Cursus "Cloud IFB pour les Sciences du Vivant"

Module IBI - 3
Développement de machines virtuelles modèles

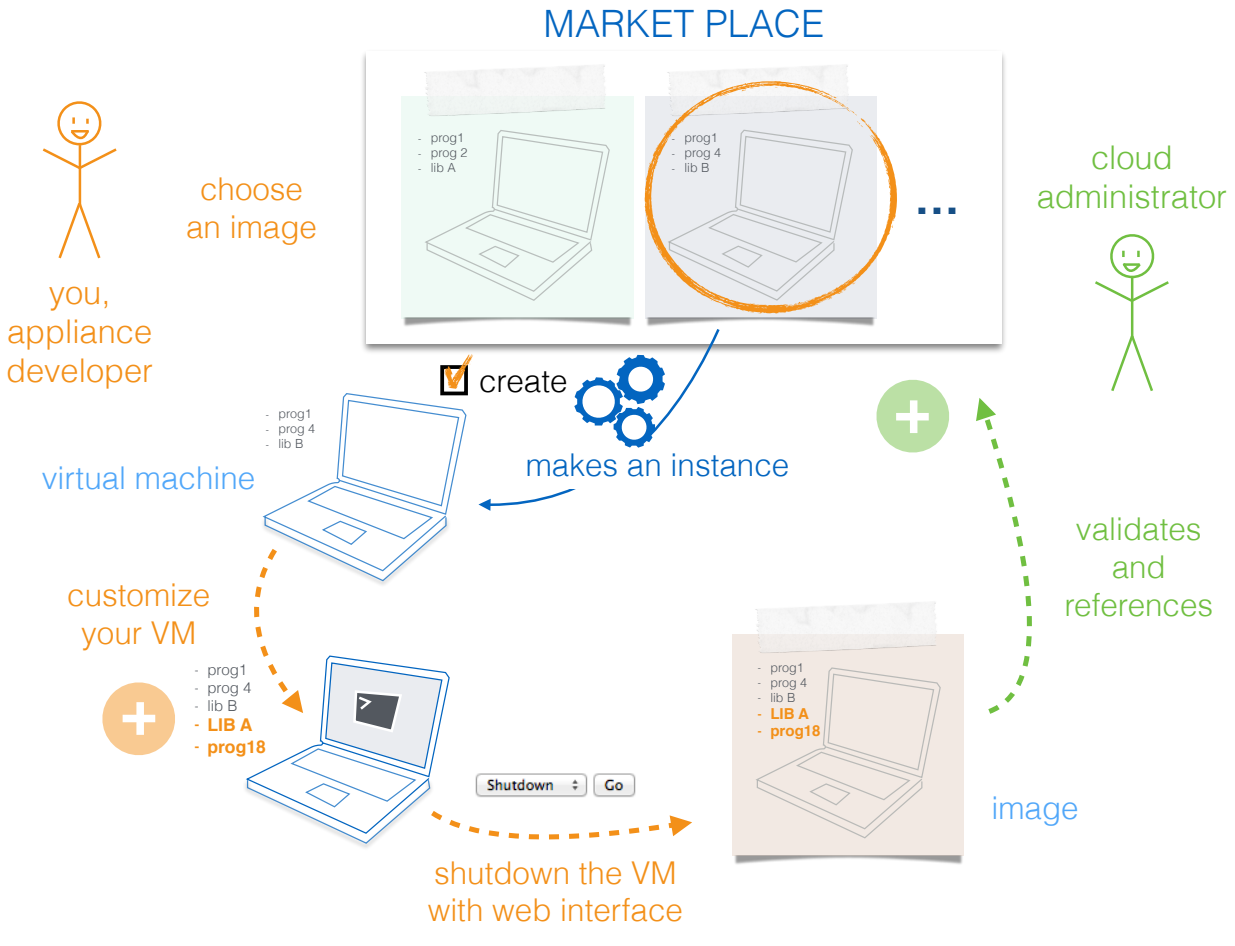
contact : support@france-bioinformatique.fr

I) Le développement d'appliances

a. Principes généraux

Les principales étapes du développement d'appliance sont les suivantes :
- Choisir une image de base (parmi les appliances déjà disponible sur le cloud)
- En faire une instance en cochant le mode create.
- Personnaliser votre appliance en y ajoutant les dépendances et les programmes souhaités.
- Eteindre votre appliance en utilisant l'interface web.

Ci-dessous, une illustration du cycle de création d'une appliance. Il est nécessaire
1. qu'un administrateur cloud valide votre appliance pour l'enregistrer et la rendre visible dans le tableau de bord du cloud
2. et que vous validiez l'appliance en vérifiant le bon fonctionnement des applications installées.



Voici le formulaire qui vous permettra de lancer une instance de votre image de base en mode create :
Une fois votre instance en mode création, elle apparait en orange dans le tableau de bord du cloud :

b. Bonnes pratiques

Pour faciliter le développement de votre appliance, nous avons défini les bonnes pratiques suivantes :

- Rédiger une description de l'appliance
- Faire la liste détaillée de l'ensemble des logiciels nécessaires avec les dépendances et les versions des logiciels, ainsi que les annotations avec l'ontologie EDAM
- Identifier l'appliance qui pourrait servir de base et les outils déjà intégrés (approver/ docker)
- Préparer le processus d'installation de vos outils sur une instance normale
- Installer les logiciels dans les répertoires standard /ifb (ou /usr/local)
- Utilisez l'interface web pour éteindre et enregistrer votre appliance avec le bouton 'shutdown', pas la commande du terminal/shell. Si vous ne souhaitez pas enregistrer l'appliance, utilisez le bouton 'kill'.
- Faire une mise a jour du système avant d'installer vos outils: yum update/apt-get update + dist-upgrade, et reboot s'il y a eu une mise à jour du kernel
- Configurer l'environnement pour un utilisateur standard et conserver le mode ssh avec clé publique
- Utiliser le mode proxy pour le serveur Web
- N'ouvrez que les ports réseau nécessaires et utilisez les ports standard (22, 80/443…)
- Une instance en mode création accumule des logs, éviter de conserver trop longtemps vos instance en mode création.
- Ne changez pas les droits de fichiers par 777…

c. Formulaire de demande de création d'appliance

Demande de création d'une appliance sur le cloud IFB
(à envoyer à l'adresse du support de l'IFB)

- Titre :
- Description :
- Domaines thématiques :
- Logiciels bioinformatiques utilisés :
- Pré-requis (données de références, portail web, bureau virtuel, SGBD, etc.) :
- Contact (unique, Nom et email) :
- Développeur(s) (Nom et email) :
- Affiliation (détaillée) :

II) Installation des logiciels

Il existe différentes méthodes pour installer des logiciels sur une appliance:
- installation interactive : à partir d'une archive des binaires ou des codes sources
- installation automatique : avec un gestionnaire de paquets (yum, apt, rpm…), des scripts (approver, Galaxy toolshed) our des recettes (puppet/ansible/chef/salt)
- avec des conteneurs docker

Lors de cette session pratique, nous allons installer un logiciel suivant ces trois façons différentes:

- en ligne de commande ;
- en créant un script approver ;
- en créant une recette conda ;
- en créant un Dockerfile pour créer une image Docker.

1) Installation interactive

Ici, on souhaite installer la version 2.0.14 de TopHat.

```
VERSION=2.0.14

wget ccb.jhu.edu/software/tophat/downloads/tophat-$
{VERSION}.Linux_x86_64.tar.gz
# or upload of the archive from your local computer if needed with a scp
command

tar -xzf tophat-${VERSION}.Linux_x86_64.tar.gz
cd tophat-${VERSION}.Linux_x86_64

# you can remove files that you don't need for instance
rm -f AUTHORS COPYING README

# then put the bin in the IFB directory
mv * /ifb/bin/

# clean the remains
cd ..
rm -r -f tophat-${VERSION}.Linux_x86_64*
```

2) Script Approver

Afin d'automatiser cette installation sous la forme d'un **script approver** utilisez des variables pour gérer les noms de programmes et paquets, leur numéro de version et leur URL. De plus il n'est pas nécessaire de passer par la phase de nettoyage.

```
tool_id="tophat"
tool_bin="tophat"
tool_version="2.0.14"
tool_url="ccb.jhu.edu/software/tophat/downloads"
tool_pkg="${tool_id}-${tool_version}.Linux_x86_64.tar.gz"

# install the tool
wget "${tool_url}/${tool_pkg}"
tar -xzf ${tool_pkg}

rm -f -r "${tool_pkg%.tar.gz}/AUTHORS ${tool_pkg%.tar.gz}/COPYING
${tool_pkg%.tar.gz}/README"
mv "${tool_pkg%.tar.gz}"/* ${tools_dir}/bin/
```

Pour exécuter l'installation de TopHat 2.0.14, il est ensuite nécessaire d'enregistrer ce script sous le nom nom-version.sh (tophat-2.0.14.sh) puis de le mettre en ligne (contacter le support de l'IFB). Enfin, la commande à lancer depuis votre instance est :

```
approver -i /ifb -t tophat-2.0.14
```

Test l'installation

```
mkdir tophat_with_approver && cd $_
wget https://ccb.jhu.edu/software/tophat/downloads/test_data.tar.gz && \
tar xzf test_data.tar.gz && \
rm test_data.tar.gz && \
cd test_data && \
tophat -p 20 test_ref reads_1.fq
```

3) Installation avec conda

Une recette d'installation conda comprend 2 fichiers :
- un fichier meta.yaml : ensemble des métadonnées de description de l'outil et des dépendances utiles;
- un fichier build.sh : script d'installation de l'outil.

Ici, dessous la recette d'installation de Tophat 2.1.0.
Exercice : identifier les différences dans la recette et à exécution avec la version disponible sur le dépôt Github (https://github.com/bioconda/bioconda-recipes).

Installation anaconda2 dans un machine virtuelle CentOS ou Ubuntu.

```
wget http://repo.continuum.io/archive/Anaconda2-4.1.0-Linux-x86_64.sh
chmod 755 Anaconda2-4.1.0-Linux-x86_64.sh
bash Anaconda2-4.1.0-Linux-x86_64.sh
source ~/.bashrc
conda info
conda config --add channels bioconda
```

Fichier meta.yaml:
https://github.com/bioconda/bioconda-recipes/blob/master/recipes/tophat2/meta.yaml

```
build:
  number: 0

about:
    home: http://ccb.jhu.edu/software/tophat/index.shtml
    license: Boost Software License
    summary: A spliced read mapper for RNA-Seq

package:
    name: tophat
    version: 2.1.1

requirements:
    build:
        - python

    run:
        - python
        - bowtie2
        - samtools

test:
    commands:
        - (tophat --version 2>&1) > /dev/null

source:
  fn: tophat-2.1.1.Linux_x86_64.tar.gz
  url: http://ccb.jhu.edu/software/tophat/downloads/
tophat-2.1.1.Linux_x86_64.tar.gz
md5: 97fe58465a01cb0a860328fdb1993660
```

Fichier : build.sh
https://github.com/bioconda/bioconda-recipes/blob/master/recipes/tophat2/build.sh

```
#!/bin/bash

mkdir -p $PREFIX/bin
binaries="\
bam2fastx \
bam_merge \
bed_to_juncs \
contig_to_chr_coords \
fix_map_ordering \
gtf_juncs \
gtf_to_fasta \
juncs_db
long_spanning_reads \
map2gtf \
prep_reads \
sam_juncs \
samtools_0.1.18 \
segment_juncs \
sra_to_solid \
tophat \
tophat2 \
tophat-fusion-post \
tophat_reports \
"
directories="sortedcontainers intervaltree"
pythonfiles="tophat bed_to_juncs contig_to_chr_coords sra_to_solid
tophat-fusion-post"
PY3_BUILD="${PY_VER%.*}"
if [ $PY3_BUILD -eq 3 ]
then
    for i in $pythonfiles; do 2to3 --write $i; done
fi
for i in $binaries; do cp $i $PREFIX/bin && chmod +x $PREFIX/bin/$i;
done
for d in $directories; do cp -r $d $PREFIX/bin; done
```

Création de la recette, se placer dans le dossier au-dessous de celui de la recette :

```
conda build <dirname>
ls -l <path>/anaconda2/conda-bld/linux-64/<tool_name>-<version>-
py35_0.tar.bz2
```

Installation de l'outil

```
conda build <tool_name>
```

Validation de l'installation avec le jeu test de tophat

```
mkdir tophat_with_conda && cd $_ && \
wget https://ccb.jhu.edu/software/tophat/downloads/test_data.tar.gz && \
tar xzf test_data.tar.gz && \
rm test_data.tar.gz && \
cd test_data && \
tophat -p 20 test_ref reads_1.fq
```

4) Image Docker

Ici, on souhaite installer la version 2.1.0 de Tophat.

Afin d'obtenir un conteneur Docker qui inclue ce logiciel, il est possible de s'inspirer d'un script approver pour générer un Dockerfile comme suit.

```
###################################################################
# Dockerfile
#
# Version:          1.0
# Date:             19/01/2016
# Software:         TopHat
# Software Version: 2.1.0
# Description:      A spliced read mapper for RNA-Seq
# Website:          http://ccb.jhu.edu/software/tophat/index.shtml
# Tags:
# Provides:
# Base Image:       ifb/tophat:2.1.0
# Build Cmd:        docker build --rm -t ifb/tophat:2.1.0
# Pull Cmd:         docker pull ??
# Test Cmd:         tophat -r 20 /data/test_ref /data/reads_1.fq /data/
reads_2.fq
###################################################################


# Set the base image to Debian
FROM debian:wheezy

# File Author / Maintainer
MAINTAINER Sandrine Perrin <support@france-bioinformatique.fr>

# Set environment variable
ENV TOOL_ID tophat
ENV TOOL_NAME TopHat
ENV TOOL_BIN tophat
ENV TOOL_VERSION 2.1.0
ENV TOOL_ARCHI Linux_x86_64
ENV TOOL_PKG ${TOOL_ID}-${TOOL_VERSION}.${TOOL_ARCHI}
ENV TOOL_ZIP ${TOOL_PKG}.tar.gz
ENV TOOL_URL ccb.jhu.edu/software/tophat/downloads/${TOOL_ZIP}

ENV DST /usr/local/bin
ENV PACKAGES \
        bzip2 \
          cpanminus \
          gcc \
          libncurses5-dev \
          make \
      python \
      unzip \
      wget \
      zlib1g-dev
```

```
# Update the repository sources list and install dependencies
RUN apt-get update && \
    apt-get install --yes ${PACKAGES} && \
    perl -MCPAN -e FindBin && \
    apt-get clean all

WORKDIR ${DST}

# Install tool
RUN wget $TOOL_URL && \
    tar xzf ${TOOL_ZIP} && \
    cd ${TOOL_PKG}

# Install dependance on samtools
RUN wget --no-check-certificate https://github.com/samtools/samtools/
releases/download/1.2/samtools-1.2.tar.bz2 && \
    tar xjf samtools-1.2.tar.bz2 && \
    cd samtools-1.2 && \
    make all && \
    make install && \
    make clean && \
    rm -r ${DST}/${TOOL_ID}* && \
    rm -r ${DST}/samtools*

# Install dependance on bowtie 2.0.X
RUN wget downloads.sourceforge.net/project/bowtie-bio/bowtie2/2.2.6/
bowtie2-2.2.6-linux-x86_64.zip && \
    unzip bowtie2-2.2.6-linux-x86_64.zip && \
    cd bowtie2-2.2.6 && \
    rm -r ${DST}/bowtie2-2.2.6*

WORKDIR /
```

Pour construire une image et lancer un conteneur, enregistrez le fichier sous le nom Dockerfile, placez-le dans un répertoire contenant l'archive de l'application.

Placez-vous dans ce répertoire et lancez la commande suivante pour construire votre image :

```
docker build -t ifb/tophat:2.1.0 .
```

Puis lancer le conteneur :

```
docker run -t -i --rm ifb/tophat:2.1.0 /bin/bash
```

Ou lancer une analyse:

```
docker run -v <path/data>:/data -w /data ifb/tophat:2.1.0 tophat <arg>

mkdir tophat_with_docker && cd $_ && \
wget https://ccb.jhu.edu/software/tophat/downloads/test_data.tar.gz && \
tar xzf test_data.tar.gz && \
rm test_data.tar.gz && \
cd test_data && \
docker run -v $(pwd):/data -w /data ifb/tophat:2.1.0 tophat -p 20
test_ref reads_1.fq
```

Pour publier vos images docker et les mettre à la disposition de la communauté, vous avez plusieurs solutions:

- le dépôt **BioShaDock (**http://docker-ui.genouest.org/ **)**, dédié aux outils bioinformatiques, utilisant l'ontologie EDAM [1] pour la description et permettant de créer un lien vers la fiche de l'outil dans le catalogue bio.tools (http://bio.tools/). Pour pouvoir contribuer au dépôt, il suffit de se créer un compte sur GenOuest (https://my.genouest.org/manager/index.html#/login);

- le dépôt **BioConda** (https://github.com/bioconda/bioconda-recipes), les recettes sont convertis en Dockerfile automatiquement pour alimenter le dépôt BioShaDock. Pour contribuer au dépôt publique, il suffit de faire une pull-request depuis voir dépôt github vers celui de BioConda (https://help.github.com/articles/creating-a-pull-request/);

- le toolshed de **Galaxy** ou le toolshed de votre organisme s'il existe (ex: toolshed de Genouest), ou le toolshed de l'IFB (http://www.france-bioinformatique.fr/fr/groupes-de-travail/galaxy).

Bonnes pratiques :

- vérifier que la version de l'outil n'est pas déjà disponible, si on souhaite mettre à disposition une version modifiée à celle existante, renseigné la partie description;
- toujours tapés les outils déposées avec la même syntaxe;
- bien renseignés les parties de métadonnées;
- commenter les fichiers;
- pour les images Docker, publier les dockerfile dans le dépôt Docker ou Github;
- indiquer qui est l'auteur ;

Exercice :

Comparer la fiche de Tophat disponible sur BioShaDock:

- la version de ifb, l'image est créée à partir d'un Dockerfile;
- la version de bioconda, l'image est créée par un script automatique qui construit un fichier Dockerfile pour construire l'image.

[1] visualisation de l'ontologie EDAM : http://rainbio.france-bioinformatique.fr/rainbio/browseEdam
présentation d'EDAM : http://edamontology.org/page