Cloud Architecure

Jan Kirenz

Table of contents

Preface **4**

1 Create SSH-Key **5**
1.1 Windows . . . 5
1.2 MacOS . . . 5
1.2.1 Create key . . . 5
1.2.2 Set permissions . . . 6
1.2.3 Mac Finder . . . 6
1.3 Import public key . . . 7

2 Create a launch instance **8**
2.1 Details . . . 8
2.2 Source . . . 8
2.3 Flavor . . . 8
2.4 Networks . . . 8
2.5 Network Ports . . . 9
2.6 Security Groups . . . 9
2.7 Key Pair . . . 9
2.8 Launch Instance . . . 9
2.9 Increae Storage . . . 9
2.9.1 Install text-based web browser . . . 12

3 Log into instance **13**
3.1 Terminal . . . 13
3.1.1 Mac . . . 13

4 Docker **14**
4.1 Set up Docker repository . . . 14
4.2 Install Docker Engine . . . 15

5 Increase Cloud Storage **16**
5.1 Create new volume in bwCloud . . . 16
5.2 Attach volume in bwCloud . . . 16
5.3 Mount volume in VM . . . 17
5.3.1 Log in your VM . . . 17
5.3.2 Find volumne . . . 17

5.3.3 Partitioning 17
5.3.4 Format partition 18
5.3.5 Mounting 18

References **22**

Preface

This online book covers the following topics:

1. Create a SSH-key to be able to access the bwCloud server and handle authentication: create-key-pairs
2. Create an instance (i.e. a virtual machine hosted on bwCloud): create-instance
3. Set up Docker: setup-docker
4. Increase storage and mount to Docker: increase-storage
5. Set up TensorFlow Extended (TFX) with Docker: tfx-setup

1 Create SSH-Key

You can use a SSH-key to access remote servers and to handle authentication for command line programs.

1.1 Windows

Follow the instructions provided in bwCloud: SSH-Key Paar erzeugen

1.2 MacOS

1.2.1 Create key

Open a terminal and run the following command:

```
ssh-keygen
```

This will output:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/username/.ssh/id_rsa):
```

Press enter to save your keys to the default `/Users/username/.ssh` directory.

After entering and confirming your password, you'll see something like the following:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/username/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/username/.ssh/id_rsa
Your public key has been saved in /Users/username/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:BOJAxs0Rkhusd9Hq/xdqWDnfd1cdxN5Uk+hD2gNwNLA1HvUM username@somename.local
```

```
The key's randomart image is:
+---[RSA 3072]----+
| .o*O ..D.BB+...o|
|  .*. .  +o+=E...|
|   ..        ..oo|
|    . ..    oo o=|
|   . . .S     o +|
|           . .  .|
|   ..o. . D.     |
|   .+. .   XC    |
|  .o.o.   L.     |
+----[SHA256]-----+
```

1.2.2 Set permissions

Next, we use chmod to change permissions (otherwise, bwCloud will refuse the connection).

> **Note**
>
> In Unix and Unix-like operating systems, `chmod` is the command and system call used to change the access permissions of files and directories. The name chmod was chosen as an abbreviation of **ch**ange **mod**e.

Chmod 700 sets folder permissions so that only the owner can read, write and execute files in this folder:

```
chmod 700 .ssh
```

Permissions of 600 mean that the owner has full read and write access to the file, while no other user can access the file:

```
chmod 600 .ssh/id_rsa
```

1.2.3 Mac Finder

1. Open Finder and navigate to `your/Users/username/`

How to show your home folder in Finder

If you don't find your home: In Finder, click on the menue and choose: `Finder > Preference > Sidebar > Show these items in the sidebar` and checkmark the box beside the home icon.

1. Now press the `Command + Shift + .` (period) keys at the same time. The hidden files will show up as translucent in your folder.
2. Open the folder `.ssh`.
3. You should see your public SSH key (`id_rsa.pub`) and private SSH key (`id_rsa`)

1.3 Import public key

1. Open your bwCloud Dashboard
2. Navigate to *Key Pairs* in the left side menue.
3. Click on *Import Public Key*
4. Provide a *Key Name* and choose *Key Type* SSH Key.
5. Now open your public key `id_rsa.pub` in a code editor like VS Code and copy and paste the content into *Public Key*

2 Create a launch instance

1. Open your bwCloud Dashboard
2. Navigate to *Instances* in the left side menue.
3. Select *Launch Instance*

Now follow the steps outlined below.

2.1 Details

- Provide the instance name. We choose: `ml-pipeline-01`
- Provide a description for the instance, e.g. `Example ML Pipeline`
- Choose count `1`

2.2 Source

- Instance source is the template used to create an instance.
- Choose `Ubuntu 22.04`

2.3 Flavor

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

- Choose `m1.large` (4 VCPUS, 8 GB Ram, 12 GB storage)

2.4 Networks

Networks provide the communication channels for instances in the cloud.

- We use public-belwue

2.5 Network Ports

Ports provide extra communication channels to your instances. You can select ports instead of networks or a mix of both.

- We dont use network ports

2.6 Security Groups

- Choose default and your custom security groups
- Follow this bwCloud-tutorial to open a port.

2.7 Key Pair

- Select the key pair from step create-key-pairs. Here, it's called id-rsa-pub

2.8 Launch Instance

We are done and you can click on Launch Instance.

You should see your newly created instance in your dashboard.

Click on the instance name to see more details

2.9 Increae Storage

1. Log in to bwCloud Dashboard
2. Click on Volumes below Project → Volumes
3. An overview of the volumes you have created so far is displayed. To create a new volume, click on Create Volume
4. A dialogue opens. Fill in the fields according to your requirements: block-device-01, We use 25 GiB.
5. Then click on Create Volume. The volume is created

In order for a volume to be used by a virtual machine, it must be added ("attached") to a VM.

1. In the table row of the desired volume, select the subitem Manage Attachments in the context menu at the right end of the row and click on the entry.

A dialogue opens In the dialogue, select the desired virtual machine (ml-pipeline-01) and click on Attach Volume

- The table updates and the path under which the new volume can be reached from within the virtual machine appears in the Attached To field

`/dev/vdb on ml-pipeline-01`

- Log into the virtual machine

Check the external device:

```
sudo fdisk -l
```

This will output

```
...
Disk /dev/vdb: 25 GiB, 26843545600 bytes, 52428800 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

so for that you have to partition it first:

```
sudo parted /dev/vdb
```

After executing the above command, within the environment of the command, you should follow the command mentioned below to allow greater than 2GB's partitions.

```
mklabel GPT
```

Within this partition environment, you can also set the size, it depends on the user's choice to allocate the space, as in our case we are limiting the partition from GB to 3GB by using the following command: you just have to change the upper bound to have a partition of greater or lesser size.

```
mkpart primary 0GB 25GB
```

Once you have successfully attempted the above steps of partition; now you must assign a file system for the newly made partition. However, you can proceed only if you leave the parted

environment; just type “quit” in the terminal and hit enter the terminal will come out of that environment:

```
quit
```

The final step to active the partition is assigning file format for the new partition; so, you have to choose the file system (.ext4) of the newly made partition by using the below-mentioned command:

```
sudo mkfs.ext4 /dev/vdb
```

Step 3: Mount Process : Once the partition step is performed, you can mount the drive; before this, you should create a new directory in the “/mnt/” directory where the drives are usually mounted in Ubuntu. Make new directory in “/mnt/” by using the terminal as mentioned below:

```
sudo mkdir /mnt/vdb
```

Once the directory is created, you can mount the inserted drive by using the command written below:

```
sudo mount /dev/vdb /mnt/vdb
```

However, it is observed that the below command will temporarily mount the drive, whenever you will restart the system, you must mount it again. To avoid such happening you must edit the file system table “fstab” by using the nano editor:

```
sudo nano /etc/fstab
```

Now, add the following content at the end of the file:

```
/dev/vdb    /mnt/vdb     ext4      defaults        0             0
```

Ctrl+O will save the modifications you’ve made to the file. Ctrl+X will close nano.

How to check the drive is mounted or not

You can check the mounted drive by using the command mentioned below: you can specify the directory name (sdb in our case) after “grep” to get the pointed information; otherwise, the mount command will list down all the mounted drives and partitions in your system:

```
sudo mount | grep vdb
```

2.9.1 Install text-based web browser

w3m is a text-based web browser as well as a pager like `more' or`less'. With w3m you can browse web pages through a terminal emulator windo

```
sudo apt-get install w3m w3m-img
```

to open a webpage simply type in a terminal window: w3m

For example:

```
w3m https://w3m.sourceforge.net/
```

- to open a new page: type Shift-U
- to go back one page: Shift-B
- open a new tab: Shift-T
- to exit: q

3 Log into instance

Prerequisites:

- key-pairs-create.md
- instance-create.md

3.1 Terminal

3.1.1 Mac

Change the IP adress and enter

```
ssh -i .ssh/id_rsa ubuntu@193.196.52.36
```

If you are asked: Are you sure you want to continue connecting (yes/no/[fingerprint])?

Enter “yes”.

Enter passphrase for key ‘.ssh/id_rsa’:

Provide your password

4 Docker

First, we install Docker Engine on Ubuntu. We simply follow the installation steps outlined in the official Docker documentation

4.1 Set up Docker repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

Update the apt package index:

```
sudo apt-get update
```

Install packages to allow apt to use a repository over HTTPS:

```
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

Confirm the installation with `Y`

Add Docker's official GPG key:

```
sudo mkdir -p /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/k
```

Use the following command to set up the repository:

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

4.2 Install Docker Engine

Again, update the apt package index:

```
sudo apt-get update
```

Install Docker Engine, containerd, and Docker Compose:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Press `Y` to continue

Verify that the Docker Engine installation is successful by running the hello-world image:

```
sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

You have now successfully installed and started Docker Engine.

The docker user group exists but contains no users, which is why you're required to use sudo to run Docker commands.

You may optionally continue to Linux post-install to allow non-privileged users to run Docker commands and for other optional configuration steps.

You can follow these optional post-installation procedures which shows you how to configure your Linux host machine to work better with Docker. In particular, it let's you run the Docker daemon as a non-root user (Rootless mode). In this tutorial, I

Let's take a look at

```
sudo docker info
```

Search for TFX images on Docker Hub:

```
sudo docker search tfx
```

Once you've identified the image that you would like to use, you can download it using the `pull` subcommand.

Execute the following command to download the latest official TFX image (8.81 GB):

```
sudo docker pull tensorflow/tfx:latest
```

5 Increase Cloud Storage

Mount a volume to Docker

In this tutorial, you'll learn how to increase your cloud storage and mount the volume to docker in order to increase the virtual machine's docker image capacity.

5.1 Create new volume in bwCloud

1. Log into bwCloud Dashboard
2. Click on `Volumes` below Project → Volumes
3. An overview of the volumes you have created so far is displayed. To create a new volume, click on Create Volume
4. A dialogue opens. Fill in the fields according to your requirements. We call the volume *block-device-01* and use `25` GiB.
5. Then click on Create Volume. The volume is created

5.2 Attach volume in bwCloud

In order for a volume to be used by a virtual machine, it must be added ("attached") to your VM inside bwCloud.

1. In the table row of the desired volume, select the subitem `Manage Attachments` in the context menu at the right end of the row and click on the entry.
2. In the dialogue, select the desired virtual machine (*ml-pipeline-01*) and click on `Attach Volume`

- The table updates and the path under which the new volume can be reached from within the virtual machine appears in the Attached To field:

```
/dev/vdb on ml-pipeline-01
```

5.3 Mount volume in VM

In Linux, the process of attaching a filesystem to a particular point in the directory tree is called mounting. This allows you to access the files and directories on the filesystem as if they were part of the filesystem on which you are currently working.

5.3.1 Log in your VM

- Log into your virtual machine (replace 111.111.11.11 with your IP)

```
ssh -i .ssh/id_rsa ubuntu@111.111.11.11
```

5.3.2 Find volumne

You can find your volume (i.e. disk) using:

```
sudo fdisk -l
```

The output should include an entry like the following:

```
Disk /dev/vdb: 25 GiB, 26843545600 bytes, 52428800 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

5.3.3 Partitioning

Next, we need to partition our volume. This command will open a partition environment:

```
sudo parted /dev/vdb
```

We use GPT (GUID Partition Table), which is a standard for the layout of the partition table on a physical storage device:

```
mklabel gpt
```

Within the partition environment, you can also set the size of the partition. We set the upper bound to 25 GB.

```
mkpart primary 0GB 25GB
```

Enter `quit` to leave the partition environment:

```
quit
```

5.3.4 Format partition

Format the partition with the *ext4* filesystem. *ext4* stands for "extended file system version 4", which is a popular filesystem used in Linux systems to store and organize files on a storage device, such as a hard drive or solid-state drive.

```
sudo mkfs.ext4 /dev/vdb
```

5.3.5 Mounting

In general, it is best to avoid using the root user account for everyday tasks, and to use sudo to run specific commands as needed, rather than switching to the root user account and running all commands with superuser privileges. This helps to reduce the risk of accidental damage to the system and ensures that all actions are logged and can be traced back to a specific user.

"su" stands for "switch user," and it allows a user to switch to another user account. When you run sudo su, you are effectively running the su command with superuser privileges, which means that you can switch to the root user account and have full access to all the files and system resources on the machine.

sudo su is a command that is used to switch to the root user account from another user account on a Unix-like operating system.

"sudo" is short for "superuser do," and it allows a user to run commands with administrative privileges.

```
sudo su
```

/dev/vdb is your new disk

Stop docker daemon,

```
systemctl stop docker
```

Move your current docker directory

```
mv /var/lib/docker /var/lib/docker-backup
```

Create a new docker directory

```
mk dir var/lib/docker
```

Mount your new file system

```
mount /dev/vdb /var/lib/docker
```

Reinstate your docker installation on the new file system

```
cp -rf /var/lib/docker-backup/* /var/lib/docker
```

Start docker

```
systemctl start docker
```

Test out your newly expanded capacity by pulling a new docker image and running it — here's a postgres image as an example:

```
docker pull postgres
```

Start a postgres instance

```
docker run --name some-postgres -e POSTGRES_PASSWORD=mysecretpassword -d postgres
```

The default postgres user and database are created in the entrypoint with initdb.

The postgres database is a default database meant for use by users, utilities and third party applications.

Mount On Startup

Assuming everything is now fine, you'll want to ensure your disk gets mounted on the /var/lib/docker directory permanently on start-up:

```
nano /etc/fstab
```

Add the following line at the end of the file, again assuming /dev/vdb is your device and also assuming is formatted as an ext4 filesystem.

Rollback If something goes wrong or for whatever reason, you want to return to your previous setup — you should be able to rollback in the following way:

```
systemctl stop docker
umount /dev/sda2
mv /var/lib/docker-backup /var/lib/docker
# Remove the additional line added to /etc/fstab (if applicable)
systemctl start docker
```

Cleanup If you're happy with the increased capacity in your docker installation and all has gone to plan, you can now remove your backup:

```
rm -rf /var/lib/docker-backup
```

Change back to our ubuntu user

```
su ubuntu
```

OLD

Before we mount the drive, we create a new directory in the `/mnt/` directory where the drives are usually mounted in Ubuntu:

```
sudo mkdir /mnt/vdb
```

Once the directory is created, you can mount the drive as follows:

```
sudo mount /dev/vdb /mnt/vdb
```

To mount the drive permanently, we need to edit the file system table `fstab`. Therefore, we open the file in the text editor nano:

```
sudo nano /etc/fstab
```

Now, add the following content at the end of the file:

```
/dev/vdb    /mnt/vdb     ext4      defaults        0             0
```

`Ctrl+O` will save the modifications you've made to the file. `Ctrl+X` will close nano. Confirm with `Y` and press Enter.

Use `sudo mount` to list all mounted drives and combine it with `grep vdb`, which only returns our volume vdb:

```
sudo mount | grep vdb
```

You can change directory into your new volume:

```
cd /mnt/vdb
```

Unmount

How to unmount a mounted drive in Ubuntu

```
sudo umount -l /dev/vdb
```

References