Lab AmazonWeb Services – EC2

V0 – Guillaume Urvoy-Keller
V0.1 – Fabien Hermenier
V0.2 – Quentin Jacquemart

During this lab, you will deploy Virtual Machines (VMs, a.k.a. instances) and install a Web service. Next, leveraging the functionalities of EC2, you will first create a template and then set up an elastic Web service, that will adapt to the number of Web server instances to the actual load.

1 Creation of a first instance

1. Log on AWS (https://aws.amazon.com/) with your credentials.
2. From the EC2 console, click "Launch Instance".
   The page "Select an Amazon Machine Image (AMI)" lists the available AMIs (template). Pick (any) "Amazon Linux AMI".
3. On the page "Select an Instance Type", keep (or enforce) the default choice, namely: **t2.micro** instance.
4. Click on "Review and Launch" to continue
5. Select "Make General Purpose (SSD) the default volume for all instance launches from the console going forward"
6. Do not modify the security settings
7. Create a pair of keys with the name "aws", and download the private key.

To automate the login process on Unix/GNU-Linux/MacOS, create a file named `config` in your ~/.ssh/ folder. Move your key file there as well. Adapt the following example so that you can log to your instance by simply typing `ssh aws`. Please note that the default username for your machine is is `ec2-user`.

```
Host server1 server1.company.com
    Hostname 12.34.56.78
    User ubuntu
    IdentityFile /media/11361B1123123634/server1.pem
```

You can add multiple "Host section" in your config file, one per host. As a result typing `ssh server1` and will be equivalent to `ssh -i /media/11361B1123123634/server1.pem ubuntu@server1.company.com`.

A possible alternative on Windows machines is to use the `ssh` client PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/). Unfortunately, the `.pem` file you just downloaded (with your private key) is not supported by PuTTY. In order to log in by using your key pair, you must first convert this private key to a format usable by PuTTY. For this, run PuTTYgen, click on "Load", and select the `.pem` file (you may have to force the file dialog to show you all file extensions). Now, click on "Save private key" and choose the location to save the key to. The key is now stored in a `.ppk` format that you can use with PuTTY. Open PuTTY and enter the hostname or IP address in the [Session] window. To specify a username, go to the [Connection → Data] window, and enter the login details. Finally, to login using a private key, go to [Connection → SSH → Auth] and select the `.ppk` file in the "Authentication parameters". Go back to [Session], enter a name (e.g. server1) in "Saved sessions" and click on "Save". PuTTY will now remember the configuration for server1. You will just need to double click on the session name to initiate a connection.

2 Customize your instance

1. Connect to the VM via ssh using the previous key and the user name ec2-user. The public name of the VM is available in its description.
2. Install the required packages:

```
$ sudo yum update -y
$ sudo yum groupinstall -y "Web Server" "PHP Support"
```

3. Start the Web daemon and and enable an automatic start-up of the Web service at VM boot time:

```
$ sudo service httpd start
$ sudo chkconfig httpd on
```

Check that the server is up and running through your browser after adapting the security group of the VM via the AWS console.

When a machine is stopped, all its data is lost. A solution consists in attaching an independent disk to this VMs. Such a volume is called an Elastic Block Store and can be configured via the EC2 Dashboard:

1. Create a 10 GB disk in the same availability zone.
2. Attach the disk to your instance as /dev/sdf.
3. In a terminal of the VM, format this new partition, create a directory and modify /etc/fstab to mount is at boot time:

```
$ sudo mkfs -t ext4 /dev/sdf
$ sudo mkdir /mnt/www
$ sudo nano /etc/fstab
$ cat /etc/fstab
#
LABEL=/      /           ext4    defaults,noatime  1   1
tmpfs        /dev/shm    tmpfs   defaults          0   0
devpts       /dev/pts    devpts  gid=5,mode=620    0   0
sysfs        /sys        sysfs   defaults          0   0
proc         /proc       proc    defaults          0   0
/dev/sdf     /mnt/www    ext4    defaults          0   0
```

4. Modify the configuration file of httpd so that /mnt/www is your root directory for this service: you have to modify the variable DocumentRoot in /etc/httpd/conf/httpd.conf.
5. Create the following PHP script in /mnt/www/bench.php:

```
<?php
function fibo($n) {
  return (($n < 2) ? 1 : fibo($n - 1) + fibo($n - 2));
}

echo $_SERVER["SERVER_NAME"] . ": " . fibo(23);
?>
```

6. Restart the httpd daemon and test if the PHP script works.

3 Creating a template (AMI in AWS vocabulary)

1. Stop your instance (do not terminate it!!!). In the EC2 console, go to "Instances", select your instance and choose "Create Image".
2. Make sure the image is available in "Images → AMIs".
3. Once this is done, go to the "EC2 Dashboard" and create a new instance out of your template, using the same keys. Update your ~/.ssh/config file (or your PuTTY configuration).
4. Check that the Web server is indeed accessible from outside by testing the `bench.php` script.

4 Load Balancing

As its name indicates, the *load balancer* role is to distribute the load among several VMs.

1. Create a load balancer from the "Network & Security" menu and associate your two instances (restart the initial one). Once associated (a few minutes), modify the Health Check parameters with 30 seconds in between 2 tests and `/bench.php` as a target.
2. Find the name of the load balancer. Once it is up and running, check that the two VMs are accessible:

   ```
   $ watch nslookup -debug load_balancer_name
   ```

   In Windows, you should use the Cygwin Bash Shell to run this command.
3. What is the lifetime of a mapping between the load balancer id and the IP of one of the servers?
4. You can now stop the two VMs.

5 Horizontal Scaling

The *auto-scaling* service enables to automate scaling operations.

1. Click on "Auto-Scaling Groups" and follow the creation process by picking your AMI.
2. Continue the process to create an auto-scaling group. Between 1 and 5 instances can run. Declare a load balancer in the "Auto scaling group details" section (you can also do it after by creating a load balancer and associating as instance the auto-scaling group). Use one of your subnets in your availability-zone.
3. Next, you have to define the scaling rules. For instance:
   - create a new VM when the CPU load is over 60%
   - remove a VM when the CPU load goes below 30%
4. Your scaling group is now ready for testing. To test elasticity, code a script (or use a program like `ab`) which execute a variable amount of requests by calling the `bench.php` script.

6 Delete everything!!!

Stop your VMs and all services used or you might receive an unpleasant invoice in a few months.