat templates/template1.html
<html>
        <head>
                <title> Un essai</title>
        </head>
        <body>
                {% if data %}
                <h1> Hello {{ data }} </h1>
                {% else %}
                <h1>Il manque un parametre</h1>
                {% endif%}
        </body>
</html>
```

- vi app.py

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route("/render",methods=['GET'])
@app.route("/render/<name>",methods=['GET'])
def fct9(name=None):
        return render_template("template1.html",data=name)

if __name__ == "__main__":
        app.run(host='0.0.0.0', port=8000,debug=True)
```

- python3 app.py

Réalisez la création d'un couple formulaire / page de traitement exploitant JINJA

- vi templates/formulaire.html

```html
<html>
    <head>
        <title>Plateforme execution de code</title>
    </head>

<body>

    <center>
        <h2>Plateforme d'exécution de code</h2>
    </center>
        <br>
        <form action="/traitement" method="post">
            <label for="type_tache">Choix du service:</label>
            <div>
                <input type="radio" id="Graphviz-" name="type_tache" value="Graphviz">
                <label for="Graphviz">Graphviz</label>
            </div>

            <div>
                <input type="radio" id="Pandoc" name="type_tache" value="Pandoc">
                <label for="Pandoc">Pandoc</label>
            </div>
            <div>
                <input type="radio" id="ImageMagic" name="type_tache" value="ImageMagic">
                <label for="ImageMagic">ImageMagic</label>
            </div>
            <br>
            <br>
            <input type="file" id="fichier" name="fichier">
            <br>

            <label for="cmd">Commande (255 caractères max):</label>
            <br>
            <input type="text" id="cmd" name="cmd" required minlength="4" maxlength="255" size="70">

            <br>
            <div class="button">
                <button type="submit">Start</button>
            </div>
        </form>
    </body>
</html>
```

- vi templates/traitement.html

```html
<html>
    <head>
        <title>Plateforme d'exécution de code</title>
    </head>
    <body>
        <center>
            <h2>Plateforme d'exécution de code</h2>
        </center>
        <p> Le service demandé est  : <br> {{ data1 }} </p>
        <p> le fichier reçu est: <br> {{data2}} </p>
        <p> la commande qui sera exécutée est : <br> {{data3}} </p>
    </body>
</html>
```

- vi app.py

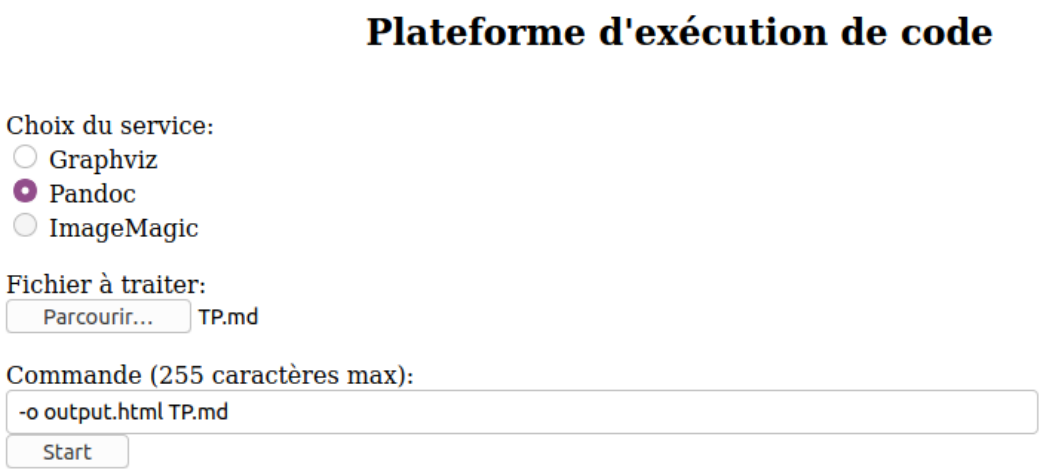
```python
from flask import *
app = Flask(__name__)

@app.route("/formulaire")
def formulaire():
        return render_template("formulaire.html")

@app.route("/traitement",methods=['POST'])
def traitement():
        mydata = {}
        service = request.form["type_tache"]
        fichier = request.form["fichier"]
        cmd = request.form["cmd"]
        return render_template("traitement.html",data1=service,data2=fichier,data3=cmd)

if __name__ == "__main__":
        app.run(host='0.0.0.0', port=8000,debug=True)
```

Page du formulaire:



Page du traitement: