

ConPaaS – User Guide

Ismail El Helw Guillaume Pierre

ConPaaS-0.9.0

Contents

1	Introduction	2
2	Usage overview	2
2.1	Controlling services using the front-end	3
2.2	Controlling services using the command-line interfaces	4
2.3	The credit system	5
3	Tutorial: hosting WordPress in ConPaaS	5
4	The PHP Web hosting service	6
4.1	Uploading application code	6
4.2	Access the application	6
4.3	Using PHP sessions	6
4.4	Debug mode	7
5	The Java Web hosting service	7
5.1	Uploading application code	7
5.2	Access the application	7
6	The MySQL database service	7
6.1	Resetting the user password	8
6.2	Accessing the database	8
6.3	Uploading a database dump	8
7	The Scalarix key-value store service	8
7.1	Accessing the key-value store	8
7.2	Managine the key-value store	9
8	The MapReduce service	9

9	The TaskFarm service	9
9.1	The bag of tasks file	9
9.2	The filesystem location	9
9.3	The demo mode	10
10	Building new types of services	10
11	About this document	10

1 Introduction

ConPaaS is an open-source runtime environment for hosting applications in the cloud which aims at offering the full power of the cloud to application developers while shielding them from the associated complexity of the cloud.

ConPaaS is designed to host both high-performance scientific applications and online Web applications. It runs on a variety of public and private clouds, and is easily extensible. ConPaaS automates the entire life-cycle of an application, including collaborative development, deployment, performance monitoring, and automatic scaling. This allows developers to focus their attention on application-specific concerns rather than on cloud-specific details.

ConPaaS is organized as a collection of **services**, where each service acts as a replacement for a commonly used runtime environment. For example, to replace a MySQL database, ConPaaS provides a cloud-based MySQL service which acts as a high-level database abstraction. The service uses real MySQL databases internally, and therefore makes it easy to port a cloud application to ConPaaS. Unlike a regular centralized database, however, it is self-managed and fully elastic: one can dynamically increase or decrease its processing capacity by requesting it to reconfigure itself with a different number of virtual machines.

ConPaaS currently contains six services:

- **Two Web hosting services** respectively specialized for hosting PHP and JSP applications;
- **MySQL database** service;
- **Scalarix service** offering a scalable in-memory key-value store;
- **MapReduce service** providing the well-known high-performance computation framework;
- **TaskFarming service** high-performance batch processing.

ConPaaS applications can be composed of any number of services. For example, a bio-informatics application may make use of a PHP and a MySQL service to host a Web-based frontend, and link this frontend to a MapReduce backend service for conducting high-performance genomic computations on demand.

2 Usage overview

Most operations in ConPaaS can be done using the ConPaaS frontend, which gives a Web-based interface to the system. The front-end allows users to register, create services, upload code and data to the services, and configure each service.

- The Dashboard page displays the list of services currently active in the system.

- Each service comes with a separate page which allows one to configure it, upload code and data, and scale it up and down.

All the functionalities of the frontend are also available using a command-line interface. This allows one to script commands for ConPaaS. The command-line interface also features additional advanced functionalities, which are not available using the front-end.

2.1 Controlling services using the front-end

The ConPaaS front-end provides a simple and intuitive interface for controlling services. We discuss here the features that are common to all services, and refer to the next sections for service-specific functionality.

Create a service. Click on “create new service”, then select the service you want to create. This operation starts a new “Manager” virtual machine instance. The manager is in charge of taking care of the service, but it does not host applications itself. Other instances in charge of running the actual application are called “agent” instances.

Start a service. Click on “start”, this will create a new virtual machine which can host applications, depending on the type of service.

Rename the service. By default all new services are named “New service.” To give a meaningful name to a service, click on this name in the service-specific page and enter a new name.

Check the list of virtual instances. A service can run using one or more virtual machine instances. The service-specific page shows the list of instances, their respective IP addresses, and the role each instance is currently having in the service. Certain services use a single role for all instances, while other services specialize different instances to take different roles. For example, the PHP Web hosting service distinguishes three roles: load balancers, web servers, and PHP servers.

Scale the service up and down. When a service is started it uses a single “agent” instance. To add more capacity, or to later reduce capacity you can vary the number of instances used by the service. Click the numbers below the list of instances to request adding or removing servers. The system reconfigures itself without any service interruption.

Stop the service. When you do not need to run the application any more, click “stop” to stop the service. This stops all instances except the manager which keeps on running.

Terminate the service. Click “terminate” to terminate the service. At this point all the state of the service manager will be lost.

2.2 Controlling services using the command-line interfaces

Command-line interfaces allow one to control services without using the graphical interface. The command-line interfaces also offer additional functionality for advanced usage of the services.

Each service can be controlled using its own command-line tool:

- The two Web hosting services are controlled using `cpsclient.web`
- The MySQL service is controlled using `cpsclient.mysql`
- The Scalarix service is controlled using `cpsclient.scalarix`
- The MapReduce service is controlled using `cpsclient.mapreduce`

To use these tools you must first set the `PYTHONPATH` variable to contain the full path name of the “src” subdirectory within the ConPaaS directory:

```
$ export PYTHONPATH=/path/to/conpaas/src
```

The `cpsclient.*` tools always take the URL of the service manager as its first argument. This URL is provided by the front-end.

List all options of the command-line tool.

```
$ ./cpsclient.servicename help
```

Start the service.

```
$ ./cpsclient.servicename http://x-x-x-x/ startup
```

Stop the service.

```
$ ./cpsclient.servicename http://x-x-x-x/ shutdown
```

Scale the service up and down.

```
$ ./cpsclient.servicename http://x-x-x-x/ add_nodes -h
$ ./cpsclient.servicename http://x-x-x-x/ remove_nodes -h
$ ./cpsclient.web http://x-x-x-x/ add_nodes -w 1 -b 1
$ ./cpsclient.web http://x-x-x-x/ remove_nodes -w 1 -b 1
```

2.3 The credit system

In Cloud computing, resources come at a cost. ConPaaS reflects this reality in the form of a credit system. Each user is given a number of credits that she can use as she wishes. One credit corresponds to one hour of execution of one virtual machine. The number of available credits is always mentioned in the top-right corner of the front-end. Once credits are exhausted, your running instances will be stopped and you will not be able to use the system until the administrator decides to give additional credit.

Note that every service consumes credit, even if it is in “stopped” state. The reason is that stopped services still have one “manager” instance running. To stop using credits you must completely terminate your services.

3 Tutorial: hosting WordPress in ConPaaS

This short tutorial illustrates the way to use ConPaaS to install and host WordPress (<http://www.wordpress.org>), a well-known third-party Web application. WordPress is implemented in PHP using a MySQL database so we will need a PHP and a MySQL service in ConPaaS.

1. Open the ConPaaS front-end in your Web browser and log in. If necessary, create yourself a user account and make sure that you have at least 5 credits. Your credits are always shown in the top-right corner of the front-end. One credit corresponds to one hour of execution of one virtual machine instance.
2. Create a MySQL service, start it, reset its password. Copy the IP address of the master node somewhere, we will need it in step 5.
3. Create a PHP service, start it.
4. Download a Wordpress tarball from <http://www.wordpress.org>, and expand it in your computer.
5. Copy file `wordpress/wp-config-sample.php` to `wordpress/wp-config.php` and edit the `DB_NAME`, `DB_USER`, `DB_PASSWORD` and `DB_HOST` variables to point to the database service. You can choose any database name for the `DB_NAME` variable as long as it does not contain any special character. We will reuse the same name in step 7.
6. Rebuild a tarball of the directory such that it will expand in the current directory rather than in a `wordpress` subdirectory. Upload this tarball to the PHP service, and make the new version active.
7. Connect to the database using the command proposed by the frontend. Create a database of the same name as in step 5 using command `"CREATE DATABASE databasename;"`

8. Open the page of the PHP service, and click “access application.” Your browser will display nothing because the application is not fully installed yet. Visit the same site at URL `http://xxx.yyy.zzz.ttt/wp-admin/install.php` and fill in the requested information (site name etc).
9. That’s it! The system works, and can be scaled up and down.

Note that the “file upload” functionality of WordPress will not work if you scale the system up. This is because WordPress stores files in the local file system of the PHP server where the upload has been processed. If a subsequent request for this file is processed by another PHP server then the file will not be found. In a next ConPaaS release we will provide a shared file system service which will allow one to avoid this issue.

4 The PHP Web hosting service

The PHP Web hosting service is dedicated to hosting Web applications written in PHP. It can also host static Web content.

4.1 Uploading application code

PHP applications can be uploaded in the form of a `tar` or `zip` archive. Attention: the archive must expand *in the current directory* rather than in a subdirectory. The service does not immediately use new applications when they are uploaded. The frontend shows the list of versions that have been uploaded; choose one version and click “make active” to activate it.

Note that the frontend only allows uploading archives smaller than a certain size. To upload large archives, you must use the command-line tool:

```
$ ./cpsclient.web http://x-x-x-x/ upload_code_version path/to/archive.zip
```

4.2 Access the application

The frontend gives a link to the running application. This URL will remain valid as long as you do not stop the service.

4.3 Using PHP sessions

PHP normally stores session state in its main memory. When scaling up the PHP service, this creates problems because multiple PHP servers running in different VM instances cannot share their memory. To support PHP sessions the PHP service features a key-value store where session states can be transparently stored. To overwrite PHP session functions such that they make use of the shared key-value store, the PHP service includes a standard “phpsession.php” file at the beginning of every .php file of your application that uses sessions, i.e. in which function `session_start()` is encountered. This file overwrites the session handlers using the `session_set_save_handler()` function.

This modification is transparent to your application so no particular action is necessary to use PHP sessions in ConPaaS.

4.4 Debug mode

By default the PHP service does not display anything in case PHP errors occur while executing the application. This setting is useful for production, when you do not want to reveal internal information to external users. While developing an application it is however useful to let PHP display errors.

```
$ ./cpsclient.web http://x-x-x-x/ update_php_configuration \
  -p "display_errors=On"
```

5 The Java Web hosting service

The Java Web hosting service is dedicated to hosting Web applications written in Java using JSP or servlets. It can also host static Web content.

5.1 Uploading application code

Applications in the Java Web hosting service can be uploaded in the form of a `war` file. The service does not immediately use new applications when they are uploaded. The frontend shows the list of versions that have been uploaded; choose one version and click “make active” to activate it.

Note that the frontend only allows uploading archives smaller than a certain size. To upload large archives, you must use the command-line tool.

```
$ ./cpsclient.web http://x-x-x-x/ upload_code_version path/to/archive.war
```

5.2 Access the application

The frontend gives a link to the running application. This URL will remain valid as long as you do not stop the service.

6 The MySQL database service

The MySQL service provides the famous database in the form of a ConPaaS service. When scaling the service up and down, it creates (or deletes) database replicas using the master-slave mechanism. At the moment, the service does not implement load balancing of database queries between the master and its slaves. Replication therefore provides fault-tolerance properties but no performance improvement.

6.1 Resetting the user password

When a MySQL service is started, a new user `mysqlldb` is created with a randomly-generated password. To gain access to the database you must first reset this password. Click “Reset password” in the front-end, and choose the new password.

Note that the user password is *not* kept by the ConPaaS frontend. If you forget the password the only thing you can do is reset the password again to a new value.

6.2 Accessing the database

The frontend provides the command-line to access the database. Copy-paste this command in a terminal. You will be asked for the user password, after which you can use the database as you wish.

Note that the `mysqlldb` user has extended privileges. It can create new databases, new users etc.

6.3 Uploading a database dump

The ConPaaS frontend allows to easily upload database dumps to a MySQL service. Note that this functionality is restricted to dumps of a relatively small size. To upload larger dumps you can always use the regular `mysql` command for this:

```
$ mysql mysql-ip-address -u mysqlldb -p < dumpfile.sql
```

7 The Scalarix key-value store service

The Scalarix service provides an in-memory key-value store. It is highly scalable and fault-tolerant. This service deviates slightly from the organization of other services in that it does not have a separate manager virtual machine instance. Scalarix is fully symmetric so any scalarix node can act as a service manager.

7.1 Accessing the key-value store

Clients of the Scalarix service need the IP address of (at least) one node to connect to the service. Copy-paste the address of any of the running instances in the client. A good choice is the first instance in the list: when scaling the service up and down, other instances may be created or removed. The first instance will however remain across these reconfigurations, until the service is terminated.

7.2 Manage the key-value store

Scalarix provides its own Web-based interface to monitor the state and performance of the key-value store, manually add or query key-value pairs, etc. For convenience reasons the ConPaaS front-end provides a link to this interface.

8 The MapReduce service

The MapReduce service provides the well-known Apache Hadoop framework in ConPaaS. Once the MapReduce service is created and started, the front-end provides useful links to the Hadoop namenode, the job tracker, and to a graphical interface which allows to upload/download data to/from the service and issue MapReduce jobs.

9 The TaskFarm service

The TaskFarm service provides a bag of tasks scheduler for ConPaaS. The user needs to provide a list of independent tasks to be executed on the cloud and a file system location where the tasks can read input data and/or write output data to it. The service first enters a sampling phase, where its agents sample the runtime of the given tasks on different cloud instances. The service then based on the sampled runtimes, provides the user with a list of schedules. Schedules are presented in a graph and the user can choose between cost/makespan of different schedules for the given set of tasks. After the choice is made the service enters the execution phase and completes the execution of the rest of the tasks according to the user's choice.

9.1 The bag of tasks file

The bag of tasks file is a simple plain text file that contains the list of tasks along with their arguments to be executed. The tasks are separated by new lines. This file needs to be uploaded to the service, before the service can start sampling. Below is an example of a simple bag of tasks file containing three tasks:

```
/bin/sleep 1 && echo "slept for 1 seconds" >> /mnt/xtreemfs/log  
/bin/sleep 2 && echo "slept for 2 seconds" >> /mnt/xtreemfs/log  
/bin/sleep 3 && echo "slept for 3 seconds" >> /mnt/xtreemfs/log
```

9.2 The filesystem location

TaskFarm service uses XtreamFS for data input/output. The actual task code can also reside in the XtreamFS. The user can optionally provide an XtreamFS location which is then mounted on TaskFarm agents.

9.3 The demo mode

With large bags of tasks and/or with long running tasks, the TaskFarm service can take a long time to execute the given bag. The service provides its users with a progress bar and reports the amount money spent so far. TaskFarm service also provides a “demo” mode where the users can try the service with custom bags without spending time and money.

10 Building new types of services

The architecture of ConPaaS allows developers to build new types of services. To learn how to do this, please check the “Internals” ConPaaS documentation.

11 About this document

Copyright (c) 2010-2012, Contrail consortium.
All rights reserved.

Redistribution and use in source and binary forms,
with or without modification, are permitted provided
that the following conditions are met:

1. Redistributions of source code must retain the
above copyright notice, this list of conditions
and the following disclaimer.
2. Redistributions in binary form must reproduce
the above copyright notice, this list of
conditions and the following disclaimer in the
documentation and/or other materials provided
with the distribution.
3. Neither the name of the Contrail consortium nor the
names of its contributors may be used to endorse
or promote products derived from this software
without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,

WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.