ConPaaS – Installation guide

Ismail El Helw Adriana Szekeres Guillaume Pierre

ConPaaS-0.9.0

Contents

1 Installation Overview **2**

2 Using ConPaaS on Amazon EC2 **2**
2.1 Pre-built EBS Amazon Machine Image 2
2.2 Create an EBS backed AMI on Amazon EC2 2
2.3 Create a Security Group 3

3 Creating a ConPaaS image for OpenNebula **4**
3.1 Make sure OpenNebula is properly configured 5

4 Setup ConPaaS's Frontend **6**
4.1 Pre-built Frontend Amazon Machine Image 6
4.2 Manual setup 6
4.2.1 Create a MySQL Database 7
4.2.2 Configure the Front-end 7

5 Miscellaneous **10**
5.1 The credit system 10
5.2 Application sandboxing 10

6 About this document **11**

1 Installation Overview

ConPaaS is composed of two parts: a front end, and a collection of services. The front-end is a regular Web site which allows ConPaaS users to access the system. It is implemented in PHP with MySQL, and can run on any PHP-enabled Web server (inside or outside the cloud). Services are designed to run either in an OpenNebula cloud installation, or in the Amazon Web Services cloud.

Installing ConPaaS requires to take the following steps:

1. Create a VM image customized for hosting the services. Details on how to do this vary depending on the choice of cloud where ConPaaS will run. Instructions on how to create a ConPaaS image can be found in Section 2.2 (for EC2) and Section 3 for OpenNebula.

2. Setup and configure the ConPaaS frontend. All system configuration takes place in the frontend. Frontend installation and configuration is discussed in Section 4.

2 Using ConPaaS on 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. You can skip this section if you plan to install ConPaaS over OpenNebula.

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/`.

2.1 Pre-built EBS Amazon Machine Image

The Web Hosting Service requires the usage of an Amazon Machine Image (AMI) to contain the dependencies of its processes. For your convenience we provide a public ConPaaS AMI which is already configured and ready to be used on Amazon EC2. The AMI ID of said image is *ami-c4d208ad*. You can use this value when configuring your ConPaaS frontend installation as described in Section 4.

2.2 Create an EBS backed AMI on Amazon EC2

Should you decide to create a new Elastic Block Store backed Amazon Machine Image yourself, the easiest method 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. Log in the AWS management console, select the "EC2" tab, then "AMIs" in the left-side menu. 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 *conpaas-services/scripts/create_vm/ec2-setup-new-vm-image.sh* script to the instance:

```
chmod 0400 yourpublickey.pem
scp -i yourpublickey.pem \
  conpaas-services/scripts/create_vm/ec2-setup-new-vm-image.sh \
  root@instancename.com:
```

3. Now, ssh to your instance:

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

   Run the `ec2-setup-new-vm-image.sh` 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.

4. Clean the filesystem by removing the `ec2-setup-new-vm-image.sh` 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 step will take several minutes. AWS documentation is available at `http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?Tutorial_CreateImage.html`.

6. After the image has been fully created, you can return to the EC2 dashboard, right-click on your instance, and terminate it.

2.3 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 accessible from the outside. The following ports shoud be open for all running instances:

- TCP ports 80, 5555, 8000, 8080 and 9000 – used by the Web Hosting service
- TCP port 3306 – used by the MySQL service
- TCP ports 8020, 8021, 8088, 50010, 50020, 50030, 50105, 54310, 54311, 50060, 50070, 50075 and 50090 – used by the Map Reduce service

- TCP ports 4369, 14194 and 14195 – used by the Scalarix service

AWS documentation is available at `http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/index.html?using-network-security.html`.

3 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 OpenNebula to run ConPaaS. You can skip this section if you plan to deploy ConPaaS over Amazon Web Services.

To create an image for OpenNebula you can execute the script *conpaas-services/scripts/create_vm/opennebula-create-new-vm-image.sh* in any 64-bit Debian or Ubuntu machine.

1. Make sure your system has the following executables installed (they are usually located in `/sbin` or `/usr/sbin`, so make sure these directories are in your `$PATH`): *dd parted losetup kpartx mkfs.ext3 tune2fs mount debootstrap chroot umount grub-install*

2. It is particularly important that you use Grub version 2. To install it:

   ```
   sudo apt-get install grub2
   ```

3. Edit the *conpaas-services/scripts/create_vm/opennebula-create-new-vm-image.sh* script if necessary: there are two sections in the script that you might need to customize with parameters that are specific to your system. These sections are marked by comment lines containing the text "TO CUSTOMIZE:". There are comments explaining each customizable parameter.

4. Execute the image generation script as root.

5. The script generates an image file called `conpaas.img` by default. You can now register it in OpenNebula:

   ```
   cat <<EOF > /tmp/conpaas-one.image
   NAME          = "Conpaas"
   PATH          = ${PWD}/conpaas.img
   PUBLIC        = YES
   DESCRIPTION   = "Conpaas vm image"
   EOF
   oneimage register /tmp/conpaas-one.image
   ```

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.

3.1 Make sure OpenNebula is properly configured

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

1. Make sure you started OpenNebula's OCCI deamon. ConPaaS relies on it to communicate with OpenNebula.
2. At the end of the OCCI profile file `occi_templates/common.erb` from your OpenNebula installation, add the content of the file `misc/common.erb` from the ConPaaS distribution. This new version features a number of improvements from the standard version:
   - The match for `OS TYPE:arch` allows the caller to specify the architecture of the machine.
   - 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.

4 Setup ConPaaS's Frontend

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

To setup ConPaaS, you only need to setup the ConPaaS's frontend. The actual ConPaaS code is just archived and put in a folder on the frontend. The ConPaaS frontend is a web application that allows users to manage their ConPaaS services. Users can create, configure and terminate services through it. This section describes the process of setting up a ConPaaS frontend.

4.1 Pre-built Frontend Amazon Machine Image

We provide an Amazon Machine Image (AMI) with the ConPaaS frontend code and all the required dependencies already installed. This is the easiest way to get started with ConPaaS, as all you need to do is setup a few configuration files.

1. Log in the AWS management console, select the "EC2" tab, then "AMIs" in the left-side menu. Search the public AMI with ID `ami-b6d208df` and run an instance of it.

2. In the "Configure Firewall" step of the Wizard, create a security group with port 80 (HTTP) open. Also, if you haven't done so already, create a security group for the Web Hosting Service as explained in Section 2.3 and write its name down.

3. Once the instance is running, log into it and fill in the following configuration values:

   - `security_group` and `keypair` in /etc/conpaas/aws.ini
   - `USER`, `PASSWORD`, `SECURITY_GROUP_NAME` and `KEY_NAME` in /etc/conpaas/config/cloud/ec2.cfg
   - `AWS_KEY`, `AWS_SECRET_KEY`, `AWS_ACCOUNT_ID` and `AWS_CANONICAL_ID` in /var/www/lib/aws-sdk/config.inc.php

4. Point your browser to the public IP address of your instance. You should see the ConPaaS frontend web page and you should be able to create a new user and start using ConPaaS.

4.2 Manual setup

To setup your frontend, you will need a PHP-enabled web server and a MySQL database. The easiest way to install them on a Debian or Ubuntu machine is:

```
sudo apt-get install libapache2-mod-php5 php5-curl \
   php5-mysql mysql-server mysql-client
```

ConPaaS uses PHP and Python OpenSSL wrappers to secure the HTTP communication. Therefore, you must also install the openssl package. On Debian or Ubuntu, openssl should have been installed with the installation of package php5-curl, as it is one of its dependencies. Else, you could issue the command:

```
sudo apt-get install openssl
```

4.2.1 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` with the MySQL administrator's name:

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

You will be prompted for the administrator's password, then the database schema will be created automatically.

4.2.2 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. This directory contains sensitive configuration parameters which must not be accessible by external users. A good location could be for example `/etc/conpaas`. Note that files in this directory must be readable by the Web server (in Debian and Ubuntu distributions the Web server runs under username `www-data`).

   Edit the following configuration files to setup the required configuration parameters:

   - `main.ini`: general ConPaaS configuration
   - `db.ini`: information about the frontend's database location
   - `aws.ini`: information about your Amazon Web Services account (only necessary if you are installing ConPaaS on EC2)
   - `opennebula.ini`: information about your OpenNebula deployment (only necessary if you are installing ConPaaS on OpenNebula)

- `welcome.txt`: the text of the email which will be sent to each new user
- `config/cloud/ec2.cfg`: information about your Amazon Web Services account (only necessary if you are installing ConPaaS on EC2)
- `config/cloud/opennebula.cfg`: information about your OpenNebula deployment (only necessary if you are installing ConPaaS on OpenNebula)

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 frontend web server such that the file named `__init__.php` is directly underneath it.

3. Edit the `CONPAAS_CONF_DIR` and `CONPAAS_HOST` variables in `config-example.php` such that they point to the configuration directory path chosen in step 1 and to the DNS name of the frontend (or its public IP address). Rename this file `config.php`.

4. (Only if you are installing ConPaaS on EC2, and you obtained it from the svn repository) To run on EC2, the frontend uses the AWS sdk for PHP. 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 web document root such that `lib/aws-sdk/` contains a file named `config-sample.inc.php` (among others).

5. (Only if you are installing ConPaaS on EC2) Inside the web document's root, 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. (Only if you are installing ConPaaS from the svn repository) Make sure to copy folders `config/manager`, `config/cloud`, `scripts/manager` and `scripts/cloud` inside the `/etc/conpass` folder. Then edit the following file: config/cloud/opennebula.cfg or config/cloud/ec2.cfg, depending on the deployment cloud.

   Make sure that the Web server's document directory contains a subdirectory named `download`, containing the archive `ConPaaS.tar.gz`, which contains the entire implementation of the conpaas framework and services. This archive is downloaded by newly created VM instances upon startup. If you are installing ConPaaS from the svn repository, this archive can be obtained by running the mkarchive.sh, inside the `conpaas-services` folder.

As ConPaaS uses SSL certificates, you must follow two more steps to setup the secure communication. First, you must configure OpenSSL, and second, you must configure Apache (or any other web server you are using) to work with

SSL. The idea is to make certificates optional (the user is not required to provide a certificate when he/she uses the frontend, as the website is protected by login checks), except for one folder, where access must be granted only based on valid certificates. This folder contains a php script that generates new certificates as requested by the manager in the name of its agents - this folder basically contains the CA implementation.

In this guide we explain how to configure Apache or Nginx to work with SSL for ConPaaS. Of course that a correct SSL configuration depends on the existing configuration of you web server (e.g., if it host multiple websites, etc.) but we will explain ConPaaS's needs on a default configuration (clean Apache or Nginx installation) which can further be applied to your existing configuration.

7. Configure OpenSSL.

   OpenSSL has a configuration file where you can set the default values of some of the fields that appear in the certificate. On Ubuntu or Debian this file can be found in */etc/ssl/openssl.cnf*. You should fill in the following fields from the section [*req_distingushed_name*]: *countryName_default*, *stateOrProvinceName_default*.

   Also, you can make any other changes you consider. For example, you could change the number of bits used to generate a password from 1024 to 2048 by filling in the *default_bits* field in the [*req*] section.

8. Configure the webserver to work with SSL.

   First, generate the CA certificate and the certificate of the frontend with the following command (assuming that the folder you chose at step 1 to keep the sensitive configuration files is `/etc/conpaas`):

   ```
   sudo php frontend/scripts/generate-certs.php \
                                          /etc/conpaas/certs
   ```

   Second, configure the web server:

   - To configure Apache with SSL support, you have to setup a new apache site that enables SSL support. Apache comes with a default ssl site that enables Apache to listen on port 443 for requests coming to the default Virtual Host. On Debian or Ubuntu this file can be found in */etc/apache2/sites-available/default-ssl*. You can start from this file and change it as you see fit for your site. For ConPaaS, it is enough just to edit a few lines in the existing default-ssl site in Apache:

     - inside <*Directory /var/www/*> change *AllowOverride* to All (we provide a .htaccess file whose instructions must be taken into account). (TODO: Check this. Note: Probably you could make SSL mandatory for a given folder directly from the default-ssl file)

- change *SSLCertificateFile* and *SSLCertificateKeyFile* to point to */etc/conpaas/certs/cert.pem* and */etc/conpaas/certs/key.pem*, respectively.

- uncomment the line containing the *SSLCACertificateFile* variable and make it point to */etc/conpaas/certs/ca_cert.pem*.

Enable SSL and the SSL site with:

```
sudo a2enmod ssl
sudo a2ensite default-ssl
sudo /etc/init.d/apache2 reload
```

- To configure Nginx with SSL support TODO:

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

5 Miscellaneous

5.1 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 subtracted 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.

5.2 Application sandboxing

The default ConPaaS configuration creates strong sandboxing 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".

6 About this document