ConPaaS – Administrator guide

Ismail El Helw Guillaume Pierre

September 22, 2011

Contents

1 Creating a ConPaaS image for Amazon EC2

The Web Hosting Service is capable of running over the Elastic Compute Cloud (EC2) of Amazon Web Services (AWS). This section describes the process of configuring an AWS account to run the Web Hosting Service.

If you are new to EC2, you will need to create an account at `http://aws.amazon.com/ec2/`. A very good EC2 documentation can be found at `http://docs.amazonwebservices.com/AWSEC2/latest/GettingStartedGuide/`.

1.1 Create an EBS backed AMI on Amazon EC2

The Web Hosting Service requires the creation of an Amazon Machine Image (AMI) to contain the dependencies of it's processes. The easiest method of creating a new Elastic Block Store backed Amazon Machine Image is to start from an already existing one, customize it and save the resulting filesystem as a new AMI. The following steps explains how to setup an AMI using this methodology.

1. Search the public AMIs for a Debian squeeze EBS AMI and run an instance of it. If you are going to use micro-instances then the AMI with ID `ami-e0e11289` could be a good choice.

2. Upload the `web-servers/conpaas_web_deps` script to the instance:

```
chmod 0400 yourpublickey.pem
scp -i yourpublickey.pem web-servers/conpaas_web_deps root@instancename.com:
```

3. Now, ssh to your instance:

```
chmod 0400 yourpublickey.pem
ssh -i yourpublickey.pem root@your.instancename.com:
```

   Run the `conpaas_web_deps` script inside the instance. This script will install all of the dependencies of the manager and agent processes as well as create the necessary directory structure. At some point the script requests to accept licenses, accept them.

4. Clean the filesystem by removing the `web-servers/conpaas_web_deps` file and any other temporary files you might have created.

5. Go to the EC2 administration page at the AWS website, right click on the running instance and select "*Create Image (EBS AMI)*". This automatically terminates the instance, AWS documentation is available at `http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?Tutorial_CreateImage.html`.

1.2 Create a Security Group

An AWS security group is an abstraction of a set of firewall rules to limit inbound traffic. The default policy of a new group is to deny all inbound traffic. Therefore, one needs to specify a whitelist of protocols and destination ports that are accesible from the outside. The Web Hosting Service uses TCP ports 80, 8080 and 9000. All three ports should be open for all running instances. AWS documentation is available at `http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?using-network-security.html`.

2 Creating a ConPaaS image for OpenNebula

The Web Hosting Service is capable of running over an OpenNebula installation. This section describes the process of configuring an AWS account to run the Web Hosting Service.

To create an image for OpenNebula you can execute the script `web-servers/scripts/opennebula-create-new-vm-image` in any 64-bit Debian or Ubuntu machine. Start by editing the first couple of lines of the script, then run it as root. The script generates an image file called `conpaasweb.img` by default. You can register it in OpenNebula:

```
oneimage register conpaasweb.img
```

If things go wrong

Note that if anything fails during the image file creation, the script will stop. However, it will not always reset your system to its original state. To undo everything the script has done, follow these instructions:

1. The image has been mounted as a separate file system. Find the mounted directory using command `df -h`. The directory should be in the form of `/tmp/tmp.X`.

2. There may be a `dev` and a `proc` directories mounted inside it. Unmount everything using:

```
sudo umount /tmp/tmp.X/dev /tmp/tmp.X/proc /tmp/tmp.X
```

3. Find which loop device your using:

```
sudo losetup -a
```

4. Remove the device mapping:

```
sudo kpartx -d /dev/loopX
```

5. Remove the binding of the loop device:

```
   sudo losetup -d /dev/loopX
```

6. Delete the image file

7. Your system should be back to its original state.

2.1 Make sure your OpenNebula is properly configured

There are two main topics that you should pay attention to:

1. Make sure your started OpenNebula’s OCCI deamon. ConPaaS relies on it to communicate with OpenNebula.

2. Modify the `occi_templates/common.erb` OCCI profile as follows:

```
#----
<% @vm_info.each(’OS’) do |os| %>
        <% if os.attr(’TYPE’, ’arch’) %>
          OS = [ arch = "<%= os.attr(’TYPE’, ’arch’).split(’/’).last %>" ]
        <% end %>
<% end %>
GRAPHICS = [type="vnc",listen="0.0.0.0",port="-1"]
#----
```

   (a) The match for OS TYPE:arch allows the caller to specify the architecture of the machine.

   (b) The graphics line allows for using vnc to connect to the VM. This is very useful for debugging purposes and is not necessary once testing is complete.

3 Setup ConPaaS’s Frontend

The ConPaaS frontend is a web application that allows users to manager their ConPaaS services. Users can create, configure and terminate services through it. This section describes the process of setting up a ConPaaS frontend.

3.1 Create a MySQL Database

The ConPaaS frontend uses a MySQL database to store data about users and their services. The script located in `frontend/scripts/frontend-db.sql` creates a new user `DB_USER` with password `DB_PASSWD` and a database `DB_NAME`. It grants all access permissions to user `DB_USER` on the new database. Finally, it creates the database schema. You must update the first four lines to change `DB_USER`, `DB_PASSWD` and `DB_NAME` to reasonable values.

Install a MysQL database if you don’t have one already. You can now create the database schema using this command, replacing `ADMIN` and `ADMINPASSWORD` with the MySQL administrator’s name and password:

```
mysql -u ADMIN -p ADMINPASSWORD  < frontend-db.sql
```

3.2 Configure the Front-end

The ConPaaS Front-end code is a collection of PHP scripts. It can run on any PHP-enabled Web server. We recommend using Apache with the `mod_php` module. The following instructions detail the configuration of the frontend once you have a working PHP-enabled Web server.

1. Copy all files from the `frontend/conf` directory to a location *outside* of the Web server's document root. A good location could be for example `/etc/conpaas`. Edit each of these files to setup the required configuration parameters. Each variable should be described in the config file itself.

2. Place the PHP code found in directory `frontend/www` at the document root of the target web server such that the file named `__init__.php` is directly underneath it.

3. Edit the `CONF_DIR` variable in `__init__.php` such that it points to the configuration directory path.

4. Download the AWS sdk for PHP from `http://aws.amazon.com/sdkforphp/`. Extract the sdk directory and rename it to `aws-sdk`. Place it under the lib directory of the front-end source code such that `lib/aws-sdk/` contains a file named `config-sample.inc.php` (among others).

5. Copy `lib/aws-sdk/config-sample.inc.php` to `lib/aws-sdk/config.inc.php` and fill in `AWS_KEY`, `AWS_SECRET_KEY`, `AWS_ACCOUNT_ID` and `AWS_CANONICAL_ID` as instructed in the file's documentation.

6. Make sure that the Web server's document directory contains a subdirectory named `code` and containing the following files: `agent-start`, `agent-stop`, `ConPaaSWeb.tar.gz`, `ec2-agent-user-data`, `ec2-manager-user-data`, and `manager-start`. These files contain the entire implementation of the Web hosting service. They are downloaded by newly created VM instances upon startup. Make sure that variable `SOURCE` from the frontend's configuration file `manager-user-data` points to the URL of this directory.

At this point, your front-end should be working!

4 Miscellaneous

4.1 The credit system

The frontend is designed to maintain accounting of resources used by each user. When a new user is created, (s)he receives a number of credits as specified in the "main.ini" configuration file. Later on, one credit is substracted each time a VM is executed for (a fraction of) one hour. The administrator can change the number of credits by directly editing the frontend's database.

4.2 Application sandboxing

The default ConPaaS configuration creates strong snadboxing so that applications cannot open sockets, access the file system, execute commands, etc. This makes the platform relatively secure against malicious applications. On the other hand, it strongly restricts the actions that ConPaaS applications can do. To reduce these security measures to a more usable level, you need to edit two files:

- To change restrictions applied to PHP applications, edit file `web-servers/etc/fpm.tmpl` to change the list of `disable\_functions`. Do not forget to recreate a file `ConPaaSWeb.tar.gz` out of the entire `web-servers` directory, and to copy it at the URL specified in file `frontend/conf/manager-user-data`.
- To change restrictions applied to Java applications, edit file "web-servers/etc/tomcat-catalina.policy". Do not forget to recreate a file ConPaaSWeb.tar.gz out of the entire "web-servers" directory, and to copy it at the URL specified in file "frontend/conf/manager-user-data".