



Grant Agreement No.: 723174

Call: H2020-ICT-2016-2017

Topic: ICT-38-2016 - MEXICO: Collaboration on ICT

Type of action: RIA



## D3.6: SmartSDK Platform Manager

Revision: v.2.0

Work package	WP 3
Task	Task 3.3
Due date	31/05/2018
Submission date	31/05/2018
Deliverable lead	FBK
Version	2.0
Authors	Daniele Pizzolli (FBK), Daniel Zozin (FBK)
Reviewers	Tomas Aliaga (MARTEL), Miguel G. Mendoza (ITESM)

Abstract	This deliverable documents the software usage, installation and maintenance of the SmartSDK platform. Particular emphasis is reserved to the integration with the FIWARE Lab and to the deployment of SmartSDK recipes.
Keywords	Container Orchestration, FIWARE Lab, docker, rancher, docker-compose, docker stack

## Disclaimer

The information, documentation and figures available in this deliverable, is written by the SmartSDK (A FIWARE-based Software Development Kit for Smart Applications for the needs of Europe and Mexico) – project consortium under EC grant agreement 723174 and does not necessarily reflect the views of the European Commission. The European Commission is not liable for any use that may be made of the information contained herein.

## Copyright notice

© 2016 – 2018 SmartSDK Consortium

Project co-funded by the European Commission in the H2020 Programme		
Nature of the deliverable:		R
Dissemination Level		
PU	Public, fully open, e.g. web	✓
CI	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to SmartSDK project and Commission Services	

*\* R: Document, report (excluding the periodic and final reports)*

*DEM: Demonstrator, pilot, prototype, plan designs*

*DEC: Websites, patents filing, press & media actions, videos, etc.*

*OTHER: Software, technical diagram, etc.*

## Document Revision History

Version	Date	Description of change	List of contributor(s)
V1.0	2017/05/31	Initial Release	Daniele Pizzolli (FBK), Daniel Zozin (FBK)
V2.0	2018/05/31	Update	Daniele Pizzolli (FBK), Daniel Zozin (FBK)

## EXECUTIVE SUMMARY

---

The SmartSDK Platform Manager allows to register, configure, manage and monitor the deployment of SmartSDK recipes.

It can be installed on the FIWARE Lab nodes. It allows the creation of fully separated environments. Every environment is composed by grouping a set of hosts. Adding an host to an existing environment is a straightforward procedure and a number of cloud providers are already supported, including the FIWARE Lab itself.

This deliverable documents why Rancher was selected as the base software for the SmartSDK Platform, the usage of the SmartSDK Platform Manager, highlights the main use cases, and offers some advice in order to deploy the SmartSDK recipes.

The final part will list all the references to the source code and documentation.

## TABLE OF CONTENTS

<b>EXECUTIVE SUMMARY .....</b>	<b>3</b>
<b>TABLE OF CONTENTS .....</b>	<b>4</b>
<b>LIST OF FIGURES .....</b>	<b>7</b>
<b>ABBREVIATIONS .....</b>	<b>9</b>
<b>1 INTRODUCTION .....</b>	<b>10</b>
1.1 The SmartSDK Platform Manager .....	10
1.2 SmartSDK Overall Architecture .....	11
1.3 Structure of the deliverable .....	12
1.4 Audience.....	13
<b>2 PLATFORM-MANAGER USAGE .....</b>	<b>14</b>
2.1 Introduction .....	14
2.2 Register in FIWARE Lab .....	14
2.3 Configure your cluster .....	16
2.4 Setup Swarm on Fiware Lab .....	18
2.5 Deploy your cluster .....	22
2.6 Deploy a stack using the web interface .....	30
2.7 Export configuration for Docker CLI .....	40
<b>3 SMARTSDK PLATFORM USAGE.....</b>	<b>48</b>
3.1 Introduction .....	48
3.2 Environment Templates .....	48
3.2.1 The Number of Swarm Managers .....	48
3.2.2 Rancher IPsec plugin MTU (Fiware LAB) .....	49
3.2.3 Configure the Rancher IPsec plugin MTU .....	50
3.2.4 (Optional) Rancher NFS plugin .....	50
3.3 Environment .....	50
3.4 Host requirements.....	50
3.5 Hosts on the FIWARE Lab .....	50
3.6 Setup the OpenStack project for hosting a SmartSDK platform environment.....	50
3.7 OpenStack client Setup .....	50
3.7.1 Install python-openstackclient.....	51
3.7.2 Load the OpenStack client settings .....	51
3.8 Start with a clean OpenStack and docker-machine environment.....	51
3.9 Add proper security group roles .....	52
3.10 Docker Machine .....	53

3.10.1	Resolve dependencies for docker-machine .....	53
3.10.2	Install the docker-machine .....	53
3.10.3	Start with a clean docker-machine environment .....	53
3.10.4	Setup docker-machine from the command line interface .....	54
3.11	Setup security groups .....	54
3.12	Download and install Rancher CLI .....	55
3.13	Create API keys .....	55
3.14	Write CLI configuration .....	56
3.15	Provision of Rancher hosts using machine drivers .....	56
3.15.1	Set host parameters .....	56
3.15.2	Create and access hosts .....	58
<b>4</b>	<b>SMARTSDK PLATFORM ADVANCED USAGE .....</b>	<b>59</b>
4.1	User management integrated with FIWARE Lab OAuth .....	59
4.2	Enable the FIWARE Lab OAuth .....	60
4.3	Machine driver and User Interface Plugin for FIWARE Lab Nodes .....	63
4.4	Using a VPN for overcoming NAT issues .....	63
4.5	Provision of Rancher hosts using custom hosts .....	64
<b>5</b>	<b>DEPLOY SMARTSDK RECIPES ON SMARTSDK PLATFORM .....</b>	<b>65</b>
5.1	Deployment using docker stack deploy .....	65
5.2	Test deployments .....	65
5.2.1	Swarm deployment with replication (docker stack) .....	65
5.2.2	Deploy smartsdk-recipes using the CLI .....	66
<b>6</b>	<b>SMARTSDK PLATFORM INSTALLATION .....</b>	<b>67</b>
6.1	Register the SmartSDK Platform on FIWARE Lab .....	68
6.2	Install and configure the FIWARE Lab Rancher UI driver .....	74
6.3	Edit Docker Swarm Settings .....	79
<b>7</b>	<b>CONCLUSION .....</b>	<b>82</b>
	<b>APPENDIX A .....</b>	<b>83</b>
A.1	How Docker Swarm networking works .....	83
A.2	How exposing services through load balancers works .....	83
A.3	Issues with Rancher hosts for natted machines .....	83
A.4	Advanced analysis of the issues related to non-standard MTU usage .....	84
A.5	Install modern openstackclient with pip .....	84
A.6	List Available Images in an OpenStack Project .....	85
A.7	List Available Flavors in an OpenStack Project .....	85
A.8	Useful script to manage Docker .....	85

A.9 Change the MTU of all interfaces to 1400 ..... 85

A.10 Workaround for sudo complaint..... 85

A.11 Add the docker group to the current user ..... 86

A.12 Workaround to view display error on FIWARE Lab portal..... 86

A.13 Rancher General cleanup..... 86

## LIST OF FIGURES

---

<i>Figure 1: The SmartSDK overall picture.....</i>	<i>10</i>
<i>Figure 2: Simple overview of the SmartSDK architecture.....</i>	<i>12</i>
<i>Figure 3: Simple overview of the SmartSDK usage .....</i>	<i>12</i>
<i>Figure 4: Home page account portal of FIWARE Lab.....</i>	<i>14</i>
<i>Figure 5: Empty form on Sign up of FIWARE Lab.....</i>	<i>15</i>
<i>Figure 6: Example of form Completion on Sign up of FIWARE Lab .....</i>	<i>16</i>
<i>Figure 7: Home page of SmartSDK Platform Manager .....</i>	<i>17</i>
<i>Figure 8: Fiware Lab Login Page.....</i>	<i>17</i>
<i>Figure 9: Fiware Lab Authorization Request.....</i>	<i>18</i>
<i>Figure 10: Manage Environments Menu .....</i>	<i>19</i>
<i>Figure 11: Add Environment .....</i>	<i>19</i>
<i>Figure 12: Select Fiware Swarm as the Environment Template .....</i>	<i>20</i>
<i>Figure 13: Select newly created environment.....</i>	<i>21</i>
<i>Figure 14: Add host warning .....</i>	<i>22</i>
<i>Figure 15: Add new hosts driver selection .....</i>	<i>23</i>
<i>Figure 16: Insert FIWARE credentials .....</i>	<i>24</i>
<i>Figure 17: Select host configuration .....</i>	<i>25</i>
<i>Figure 18: Region selection.....</i>	<i>26</i>
<i>Figure 19: Add hosts configuration details .....</i>	<i>27</i>
<i>Figure 20: Save hosts configuration .....</i>	<i>29</i>
<i>Figure 21: Wait for hosts.....</i>	<i>30</i>
<i>Figure 22: Host added .....</i>	<i>31</i>
<i>Figure 23: Portainer Template Settings.....</i>	<i>32</i>
<i>Figure 24: Network Creation .....</i>	<i>33</i>
<i>Figure 25: Application listing .....</i>	<i>34</i>
<i>Figure 26: Orion Context Broker Application settings .....</i>	<i>35</i>
<i>Figure 27: Link to the Original Context Broker Documentation .....</i>	<i>36</i>
<i>Figure 28: Complete the Orion Context Broker Form.....</i>	<i>37</i>
<i>Figure 29: Successful start of a deploy.....</i>	<i>38</i>
<i>Figure 30: Edit the configuration of a running deploy.....</i>	<i>39</i>
<i>Figure 31: Add hosts configuration details .....</i>	<i>40</i>
<i>Figure 32: Download Rancher CLI .....</i>	<i>42</i>
<i>Figure 33: API Creation and download page.....</i>	<i>43</i>
<i>Figure 34: New account API key creation.....</i>	<i>44</i>

<b>Figure 35: Account key tokens .....</b>	<b>45</b>
<b>Figure 36: New environment key creation.....</b>	<b>46</b>
<b>Figure 37: Environment key tokens.....</b>	<b>47</b>
<b>Figure 38: Setting the manager number in environment template settings .....</b>	<b>49</b>
<b>Figure 39: Screenshot of the Access control Setup for FIWARE Lab OAuth.....</b>	<b>60</b>
<b>Figure 40: Login on the account of FIWARE Lab .....</b>	<b>69</b>
<b>Figure 41: Summary on the account of FIWARE Lab.....</b>	<b>70</b>
<b>Figure 42: Application Info on FIWARE Lab .....</b>	<b>71</b>
<b>Figure 43: Upload optional logo on FIWARE Lab.....</b>	<b>71</b>
<b>Figure 44: Resize logo on FIWARE Lab .....</b>	<b>72</b>
<b>Figure 45: Select app Roles on FIWARE Lab .....</b>	<b>73</b>
<b>Figure 46: Recap of app on FIWARE Lab.....</b>	<b>74</b>
<b>Figure 47: Machine drivers menu .....</b>	<b>75</b>
<b>Figure 48: Enable the FIWARE Lab Rancher UI driver .....</b>	<b>75</b>
<b>Figure 49: Add new Machine Driver .....</b>	<b>76</b>
<b>Figure 50: Driver list.....</b>	<b>77</b>
<b>Figure 51: Admin Settings.....</b>	<b>78</b>
<b>Figure 52: Admin advanced settings.....</b>	<b>78</b>
<b>Figure 53: Admin Advanced Setting edit .....</b>	<b>79</b>
<b>Figure 54: Setting api.proxy.whitelist .....</b>	<b>79</b>
<b>Figure 55: Select “Default Swarm template” .....</b>	<b>80</b>
<b>Figure 56: Edit config of Rancher IPSEC .....</b>	<b>80</b>
<b>Figure 57: End of Template configuration.....</b>	<b>81</b>



## **ABBREVIATIONS**

---

<b>API</b>	Application Programming Interface
<b>CLI</b>	Command Line Interface
<b>DNS</b>	Domain Name System
<b>FQDN</b>	Fully Qualified Domain Name
<b>HA</b>	High Availability
<b>HTTP</b>	Hypertext Transfer Protocol
<b>IP</b>	Internet Protocol
<b>TCP</b>	Transmission Control Protocol
<b>UI</b>	User Interface
<b>VPN</b>	Virtual Private Network
<b>VM</b>	Virtual Machine

# 1 INTRODUCTION

The SmartSDK Platform Manager allows to:

- ➔ Register
- ➔ Configure
- ➔ Manage
- ➔ Monitor

the deployment of SmartSDK recipes. The documentation of SmartSDK recipes is detailed in the SmartSDK recipes deliverable 1 . The relationship with the other components of SmartSDK is detailed in the *Figure 1: The SmartSDK overall picture*.

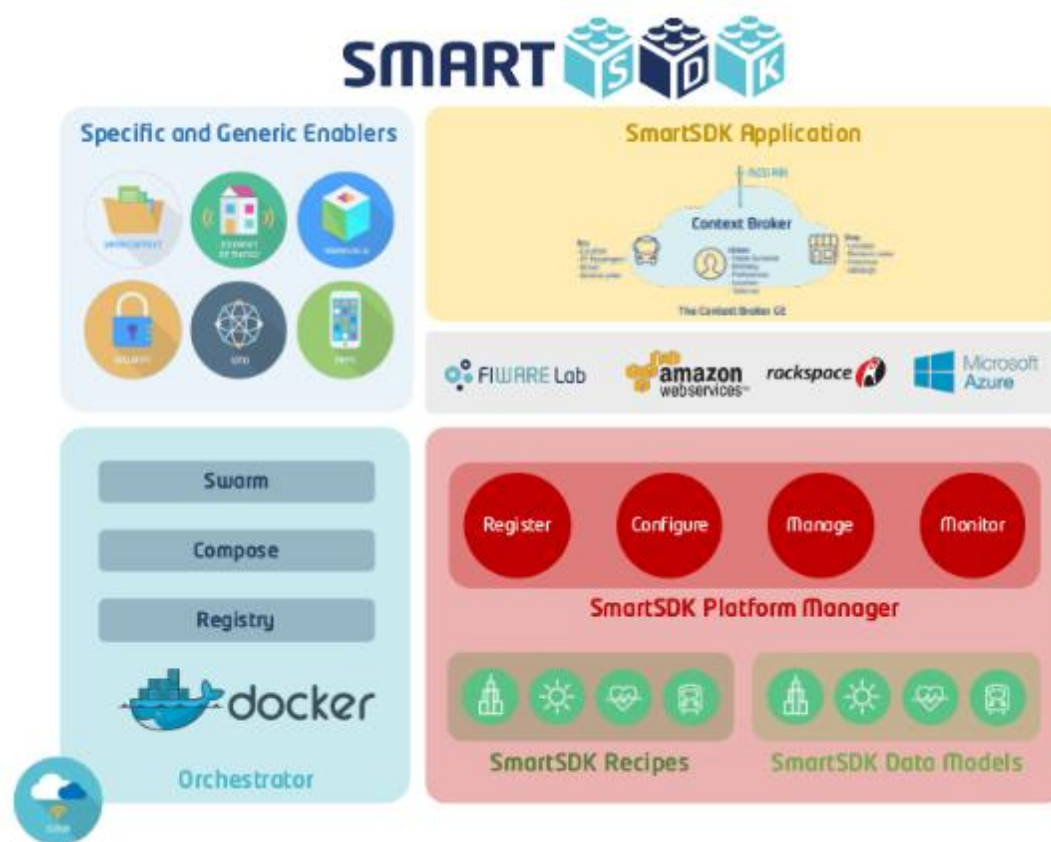


Figure 1: The SmartSDK overall picture

## 1.1 The SmartSDK Platform Manager

The SmartSDK Platform Manager duties are related to the deployment of SmartSDK recipes. It must also be compliant with the “design principles” of the SmartSDK project. Rancher 2 has been chosen as we have evaluated it to be mature enough to support the project initial needed features, and in the

<sup>1</sup> [https://smartsdk.media.martel-innovate.com/wp-content/uploads/sites/8/2017/03/SmartSDK\\_D3.1v1.0\\_FINAL.pdf](https://smartsdk.media.martel-innovate.com/wp-content/uploads/sites/8/2017/03/SmartSDK_D3.1v1.0_FINAL.pdf)

<sup>2</sup> <https://rancher.com/>

reasonable future all other features as well. In the areas where Rancher is currently lacking, we have developed custom extensions and documented suitable workarounds when required.

So far, at the final state of the project, it is possible to use the SmartSDK Platform Manager to deploy SmartSDK recipes on the FIWARE Lab.

Rancher has been chosen as the SmartSDK Platform Manager because it fits well the needs and the requirements of the project. It also respects all the “design principles” of the SmartSDK project, specifically:

- ➔ **Restful APIs.** Rancher offers most of its functionalities via API and are well documented on the [Rancher API documentation site](http://docs.rancher.com/rancher/v1.6/en/api/v2-beta/)<sup>3</sup>.
- ➔ **Reusability and Openness.** Rancher is released under an [Apache License Version 2.0](https://github.com/rancher/rancher/blob/master/LICENSE)<sup>4</sup>, The development is done on [github](https://github.com/rancher/rancher)<sup>5</sup>. Rancher is still a young project, currently there are over 1500 open issues, but over 7000 were closed since November 2014. Rancher relies on other third party technologies, all of them are released under an Open Source License.
- ➔ **Cloudification and Microservices.** The Rancher application itself is made by different components, respecting the good design patterns for microservices-based application. The deployment of applications by Rancher can be done using a catalog or Compose stack recipes, currently one of the most advanced ways available to deploy applications in the cloud.
- ➔ **Market and community relevance.** Rancher has a very active community, that mostly discuss on the [official forum](https://forums.rancher.com/)<sup>6</sup>. By supporting the deployment of application using docker swarm mode and [Kubernetes](https://kubernetes.io/)<sup>7</sup> it can be adopted by two broad and growing communities.

## 1.2 SmartSDK Overall Architecture

The SmartSDK Platform Manager uses Rancher 8 as a base to offer its services. A SmartSDK Platform Manager user will be able to instantiate one or multiple “environments” in order to deploy his application. See *Figure 2: Simple overview of the SmartSDK architecture* for a component representation. See *Figure 3: Simple overview of the SmartSDK usage* for a reference of the steps involved.

---

<sup>3</sup> <http://docs.rancher.com/rancher/v1.6/en/api/v2-beta/>

<sup>4</sup> <https://github.com/rancher/rancher/blob/master/LICENSE>

<sup>5</sup> <https://github.com/rancher/rancher>

<sup>6</sup> <https://forums.rancher.com/>

<sup>7</sup> <https://kubernetes.io/>

<sup>8</sup> <https://rancher.com/>

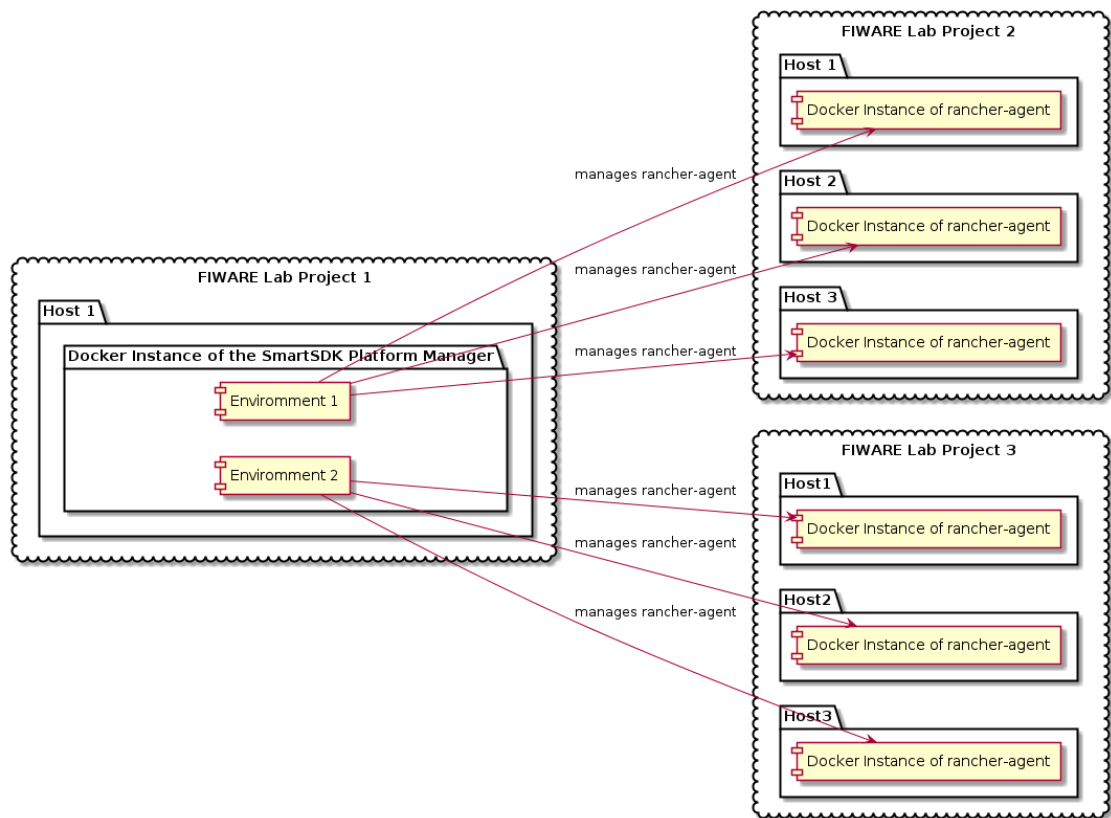


Figure 2: Simple overview of the SmartSDK architecture



Figure 3: Simple overview of the SmartSDK usage

### 1.3 Structure of the deliverable

The deliverable is structured as follow:

- ➔ INTRODUCTION introduces the main concepts of the SmartSDK Platform.
- ➔ PLATFORM-MANAGER USAGE introduces the usage of the already configured instance of the platform-manager.

- ➔ SMARTSDK PLATFORM USAGE introduces the generic usage of SmartSDK platform.
- ➔ SMARTSDK PLATFORM ADVANCED USAGE details some advanced configurations.
- ➔ DEPLOY SMARTSDK RECIPES ON SMARTSDK PLATFORM shows some SmartSDK recipes deployment examples.
- ➔ SMARTSDK PLATFORM INSTALLATION introduces the installation and administration of SmartSDK.
- ➔ CONCLUSION summarizes the outcomes of the project with all the references to the platform manager instance, source code and documentation.

## 1.4 Audience

This deliverable is mainly intended for:

- ➔ Developers interested into deploying SmartSDK application recipes.
- ➔ Operators interested into adopting the SmartSDK Platform Manager in a production context.

## 2 PLATFORM-MANAGER USAGE

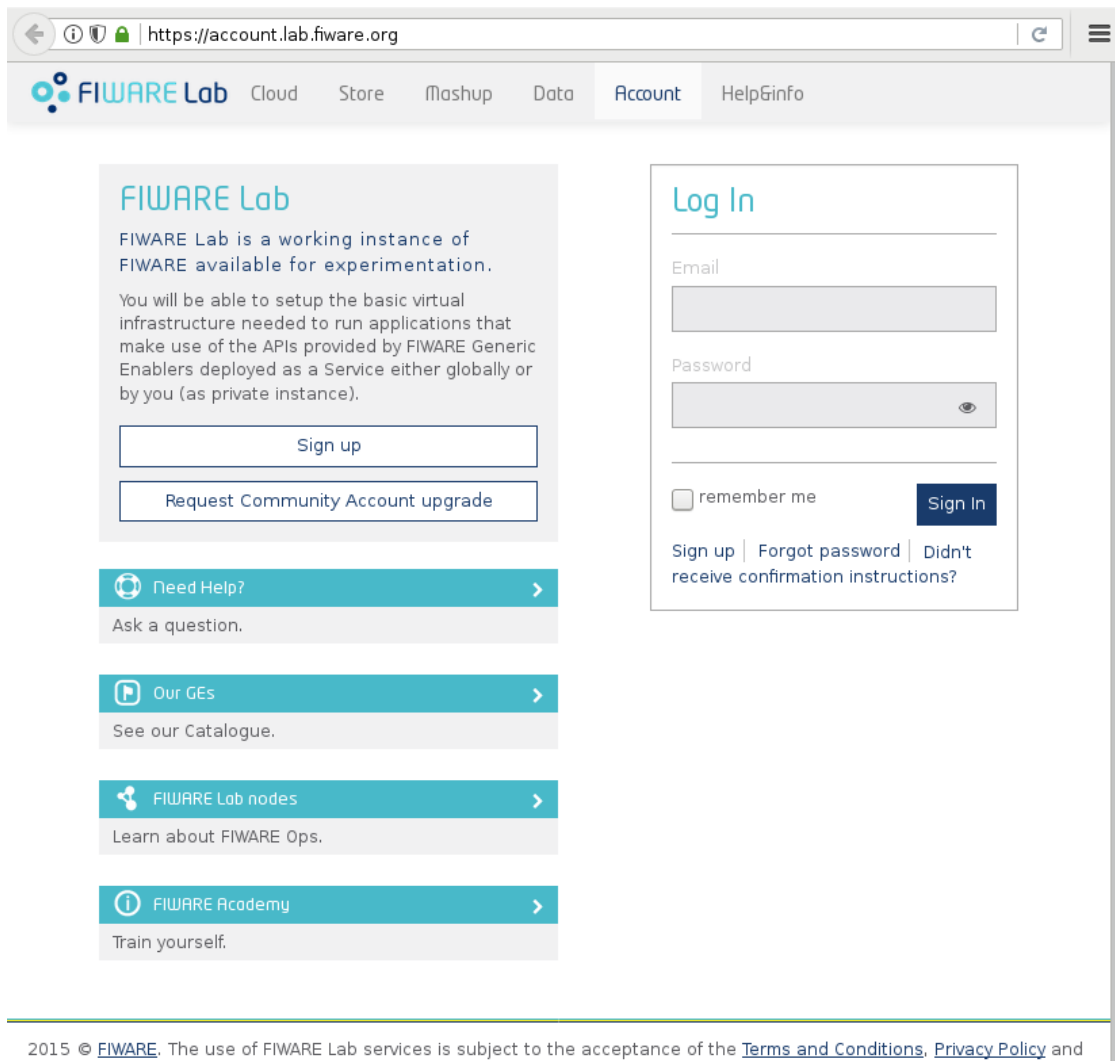
### 2.1 Introduction

This section details the usage of the Platform-Manager as available at <https://platform-manager.smartsdk.eu> see PLATFORM-MANAGER USAGE for a more generic description.

By the end of this chapter you will be able to create your Docker Swarm Cluster in FIWARE Lab and deploy some smartsdk-recipes on it.

### 2.2 Register in FIWARE Lab

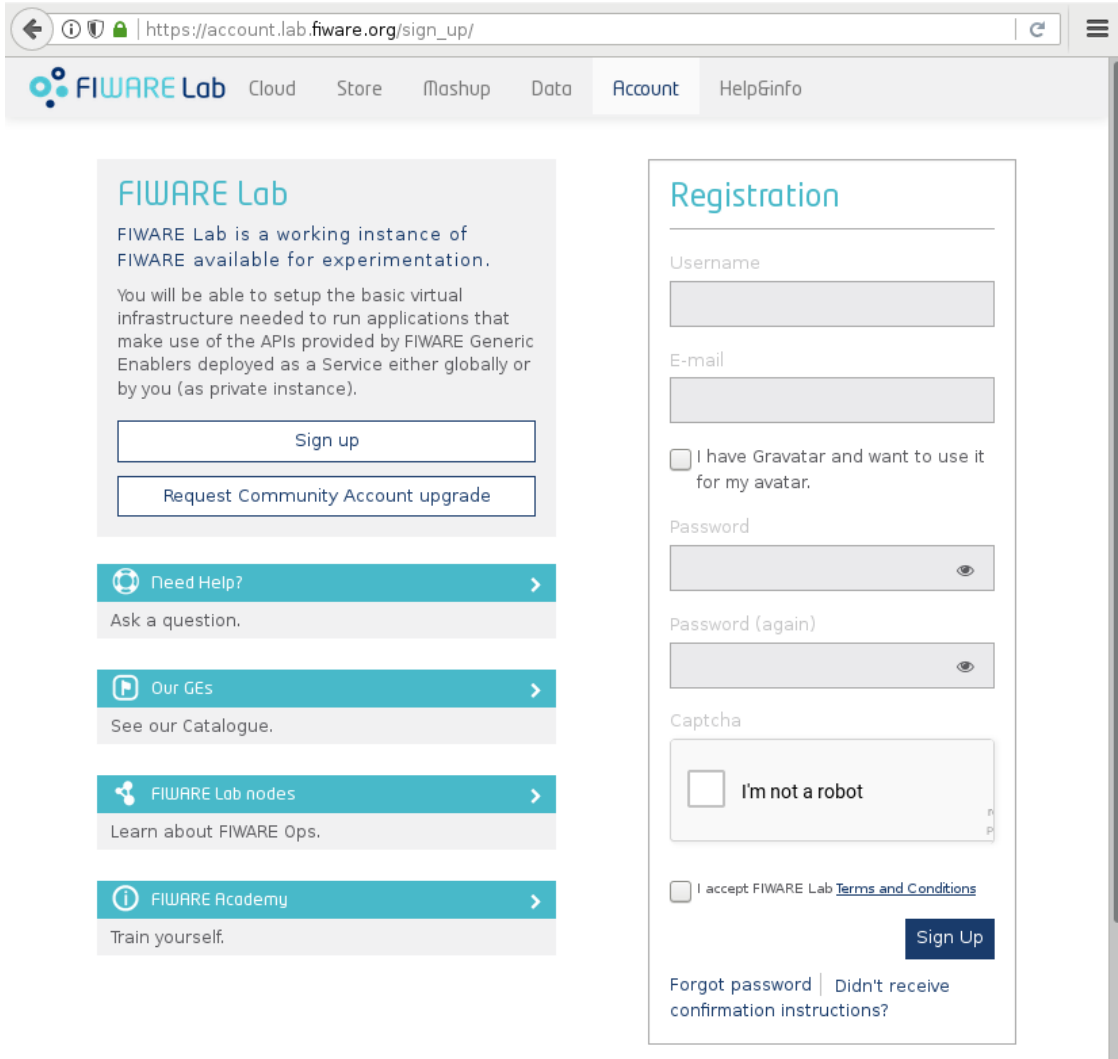
First of all, you need to register at the site <https://account.lab.fiware.org/>. The first time you have to click the “Sign up” button to be redirected to the Sign up form.



2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookie Policy](#).

*Figure 4: Home page account portal of FIWARE Lab*

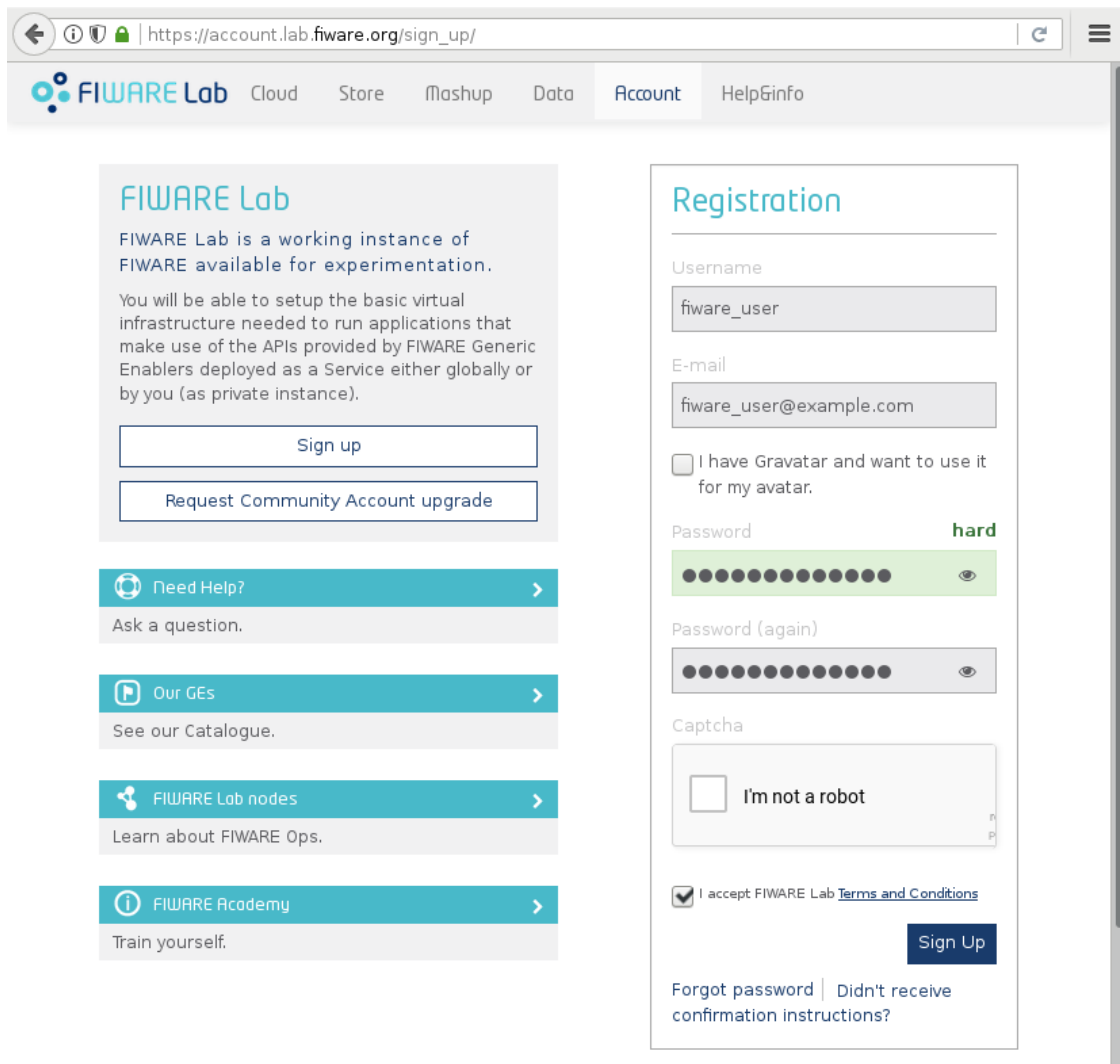
The first time you have to click the “Signup” button to be redirected to the Sign up form:



The screenshot shows the sign-up page for FIWARE Lab. The browser address bar displays [https://account.lab.fiware.org/sign\\_up/](https://account.lab.fiware.org/sign_up/). The navigation bar includes links for Cloud, Store, Mashup, Data, Account, and Help&Info. The main content area is divided into two sections. On the left, the 'FIWARE Lab' section explains that it is a working instance of FIWARE available for experimentation, allowing users to setup basic virtual infrastructure. It includes two buttons: 'Sign up' and 'Request Community Account upgrade'. Below this is a sidebar with four links: 'Need Help?' (Ask a question), 'Our GEs' (See our Catalogue), 'FIWARE Lab nodes' (Learn about FIWARE Ops), and 'FIWARE Academy' (Train yourself). On the right, the 'Registration' form is empty. It contains fields for Username, E-mail, Password, and Password (again), each with a toggle to show or hide the input. There is a checkbox for 'I have Gravatar and want to use it for my avatar.' and a Captcha section with a checkbox for 'I'm not a robot'. At the bottom of the form, there is a checkbox for 'I accept FIWARE Lab Terms and Conditions' and a 'Sign Up' button. Links for 'Forgot password' and 'Didn't receive confirmation instructions?' are also present.

*Figure 5: Empty form on Sign up of FIWARE Lab*

Complete the form with your personal data and agree with the FIWARE Lab Term and Conditions:



The screenshot shows the FIWARE Lab sign-up page. The browser address bar displays [https://account.lab.fiware.org/sign\\_up/](https://account.lab.fiware.org/sign_up/). The page header includes the FIWARE Lab logo and navigation links: Cloud, Store, Mashup, Data, Account, and Help&info.

**FIWARE Lab**  
FIWARE Lab is a working instance of FIWARE available for experimentation.  
You will be able to setup the basic virtual infrastructure needed to run applications that make use of the APIs provided by FIWARE Generic Enablers deployed as a Service either globally or by you (as private instance).

[Sign up](#)  
[Request Community Account upgrade](#)

**Need Help?** >  
Ask a question.

**Our GEs** >  
See our Catalogue.

**FIWARE Lab nodes** >  
Learn about FIWARE Ops.

**FIWARE Academy** >  
Train yourself.

**Registration**

Username

E-mail

☐ I have Gravatar and want to use it for my avatar.

Password **hard**

Password (again)

Captcha  
☐ I'm not a robot

☒ I accept FIWARE Lab [Terms and Conditions](#)

[Sign Up](#)

[Forgot password](#) | [Didn't receive confirmation instructions?](#)

*Figure 6: Example of form Completion on Sign up of FIWARE Lab*

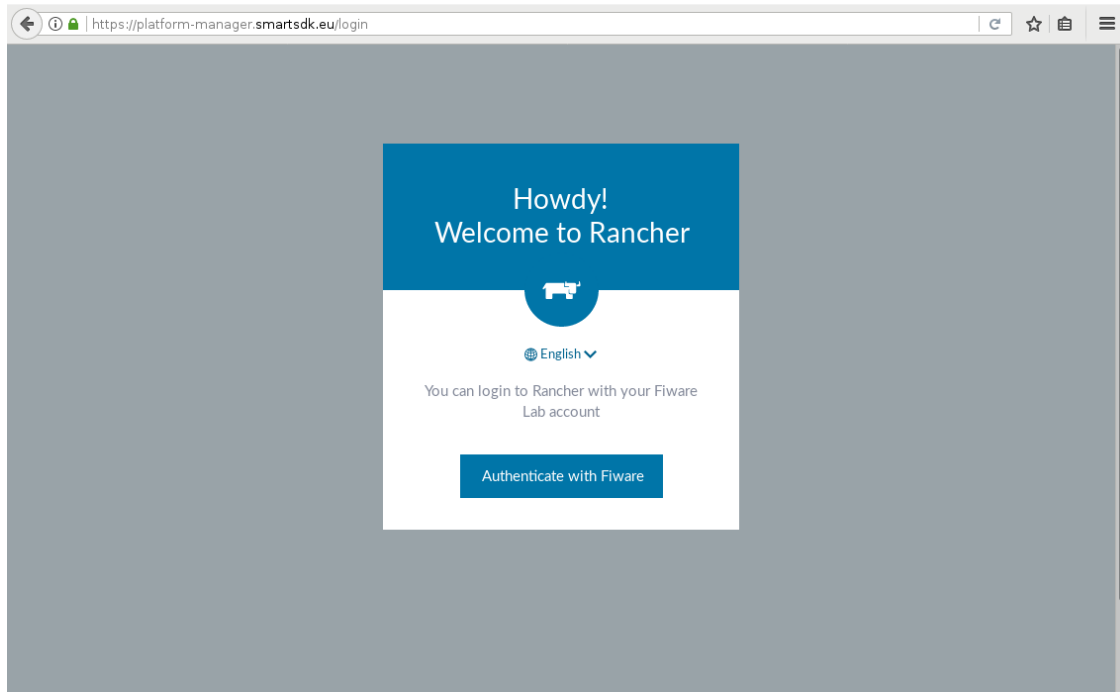
Complete the registration steps by following the instructions found in the registration email.

## 2.3 Configure your cluster

Go to the home page of the [SmartSDK Platform Manager](https://platform-manager.smartsdk.eu/)<sup>9</sup> and click on “Authenticate with Fiware”.

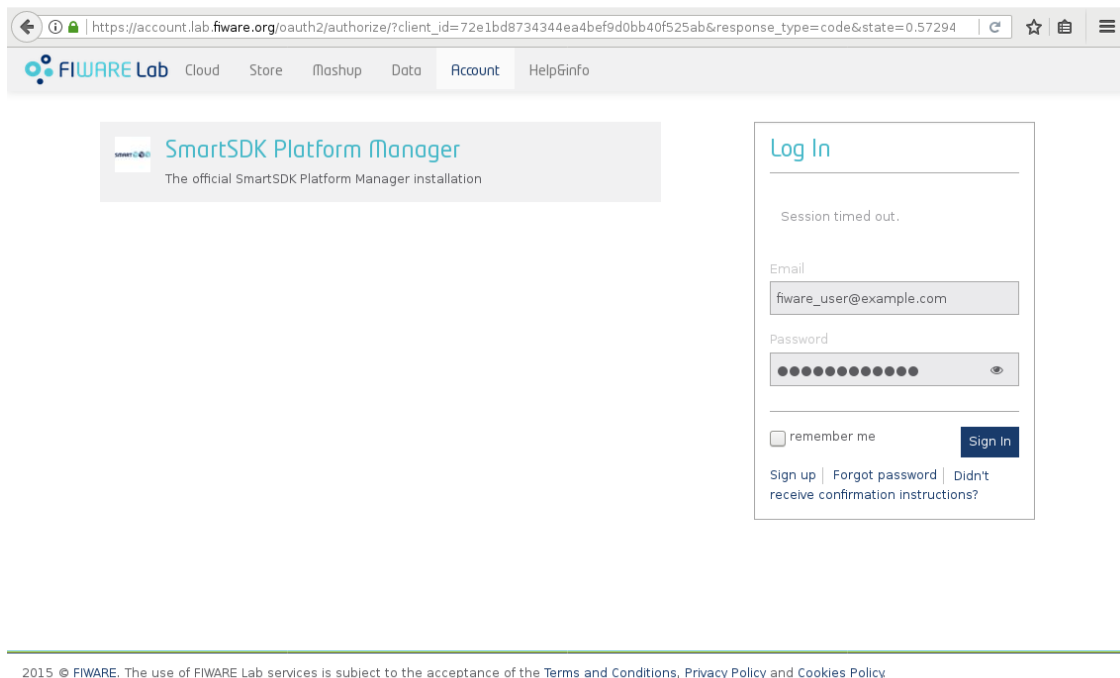
<sup>9</sup> <https://platform-manager.smartsdk.eu/>





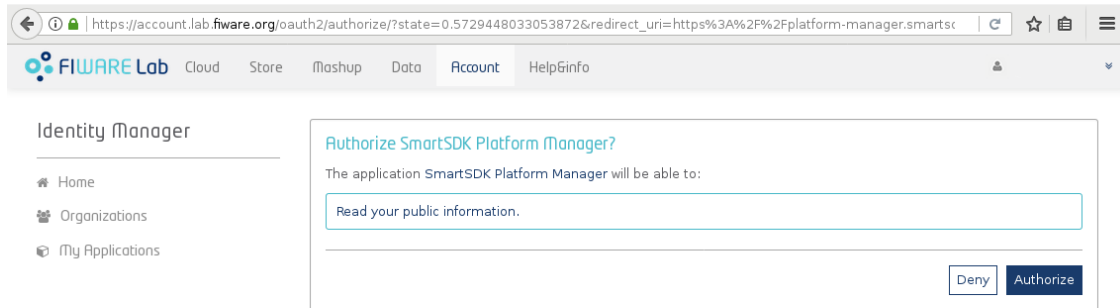
*Figure 7: Home page of SmartSDK Platform Manager*

You will be redirected to the Fiware Lab login page. Insert your credentials.



*Figure 8: Fiware Lab Login Page*

Once you login, you need to authorize the SmartSDK Platform to access your public information in order to create and enable your account.



2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

*Figure 9: Fiware Lab Authorization Request*

Then you will be redirected to the SmartSDK Platform as an authorized user.

## 2.4 Setup Swarm on Fiware Lab

In the SmartSDK platform, depending on what is enabled by the administrator, you can create your own environment.

Once an environment is created, you can add new hosts to the environment.

Once an host is added you can deploy your application on it.

Here we document the creation of a “Docker Swarm” environment, with hosts running on the FIWARE Lab.

First, in the “Environment” tab select the “Manage Environments”.

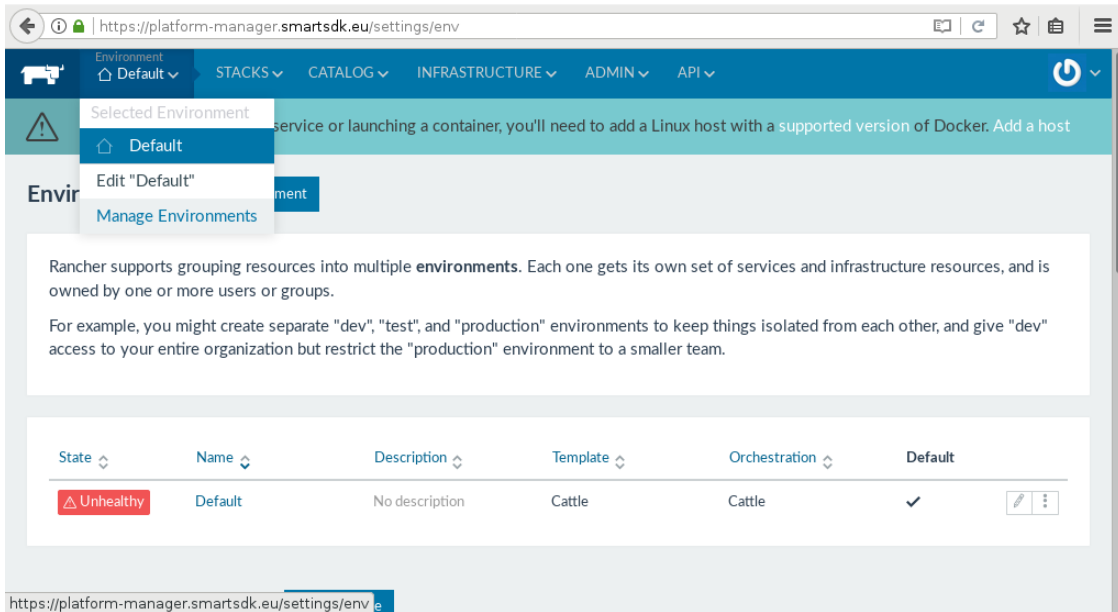


Figure 10: Manage Environments Menu

Then click the “Add Environment” button.

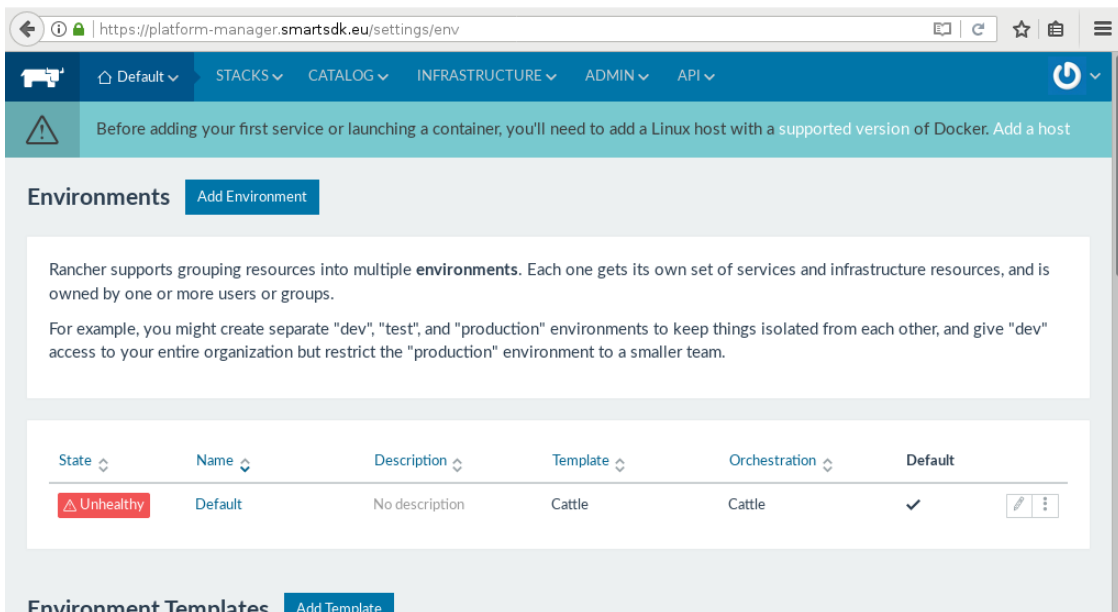
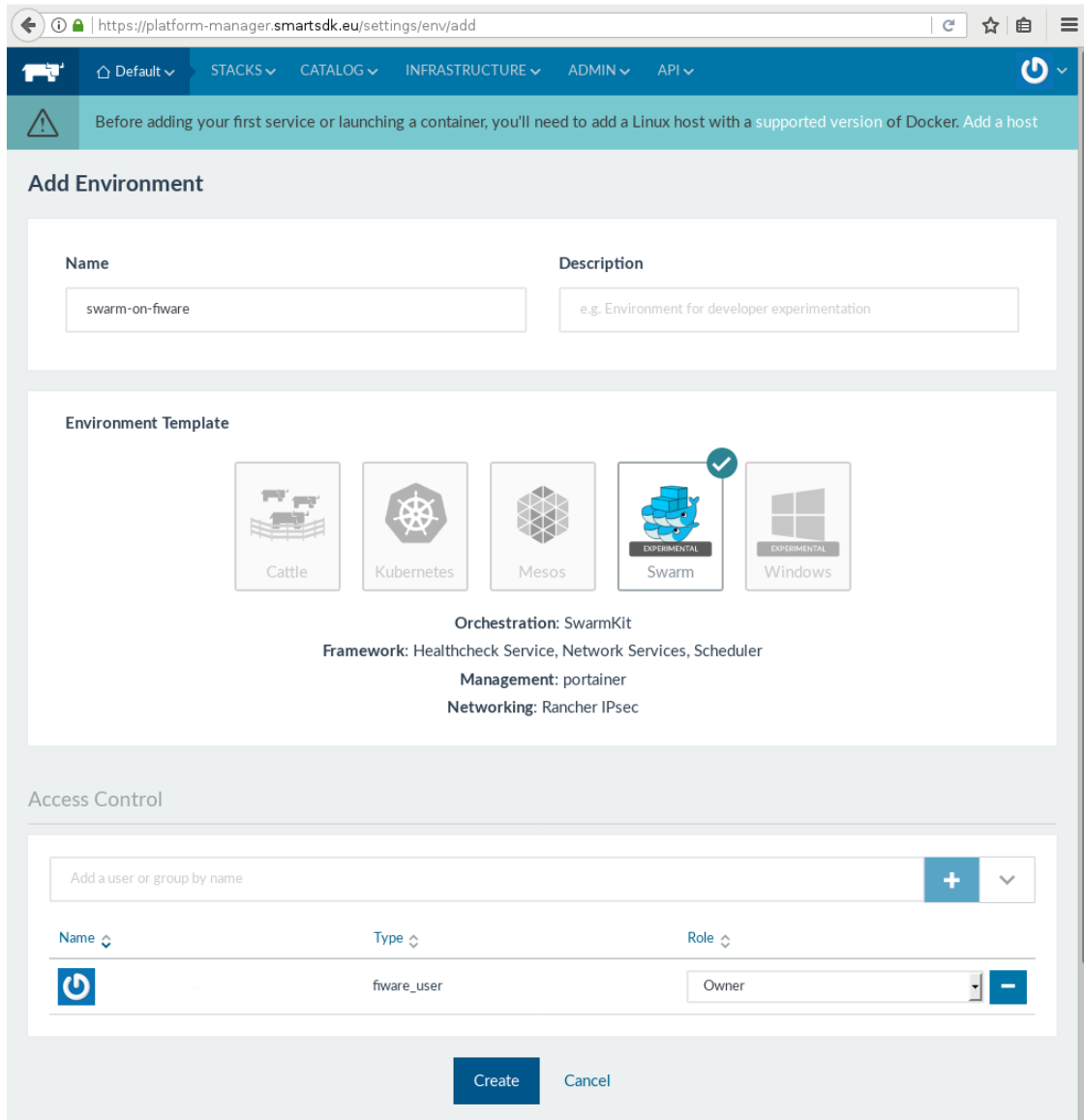


Figure 11: Add Environment

Fill the **Name** and the optional **Description** and ensure the Environment Template is set to **Fiware Swarm**.



https://platform-manager.smartsdk.eu/settings/env/add

Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Add Environment

**Name**  
swarm-on-fiware

**Description**  
e.g. Environment for developer experimentation


**Environment Template**

Cattle Kubernetes Mesos **Swarm** Windows

**Orchestration:** SwarmKit  
**Framework:** Healthcheck Service, Network Services, Scheduler  
**Management:** portainer  
**Networking:** Rancher IPsec

**Access Control**

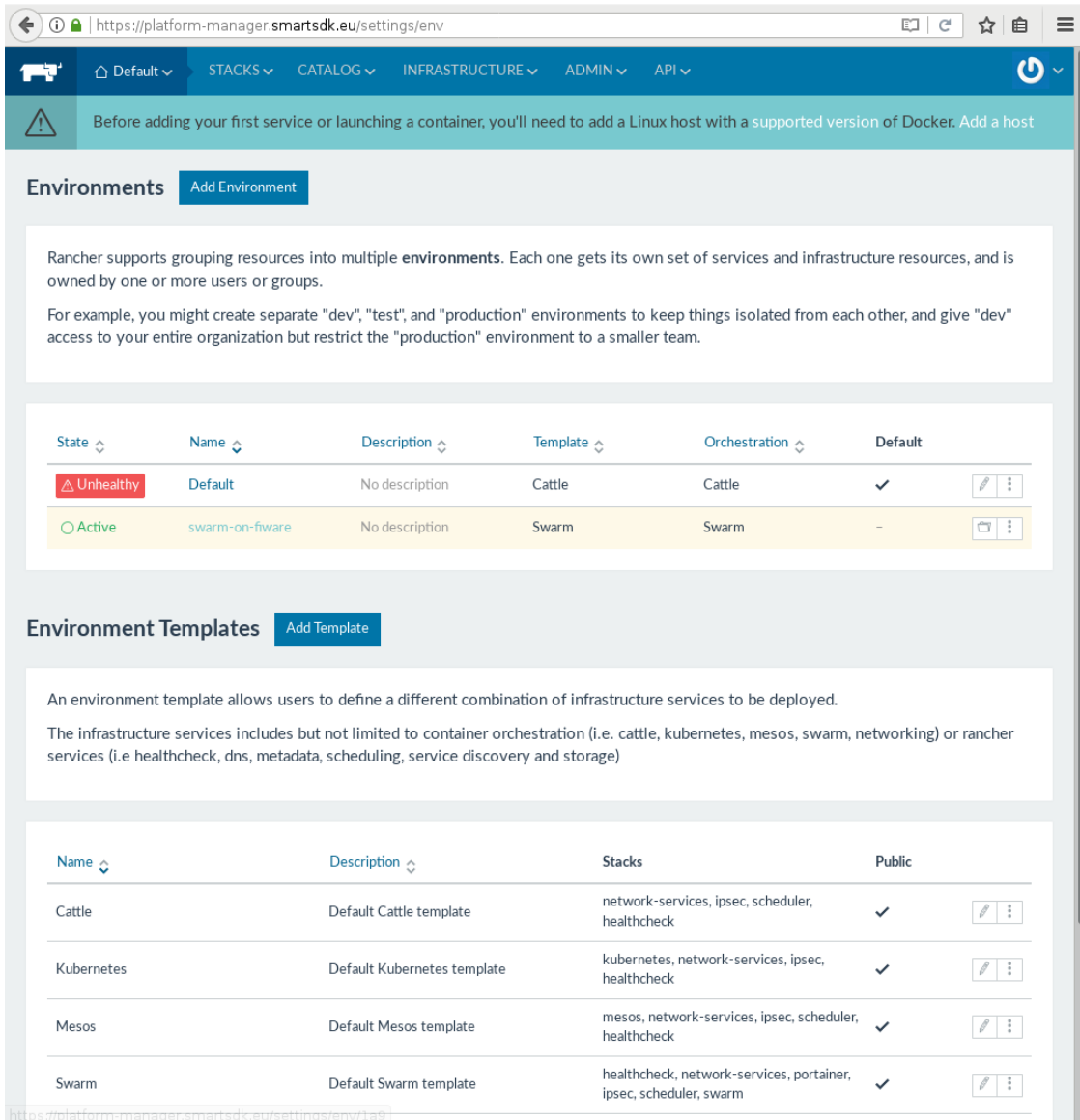
Add a user or group by name

Name	Type	Role
	fiware_user	Owner

Create Cancel

*Figure 12: Select Fiware Swarm as the Environment Template*

You will be redirected to the environments list. Select the newly created environment.



Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Environments

Add Environment

Rancher supports grouping resources into multiple **environments**. Each one gets its own set of services and infrastructure resources, and is owned by one or more users or groups.

For example, you might create separate "dev", "test", and "production" environments to keep things isolated from each other, and give "dev" access to your entire organization but restrict the "production" environment to a smaller team.

State	Name	Description	Template	Orchestration	Default
Unhealthy	Default	No description	Cattle	Cattle	✓
Active	swarm-on-fware	No description	Swarm	Swarm	-

### Environment Templates

Add Template

An environment template allows users to define a different combination of infrastructure services to be deployed.

The infrastructure services includes but not limited to container orchestration (i.e. cattle, kubernetes, mesos, swarm, networking) or rancher services (i.e healthcheck, dns, metadata, scheduling, service discovery and storage)

Name	Description	Stacks	Public
Cattle	Default Cattle template	network-services, ipsec, scheduler, healthcheck	✓
Kubernetes	Default Kubernetes template	kubernetes, network-services, ipsec, healthcheck	✓
Mesos	Default Mesos template	mesos, network-services, ipsec, scheduler, healthcheck	✓
Swarm	Default Swarm template	healthcheck, network-services, portainer, ipsec, scheduler, swarm	✓

Figure 13: Select newly created environment

In the new environment you will see the list of the users. A warning at the top of the page will invite you to click on the “Add a host” link. Click the link and continue reading.

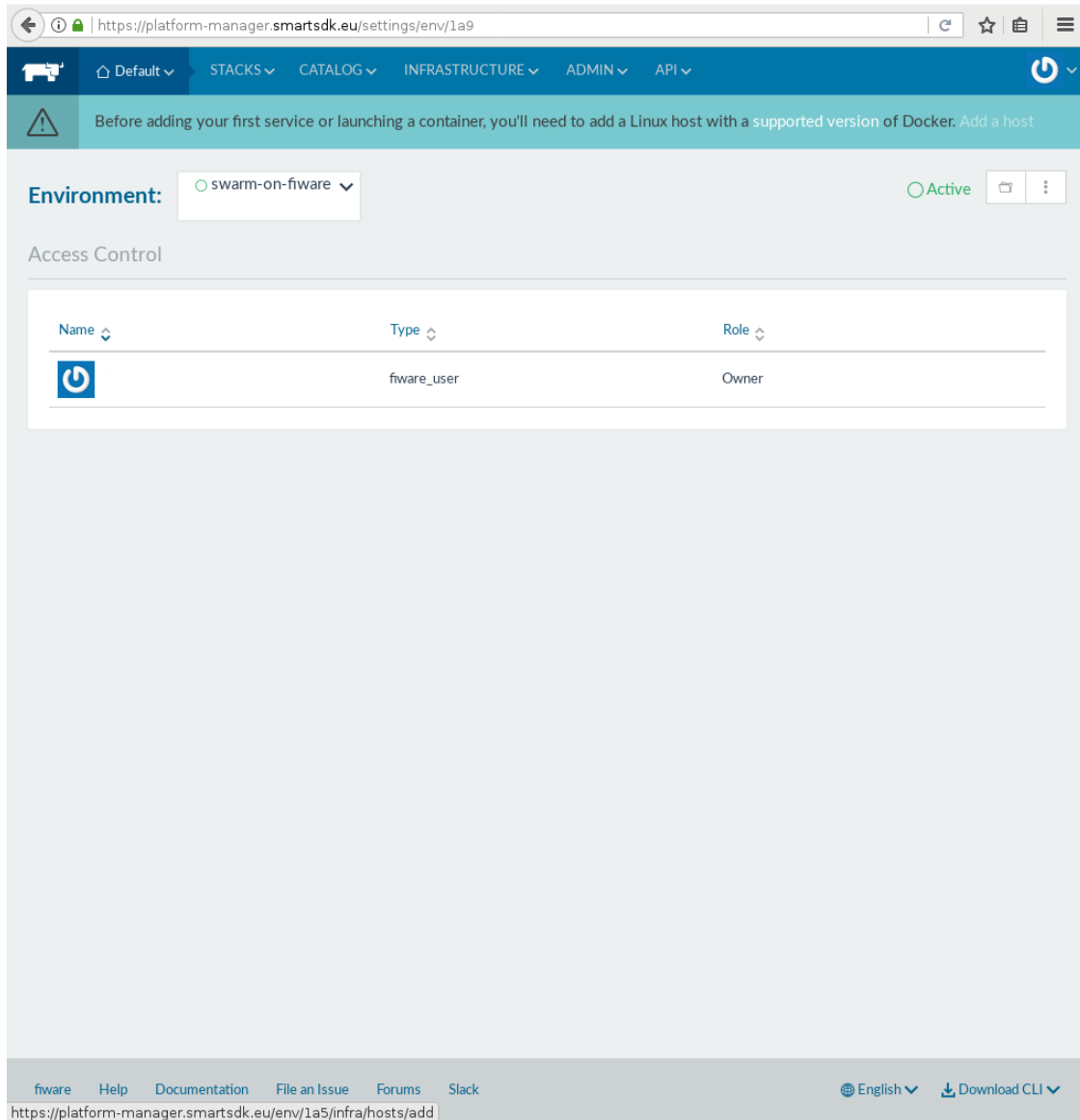


Figure 14: Add host warning

## 2.5 Deploy your cluster

In the “Add Host” procedure we can leverage the FIWARE Lab Rancher UI driver in order to automatically create hosts on the FIWARE Lab.

In the initial page select the “FIWARE Lab” driver.

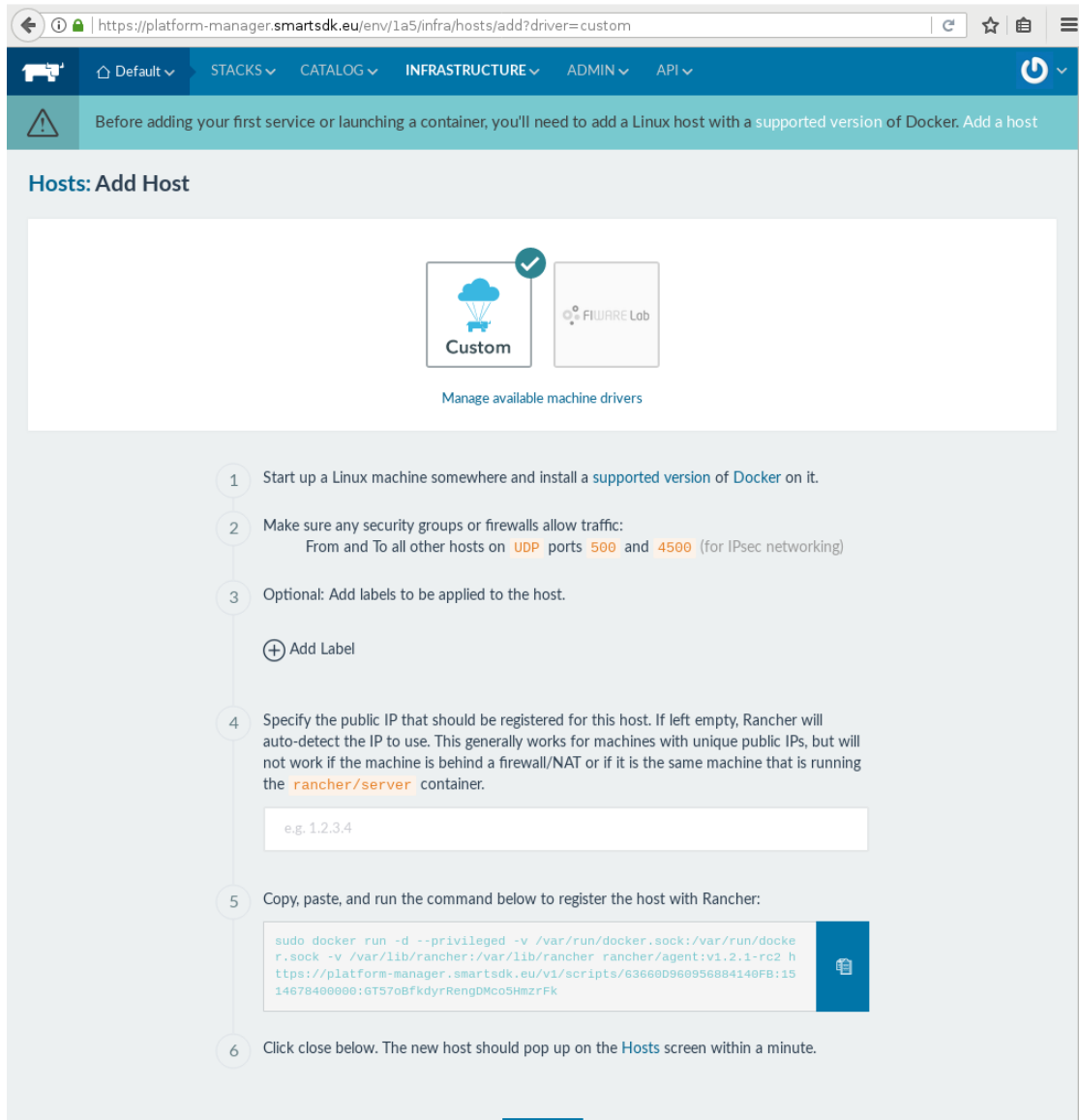
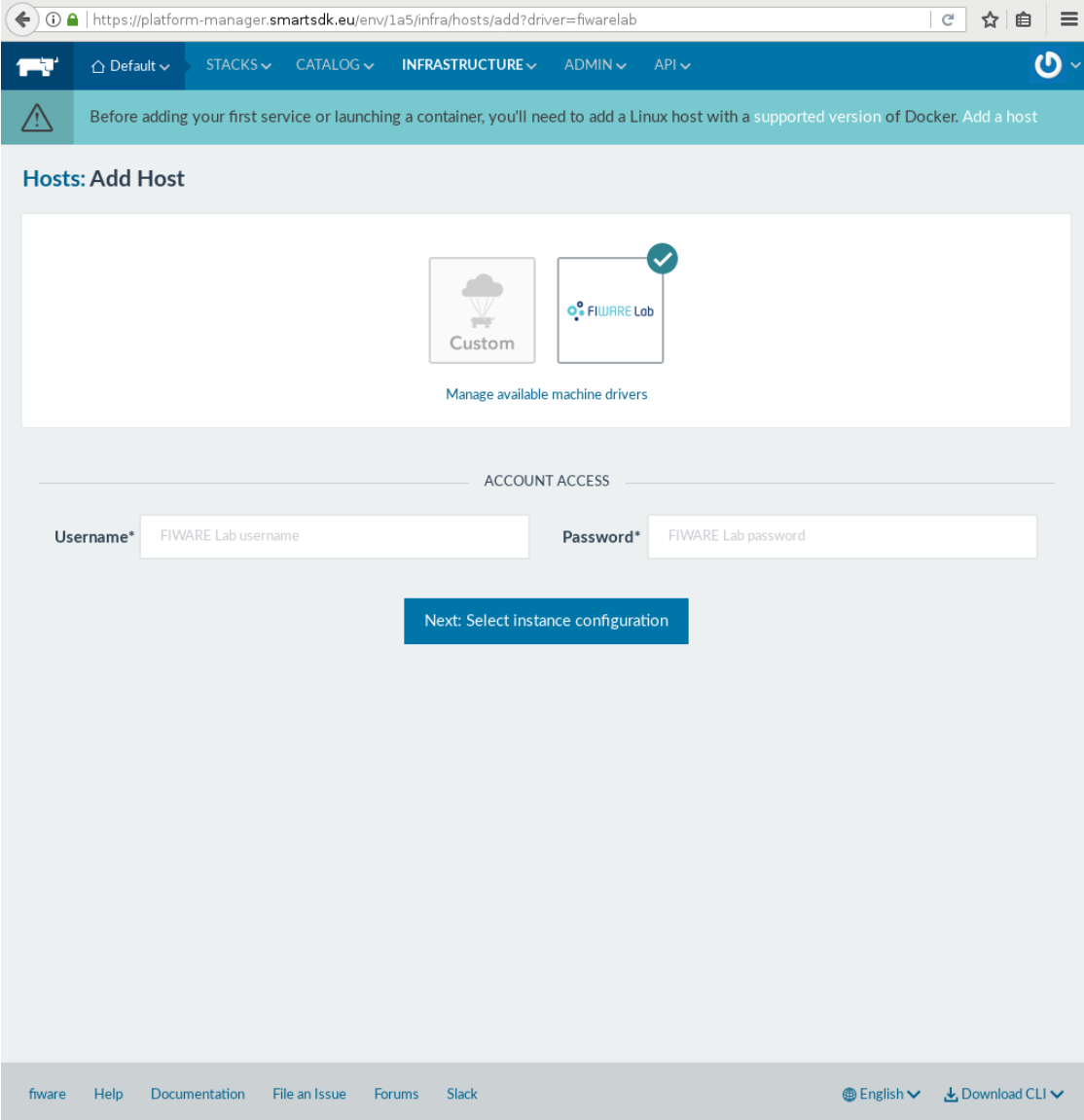


Figure 15: Add new hosts driver selection

Then insert your FIWARE Cloud Lab credentials. Please note that those credentials are usually different from the ones used for the OAuth2 procedure. Those credentials are the ones used for the OpenStack authentication and are the same you would use on the [cloud lab](https://cloud.lab.fiware.org)<sup>10</sup>.

<sup>10</sup> <https://cloud.lab.fiware.org>




https://platform-manager.smartsdk.eu/env/1a5/infra/hosts/add?driver=fiwarelab


Default STACKS CATALOG INFRASTRUCTURE ADMIN API

Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Hosts: Add Host



Custom



FIWARE Lab

Manage available machine drivers

ACCOUNT ACCESS

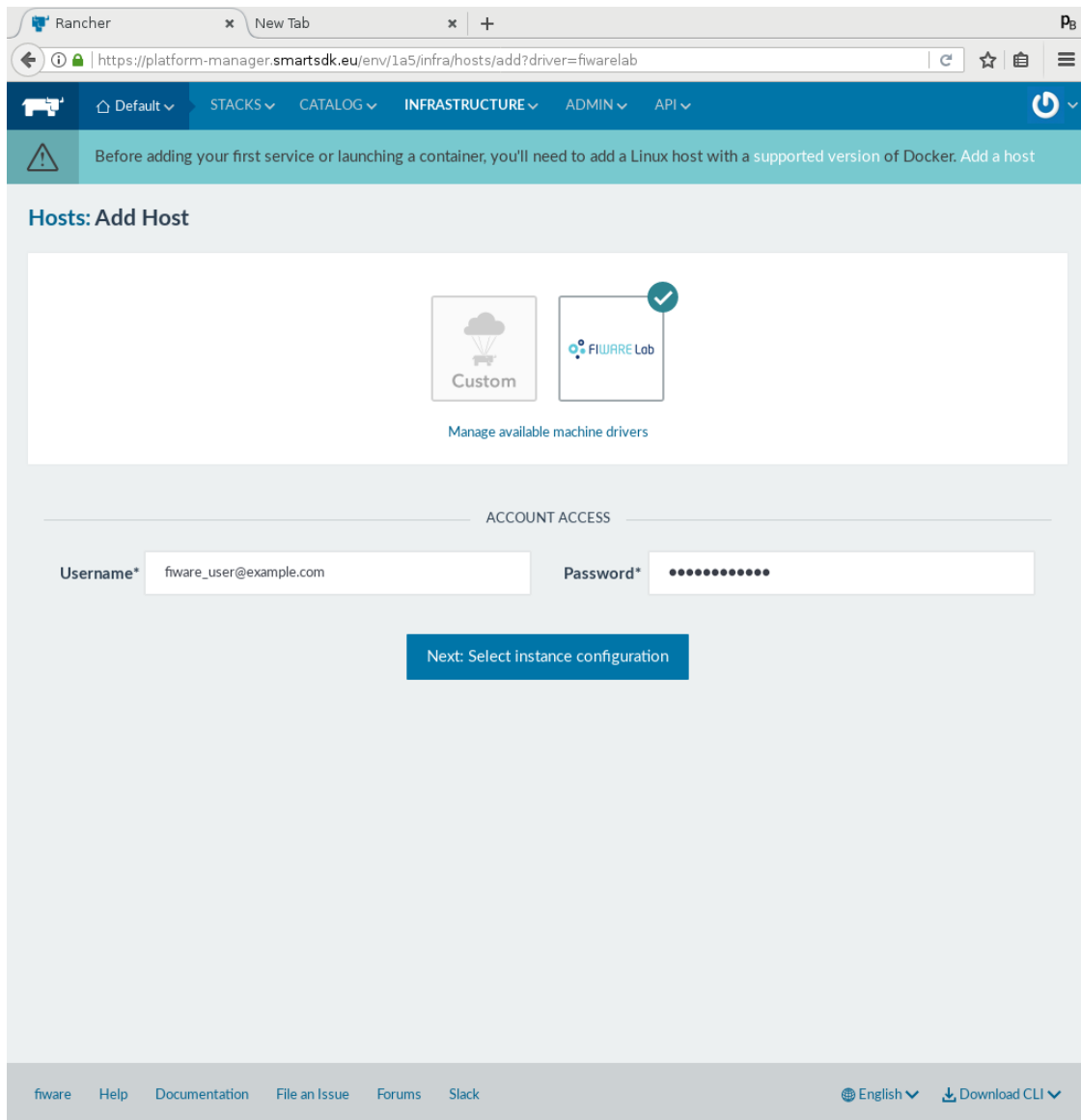
Username\* FIWARE Lab username Password\* FIWARE Lab password

Next: Select instance configuration

fiware Help Documentation File an Issue Forums Slack English Download CLI

Figure 16: Insert FIWARE credentials





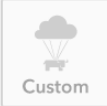
Rancher New Tab + P8

https://platform-manager.smartsdk.eu/env/1a5/infra/hosts/add?driver=fiwarelab


Default STACKS CATALOG INFRASTRUCTURE ADMIN API

Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Hosts: Add Host



Custom



FIWARE Lab

Manage available machine drivers

ACCOUNT ACCESS

Username\* fiware\_user@example.com Password\* .....

Next: Select instance configuration

fiware Help Documentation File an Issue Forums Slack English Download CLI

*Figure 17: Select host configuration*

If you have more than one region enabled, you can choose where to create new hosts.

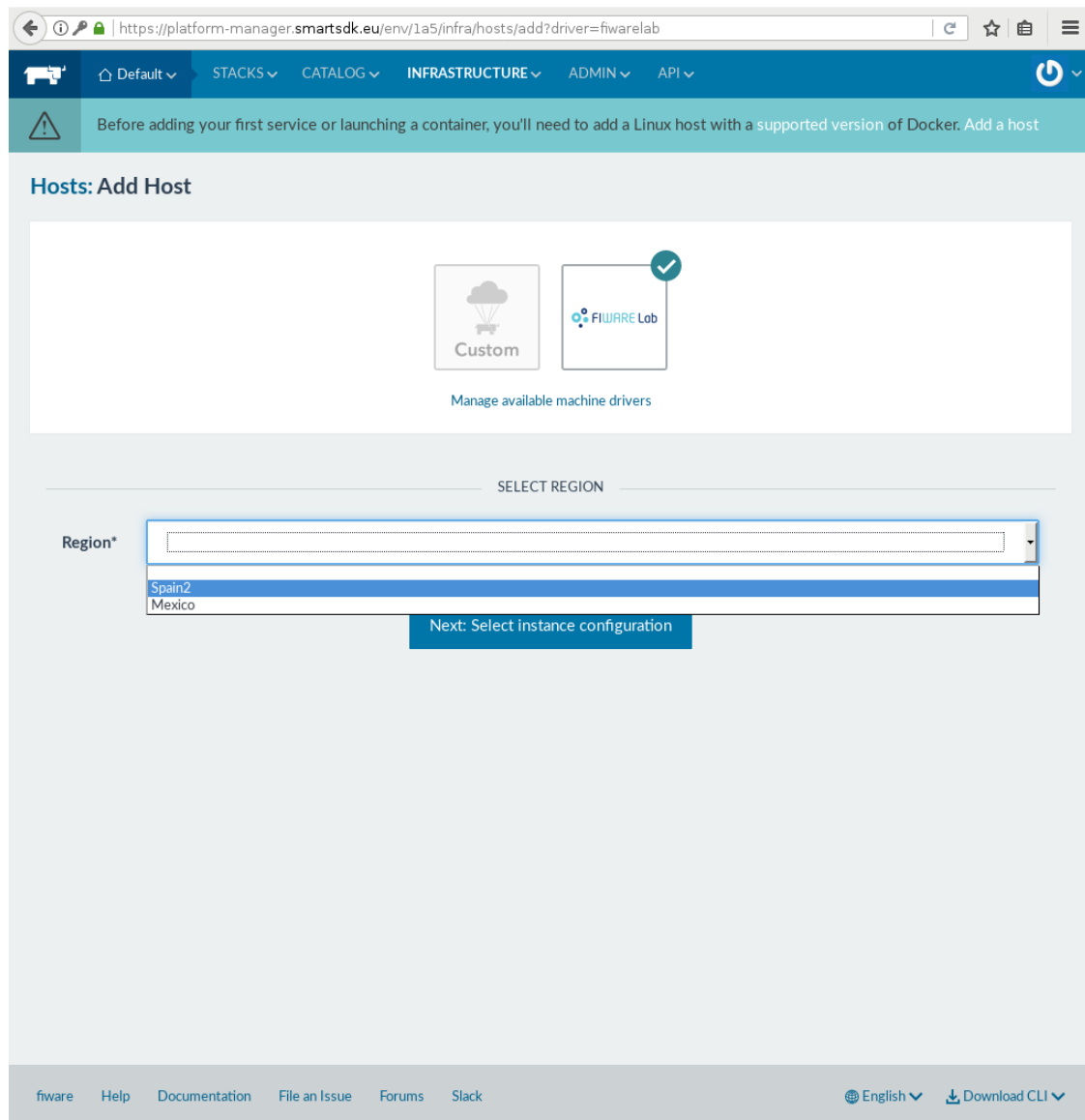
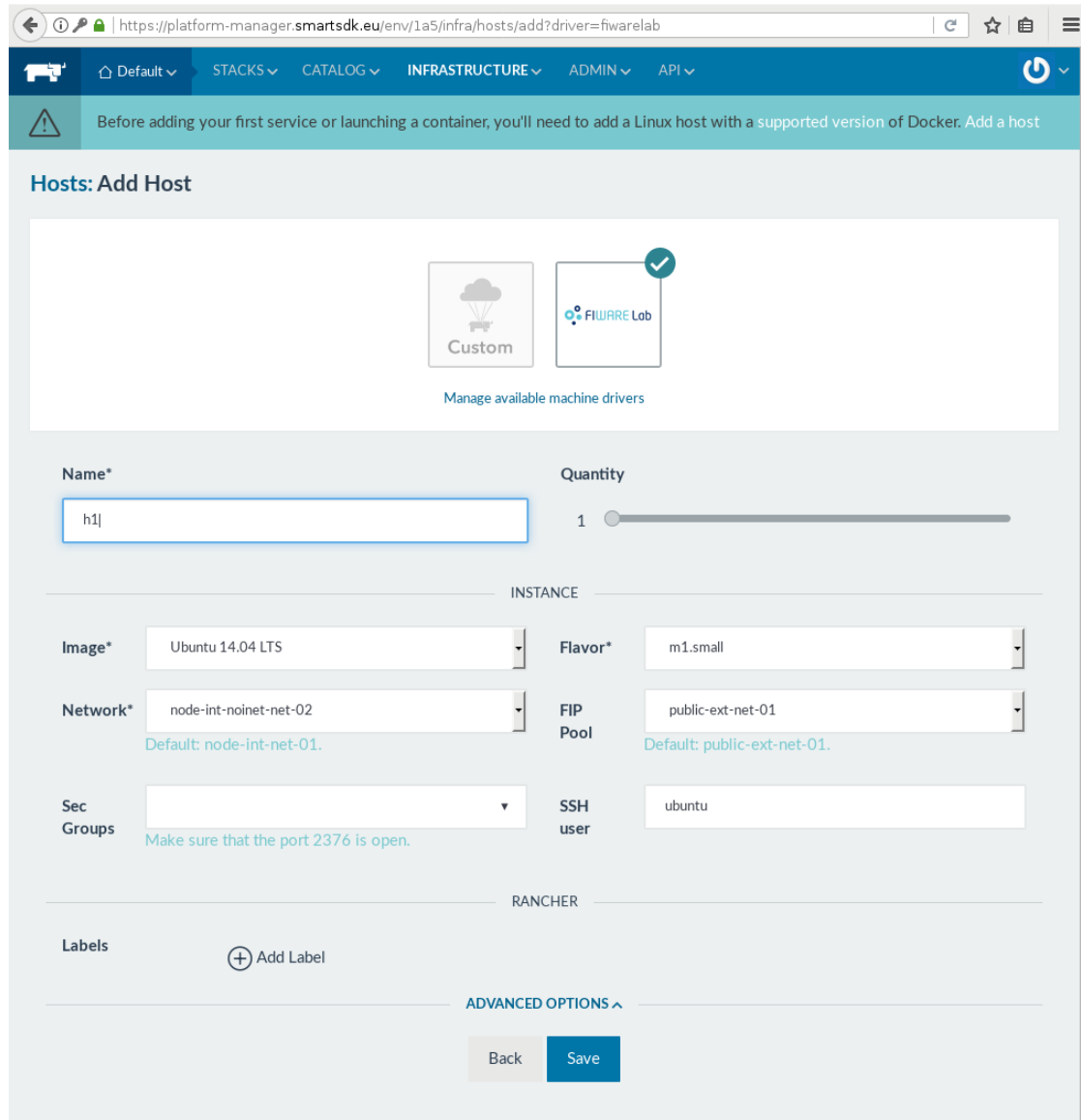


Figure 18: Region selection

Then you need to provide some information regarding the host configuration you want to deploy.



Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Hosts: Add Host

Manage available machine drivers

**Name\***  **Quantity**

**INSTANCE**

**Image\***  **Flavor\***

**Network\***  **FIP Pool**   
Default: node-int-net-01. Default: public-ext-net-01.

**Sec Groups**  **SSH user**   
Make sure that the port 2376 is open.

**RANCHER**

**Labels**

**ADVANCED OPTIONS ^**

Figure 19: Add hosts configuration details

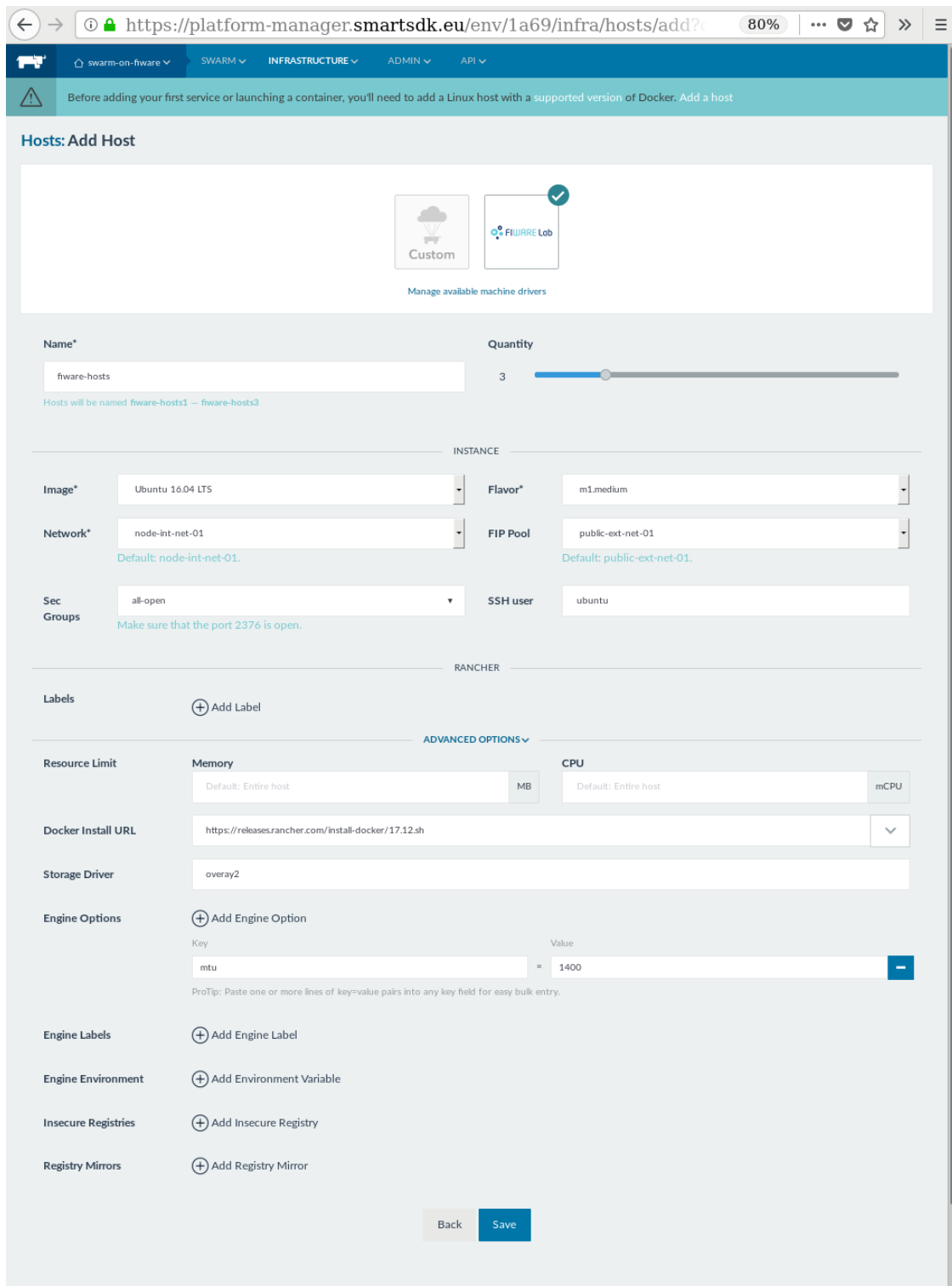
The supported configuration on the default instance of the platform manager with host running on the FIWARE Lab requires the following settings:

- ➔ Image: **Ubuntu 16.04 LTS**
- ➔ Flavor: **m1.medium**
- ➔ Security Groups: **Ports 22/TCP and 2376:2378/TCP Open**
- ➔ Storage Engine: **overlay2**
- ➔ Docker Install Url:
  - for canonical install: **<https://releases.rancher.com/install-docker/17.12.sh>**
  - for custom install supporting “smartsdk-recipes”: **<https://platform-manager-legacy.smartsdk.eu/install-docker/17.12-smartsdk.sh>**
- ➔ Docker Engine Options: key: **mtu**, value **1400**
- ➔ Network: **node-int-net-01**

➔ FIP Pool: **federation-ext-net-01**

**Note:** for the “Security Groups” a suitable group with the correct ports open must be already created in your OpenStack Project.

**Note:** if your OpenStack installation uses a lower MTU than the de-facto standard of 1500 bytes, you need to configure the Docker Engine Option properly. For a detailed discussion on MTU see Rancher IPsec plugin MTU (Fiware LAB).



Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Hosts: Add Host

Manage available machine drivers

**Name\***  **Quantity**

Hosts will be named fware-hosts1 – fware-hosts3

**INSTANCE**

**Image\***  **Flavor\***

**Network\***  **FIP Pool**

Default: node-int-net-01. Default: public-ext-net-01.

**Sec Groups**  **SSH user**

Make sure that the port 2376 is open.

**RANCHER**

**Labels** [+ Add Label](#)

**ADVANCED OPTIONS**

**Resource Limit**

**Memory**  **MB** **CPU**  **mCPU**

**Docker Install URL**

**Storage Driver**

**Engine Options** [+ Add Engine Option](#)

**Key**  **=** **Value**

ProTip: Paste one or more lines of key=value pairs into any key field for easy bulk entry.

**Engine Labels** [+ Add Engine Label](#)

**Engine Environment** [+ Add Environment Variable](#)

**Insecure Registries** [+ Add Insecure Registry](#)

**Registry Mirrors** [+ Add Registry Mirror](#)

[Back](#) [Save](#)

*Figure 20: Save hosts configuration*

Now you can go to the end of the page and click the “Save” button.

For a few minutes you will see a waiting page. In the background, the driver is starting and provisioning the newly created hosts.

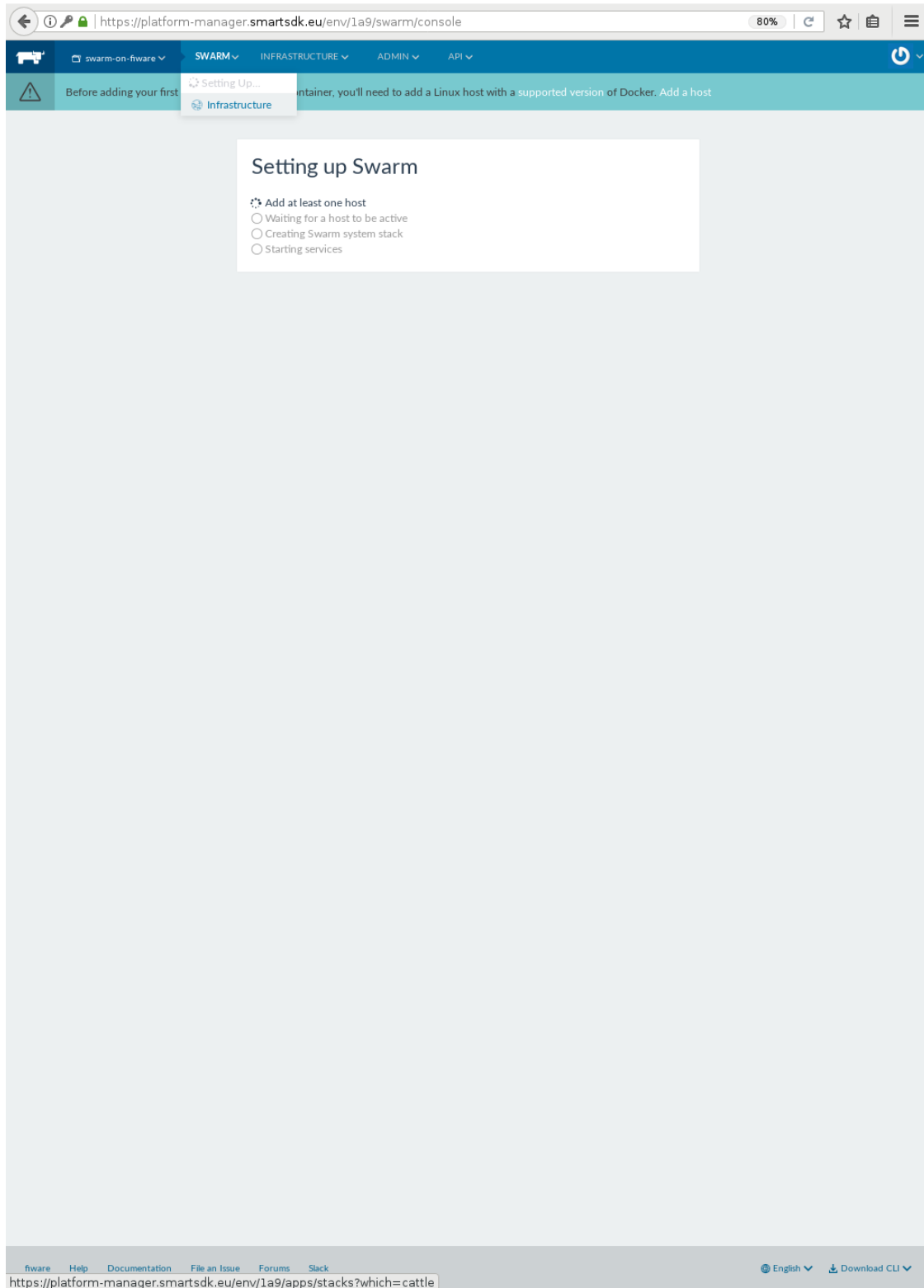


Figure 21: Wait for hosts

## 2.6 Deploy a stack using the web interface

After waiting for a while (usually a couple of minutes) your host should be in the “active” state.

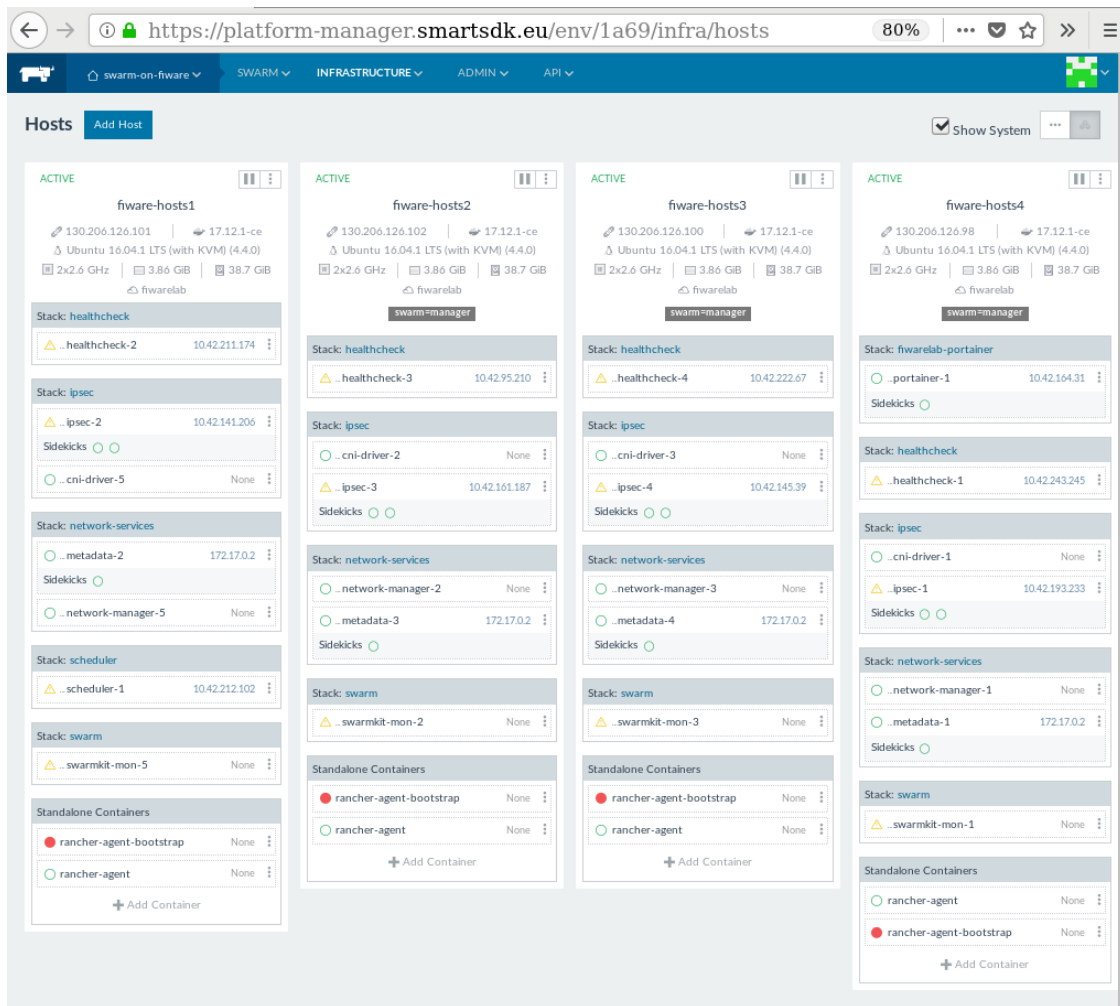


Figure 22: Host added

Following the menu “Swarm - Portainer” menu you can start our customized portainer web interface.

First be sure that in the settings the correct templates are loaded from the url:

<https://raw.githubusercontent.com/smartsdk/smartsdk-recipes/master/portainer/templates.json>.

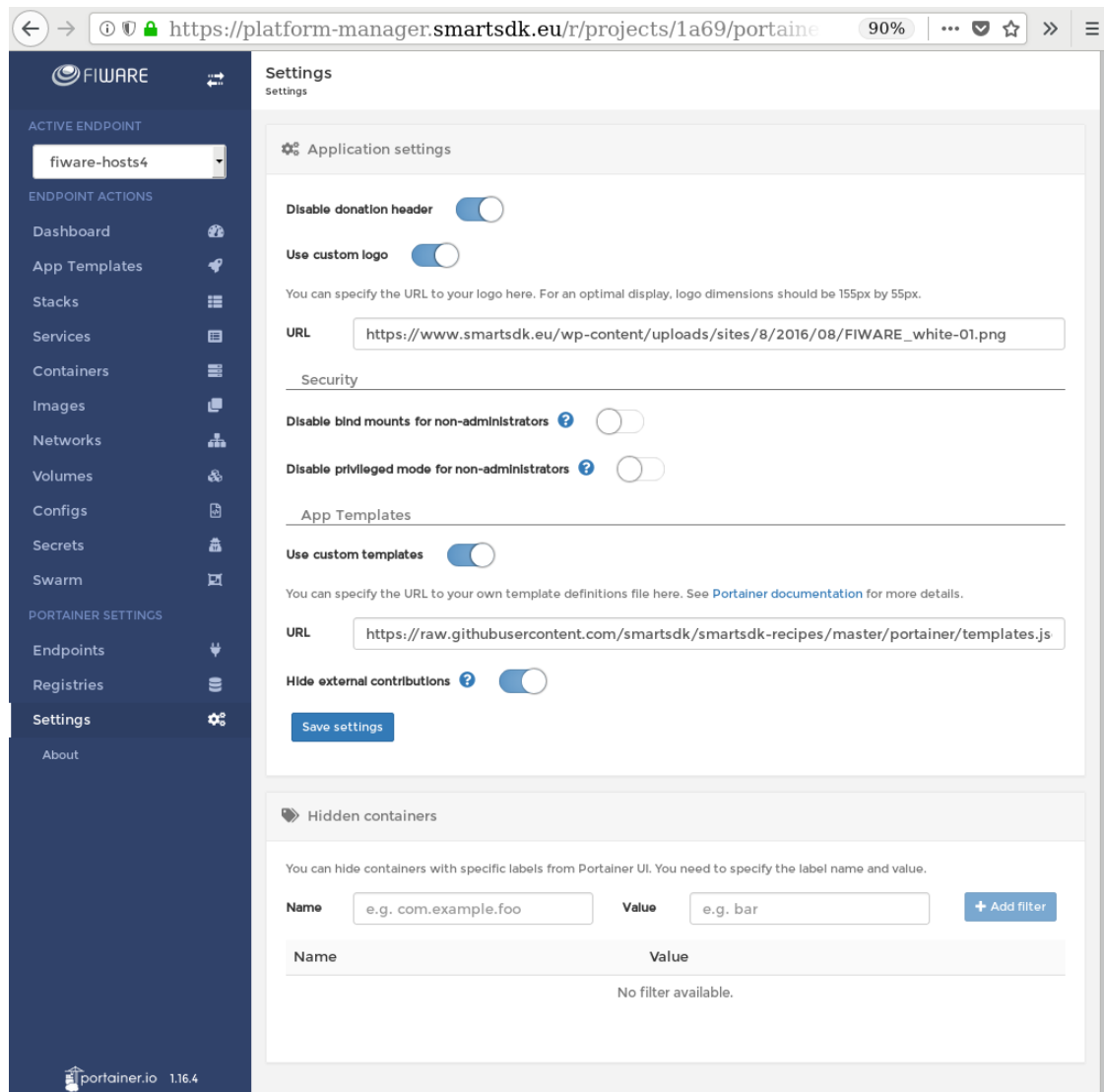


Figure 23: Portainer Template Settings

Usually for SmartSDK recipes the required networks **frontend** and **backend** have to be created as shown in the following screenshot. Please add these Networks with the option **com.docker.network.driver.mtu** and value of **1400**. Also, prefer using the **overlay** driver.



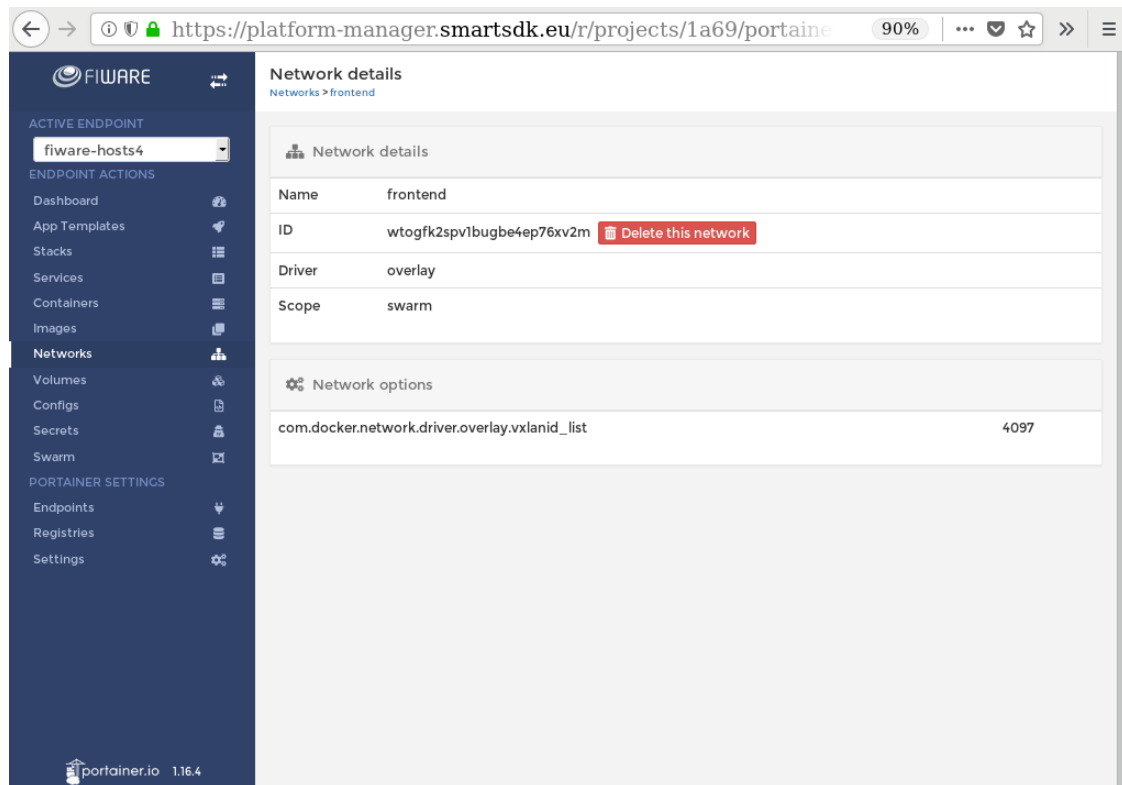


Figure 24: Network Creation

To deploy a stack from our templates, follow the “App Templates” link.

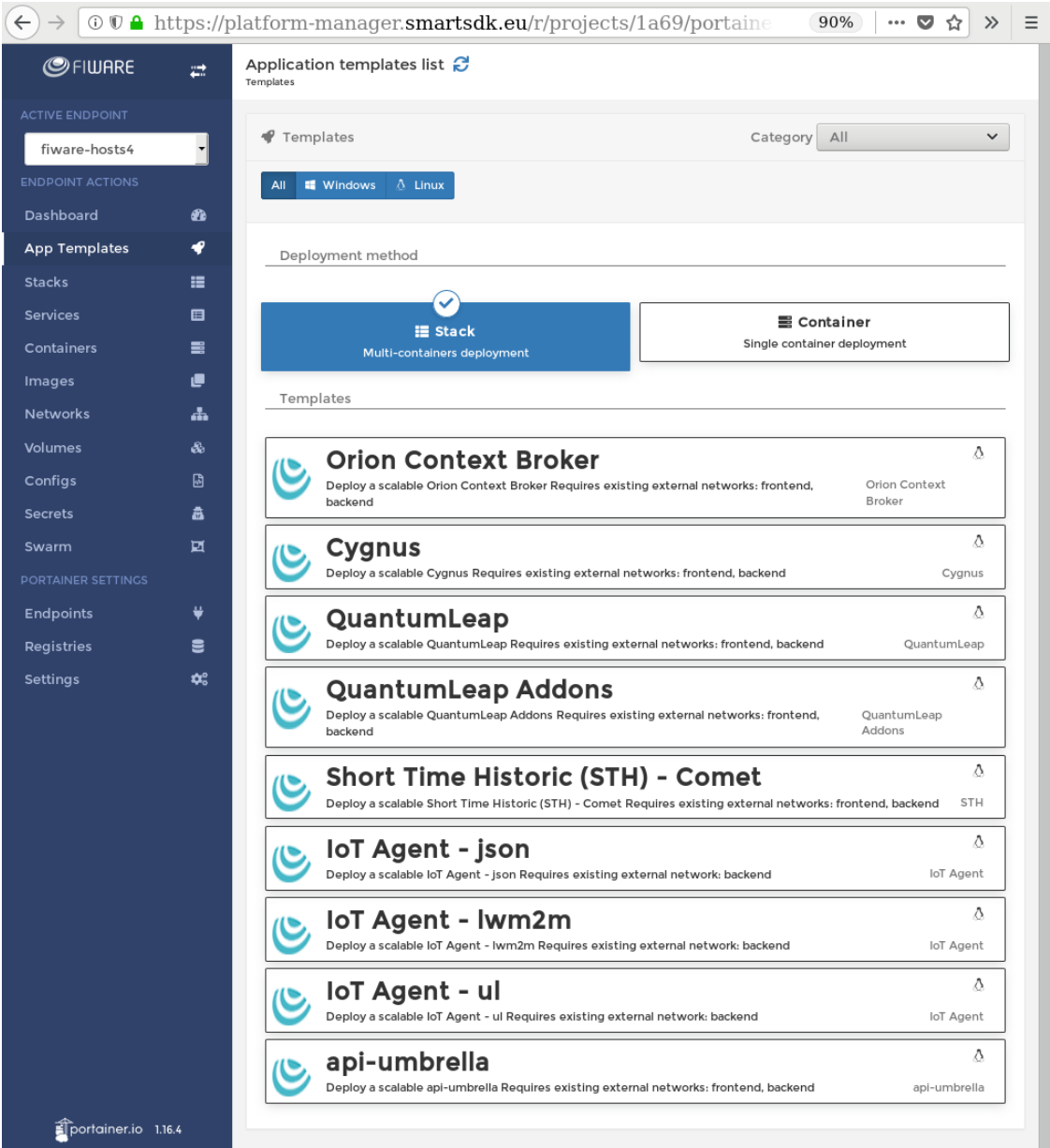
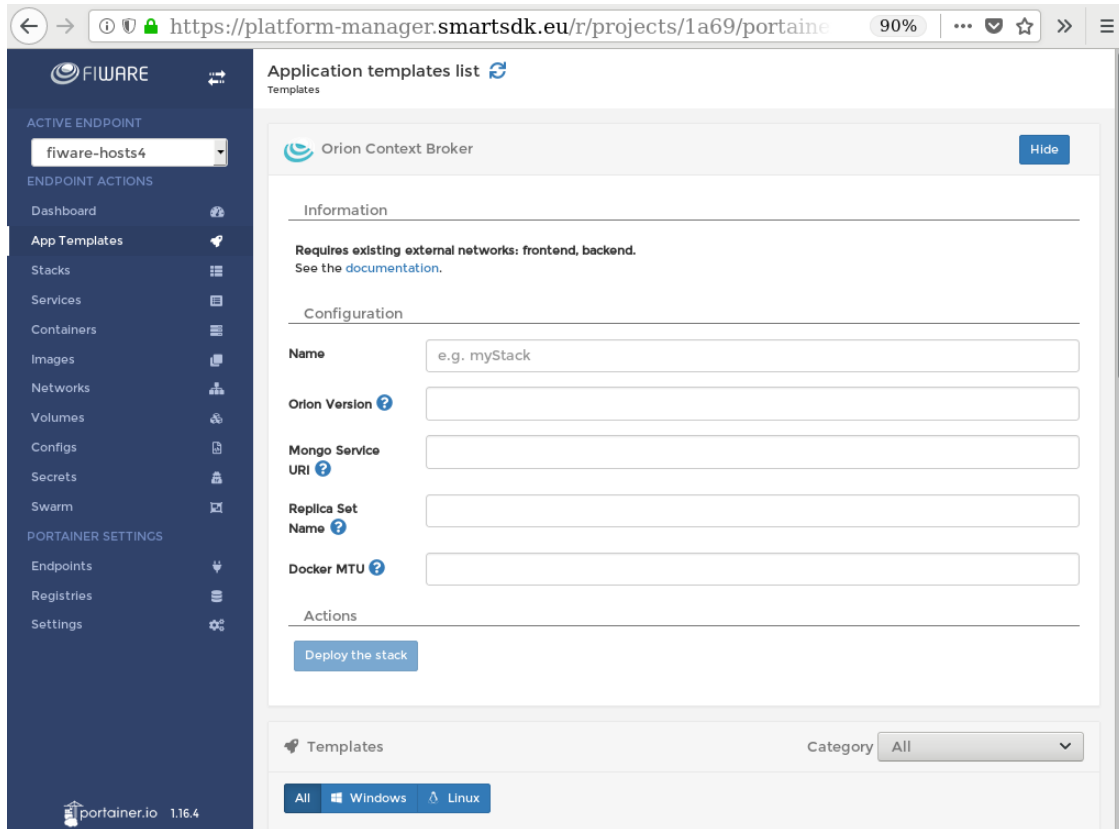


Figure 25: Application listing

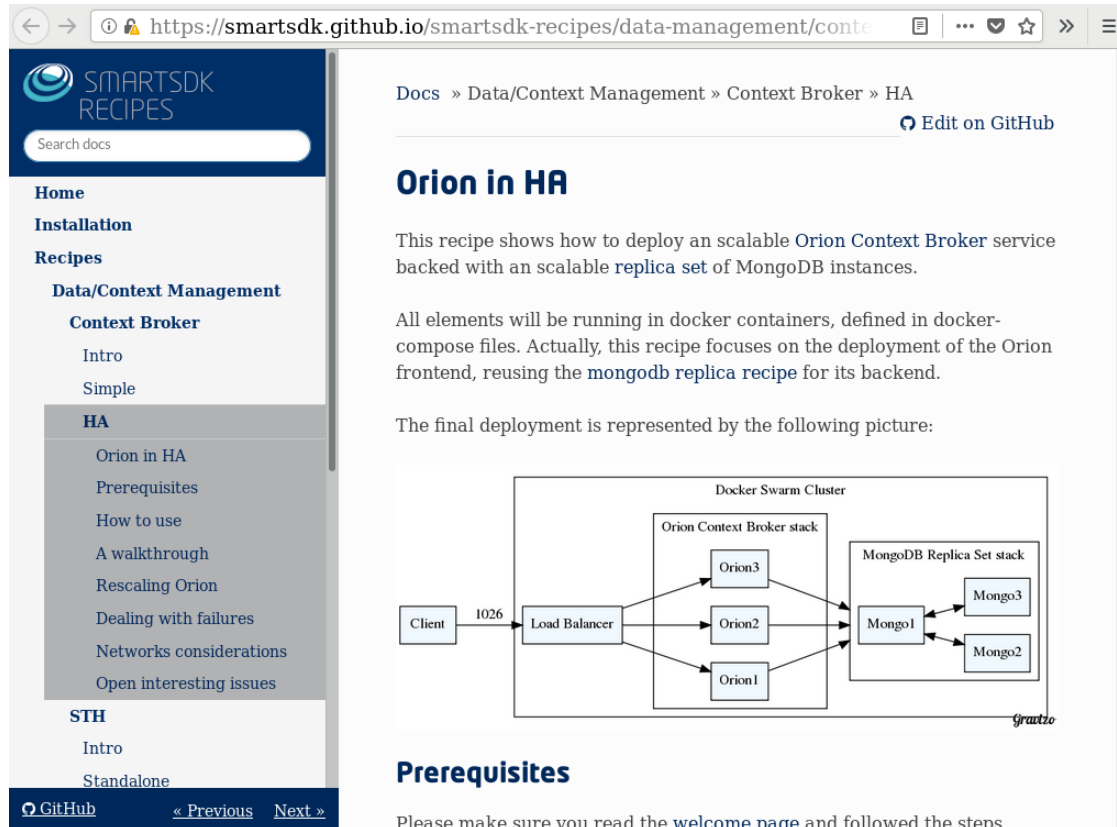
For the “Orion Context Broker” there are optional values that can be changed.



The screenshot shows a web browser window with the URL `https://platform-manager.smartsdk.eu/r/projects/1a69/portaine`. The page is titled "Application templates list" and "Templates". The left sidebar shows the "FIWARE" logo and navigation menus for "ACTIVE ENDPOINT" (with "fiware-hosts4" selected), "ENDPOINT ACTIONS" (Dashboard, App Templates, Stacks, Services, Containers, Images, Networks, Volumes, Configs, Secrets, Swarm), and "PORTAINER SETTINGS" (Endpoints, Registries, Settings). The main content area is for the "Orion Context Broker" template. It includes an "Information" section stating "Requires existing external networks: frontend, backend. See the [documentation](#)." and a "Configuration" section with input fields for "Name" (placeholder: "e.g. myStack"), "Orion Version", "Mongo Service URI", "Replica Set Name", and "Docker MTU". An "Actions" section contains a "Deploy the stack" button. At the bottom, there is a "Templates" section with a "Category" dropdown set to "All" and filter buttons for "All", "Windows", and "Linux".

Figure 26: Orion Context Broker Application settings

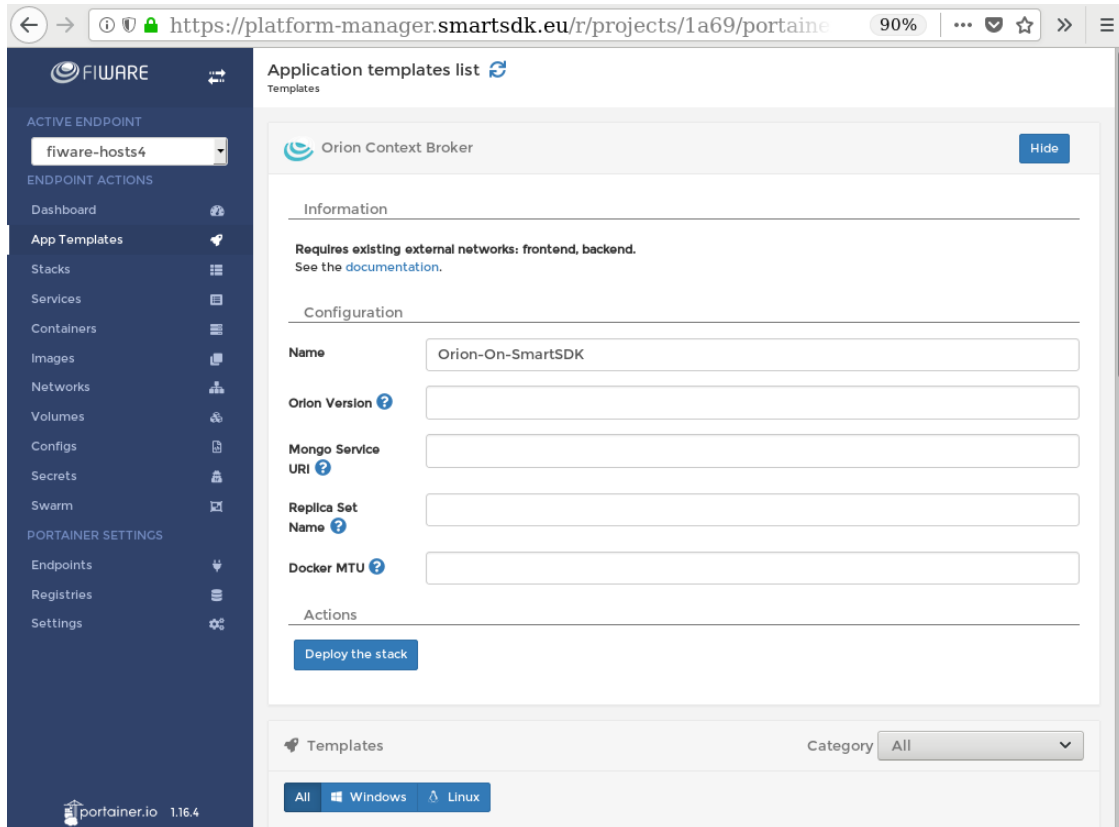
A link to the documentation is provided in order to clarify the exact meaning of the variables.



The screenshot shows a web browser displaying the SMARTSDK Recipes documentation. The left sidebar contains a navigation menu with sections: Home, Installation, Recipes, Data/Context Management, Context Broker, and HA. The main content area is titled "Orion in HA" and includes a breadcrumb trail: Docs » Data/Context Management » Context Broker » HA. Below the title, there is a paragraph explaining that the recipe shows how to deploy a scalable Orion Context Broker service backed with a scalable replica set of MongoDB instances. It mentions that all elements will be running in Docker containers, defined in docker-compose files, and focuses on the deployment of the Orion frontend, reusing the mongodb replica recipe for its backend. A diagram illustrates the final deployment architecture within a Docker Swarm Cluster. A Client connects to a Load Balancer on port 1026, which distributes traffic to three Orion instances (Orion1, Orion2, Orion3). These Orion instances are connected to a MongoDB Replica Set stack consisting of three MongoDB instances (Mongo1, Mongo2, Mongo3). The diagram is signed "gratzio". Below the diagram, there is a "Prerequisites" section with a note: "Please make sure you read the welcome page and followed the steps".

Figure 27: Link to the Original Context Broker Documentation

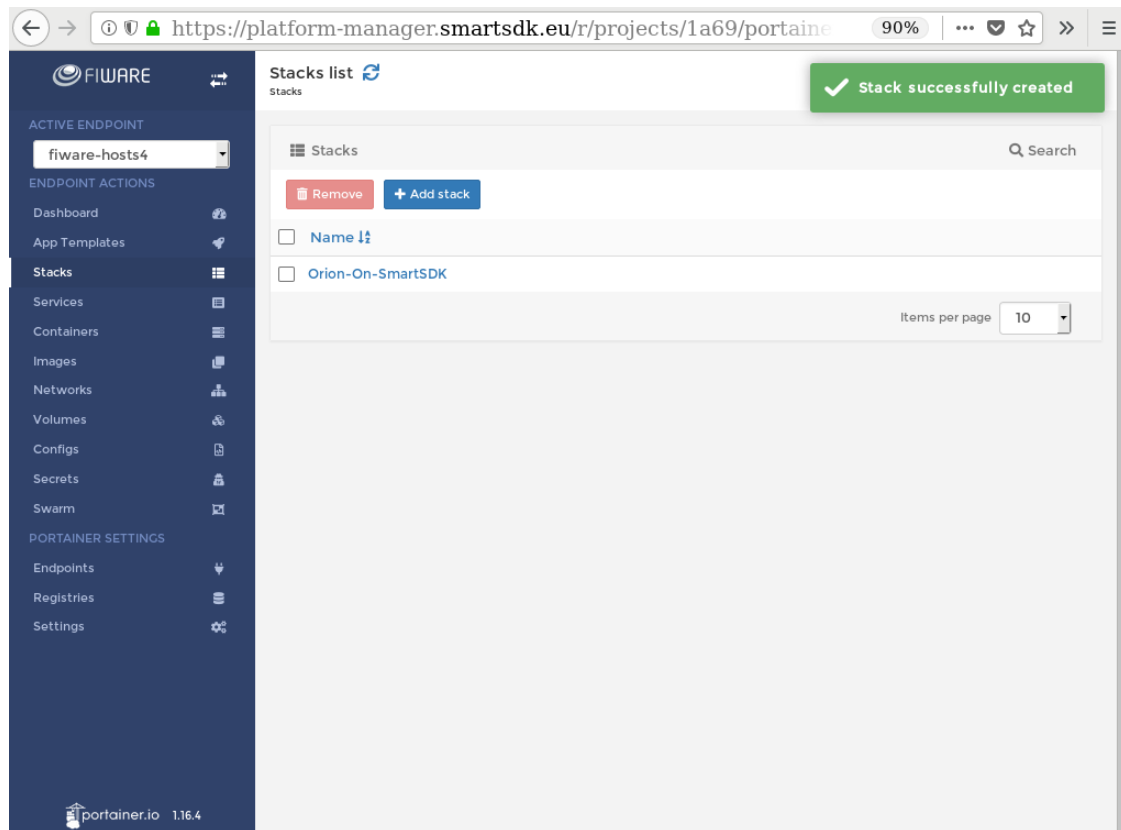
Complete the form with at least the “Stack Name”:



The screenshot shows a web browser window with the URL <https://platform-manager.smartsdk.eu/r/projects/1a69/portaine>. The page displays the "Application templates list" for "Orion Context Broker". The left sidebar contains the "FIWARE" logo and navigation menus for "ACTIVE ENDPOINT" (showing "fiware-hosts4"), "ENDPOINT ACTIONS" (Dashboard, App Templates, Stacks, Services, Containers, Images, Networks, Volumes, Configs, Secrets, Swarm), and "PORTAINER SETTINGS" (Endpoints, Registries, Settings). The main content area is titled "Orion Context Broker" and includes a "Hide" button. It contains sections for "Information" (requiring external networks), "Configuration" (with fields for Name, Orion Version, Mongo Service URI, Replica Set Name, and Docker MTU), and "Actions" (with a "Deploy the stack" button). At the bottom, there is a "Templates" section with a "Category" dropdown set to "All" and filters for "All", "Windows", and "Linux". The footer of the sidebar shows "portainer.io 1.16.4".

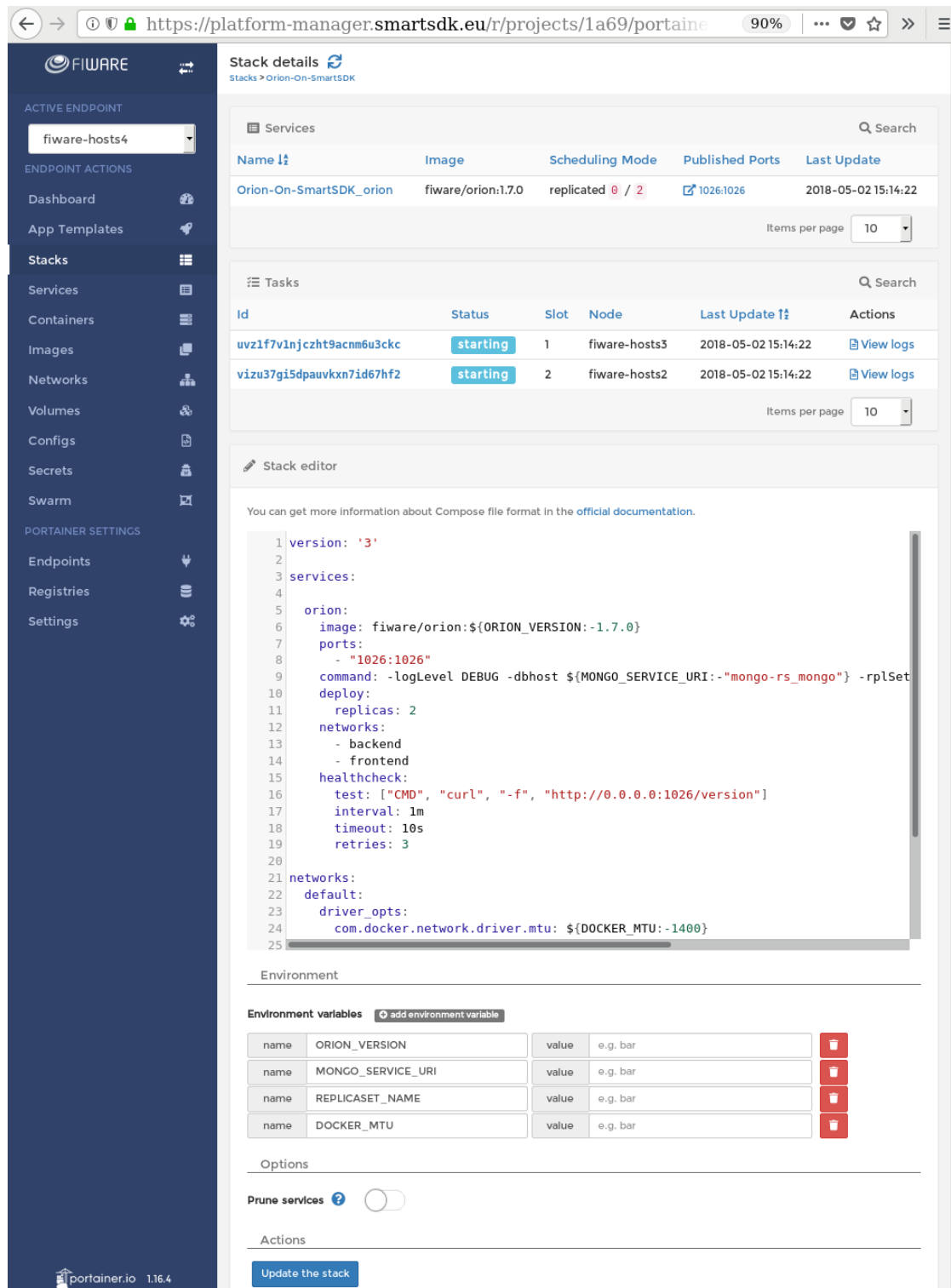
Figure 28: Complete the Orion Context Broker Form

Click on “Deploy the stack” and wait a bit for the starting of the stack.



*Figure 29: Successful start of a deploy*

Note that a configuration can be edited at any time in order to change suitable parameters.



The screenshot shows the Portainer web interface at <https://platform-manager.smartsdk.eu/r/projects/1a69/portainer>. The left sidebar contains navigation menus for 'ACTIVE ENDPOINT' (fiware-hosts4), 'ENDPOINT ACTIONS' (Dashboard, App Templates, Stacks, Services, Containers, Images, Networks, Volumes, Configs, Secrets, Swarm), and 'PORTAINER SETTINGS' (Endpoints, Registries, Settings). The main content area is titled 'Stack details' and shows the configuration for the 'Orion-On-SmartSDK\_orion' stack.

**Services Table:**

Name	Image	Scheduling Mode	Published Ports	Last Update
Orion-On-SmartSDK_orion	fiware/orion:1.7.0	replicated 0 / 2	1026:1026	2018-05-02 15:14:22

**Tasks Table:**

Id	Status	Slot	Node	Last Update	Actions
uvz1f7v1njczht9acnm6u3ckc	starting	1	fiware-hosts3	2018-05-02 15:14:22	<a href="#">View logs</a>
vizu37gi5dpauvkn7id67hf2	starting	2	fiware-hosts2	2018-05-02 15:14:22	<a href="#">View logs</a>

**Stack editor:** A code editor showing the Compose file configuration for the stack.

```

1 version: '3'
2
3 services:
4
5   orion:
6     image: fiware/orion:${ORION_VERSION:-1.7.0}
7     ports:
8       - "1026:1026"
9     command: -logLevel DEBUG -dbhost ${MONGO_SERVICE_URI:-"mongo-rs-mongo"} -rplSet
10    deploy:
11      replicas: 2
12    networks:
13      - backend
14      - frontend
15    healthcheck:
16      test: ["CMD", "curl", "-f", "http://0.0.0.0:1026/version"]
17      interval: 1m
18      timeout: 10s
19      retries: 3
20
21 networks:
22   default:
23     driver_opts:
24       com.docker.network.driver.mtu: ${DOCKER_MTU:-1400}
25

```

**Environment variables:** A table for defining environment variables.

name	value	actions
ORION_VERSION	e.g. bar	<a href="#">edit</a> <a href="#">delete</a>
MONGO_SERVICE_URI	e.g. bar	<a href="#">edit</a> <a href="#">delete</a>
REPLICASET_NAME	e.g. bar	<a href="#">edit</a> <a href="#">delete</a>
DOCKER_MTU	e.g. bar	<a href="#">edit</a> <a href="#">delete</a>

**Options:** A section for additional options, including a toggle for 'Prune services'.

**Actions:** A section with an 'Update the stack' button.

Figure 30: Edit the configuration of a running deploy

This ends our web graphical user interface tour. The next section explores the command line oriented tools.

## 2.7 Export configuration for Docker CLI

Once the host is up you can export the machine configuration. This configuration is useful if you want to manage the host using the **docker-machine** tool. You can also use the configuration to connect to the host directly using **ssh**.

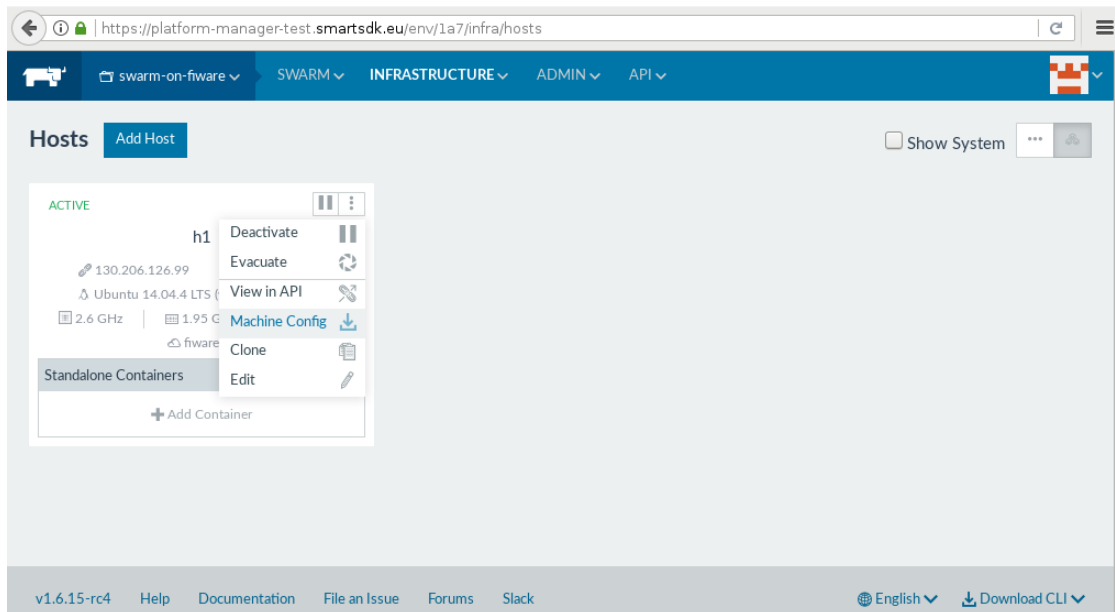


Figure 31: Add hosts configuration details

For the ssh connection see the following example. Extract the configuration.

```
user@localhost tar xvzf h1.tar.gz
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/certs
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/certs/ca-key.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/certs/ca.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/certs/cert.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/certs/key.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/ca.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/cert.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/config.json
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/created
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/id_rsa
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/id_rsa.pub
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/key.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/server-key.pem
f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/server.pem
```

Use ssh to connect to the host and show the running docker container.

```
user@localhost ssh -i f92db4d8-5b28-44d8-ae54-7fcb823e2e4a/machines/h1/id_rsa \
-o IdentitiesOnly=yes ubuntu@130.206.126.99 sudo docker ps
```



```
#####
#####
NOTE: You have accessed a system owned by FIWARE Lab. You must have
authorisation
before using it, and your use will be strictly limited to that indicated in
the
authorisation.
Unauthorised access to this system or improper use of the same is
prohibited and
is against the FIWARE Terms & Conditions Policy and the legislation in
force. The
use of this system may be monitored.
#####
#####
```

CONTAINER ID	IMAGE	STATUS	PORTS	COMMAND	NAMES
1f6bc6ebfee8	portainer/portainer:pr572			"/portainer --no-	r-
a..."	2 hours ago	Up 2 hours			
portainer-portainer-ui-1-adaec9cb					
15a9693cbca5	rancher/portainer-agent:v0.1.0			"/.r/r portainer-	r-
a..."	2 hours ago	Up 2 hours			
portainer-portainer-1-08b16b2d					
95b1d98105b9	rancher/scheduler:v0.8.3			"/.r/r /rancher-	r-
en..."	2 hours ago	Up 2 hours			
scheduler-scheduler-1-59a39b48					
13a513eddb52	rancher/net:v0.13.9			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-ipsec-ipsec-connectivity-check-3-25da01ae					
1d8863a459c6	rancher/net:v0.13.9			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-ipsec-ipsec-router-3-8d16ea87					
5ac088c73d44	rancher/net:holder			"/.r/r /rancher-	r-
en..."	2 days ago	Up 2 days			
ipsec-ipsec-3-e7a7301d					
2277dc19441a	rancher/net:v0.13.9			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-ipsec-cni-driver-1-81ee523d					
04262f5583fe	rancher/dns:v0.17.2			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-network-services-metadata-dns-1-30407e50					
dfe285a4a9cb	rancher/healthcheck:v0.3.3			"/.r/r /rancher-	r-
en..."	2 days ago	Up 2 days			
healthcheck-healthcheck-1-fef6c66b					
c40e56bd9b43	rancher/metadata:v0.10.2			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-network-services-metadata-1-5dc37eca					
81391c45319b	rancher/network-manager:v0.7.20			"/rancher-	
entrypoi..."	2 days ago	Up 2 days			
r-network-services-network-manager-1-870cfe55					

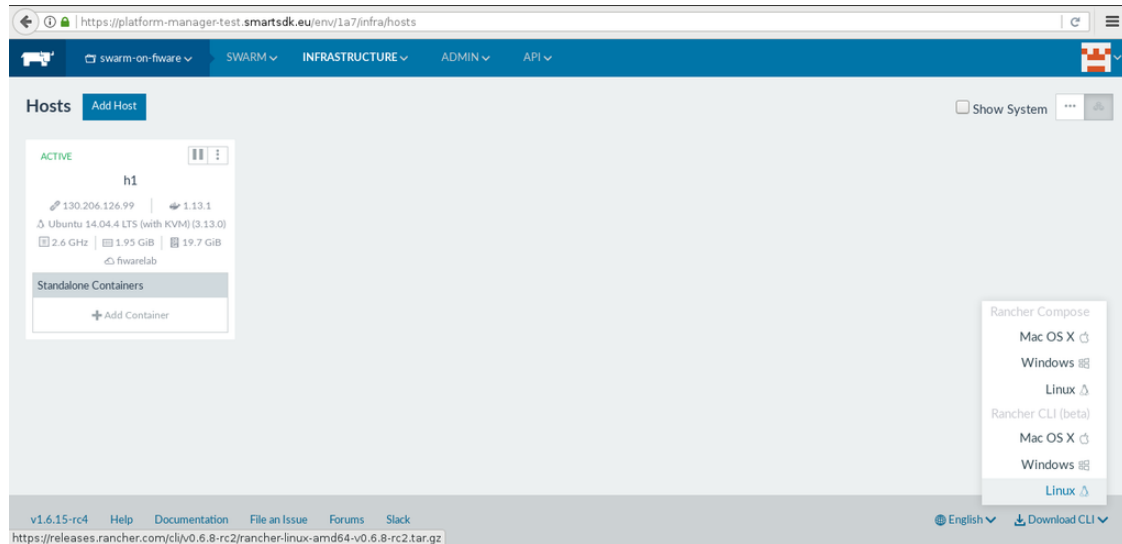
**1d3df351c60e**  
2 days ago

**rancher/agent:v1.2.10-rc3**  
Up 2 days

**"/run.sh run"**  
**rancher-agent**

In order to use the rancher CLI, you need to download the tools and the API keys.

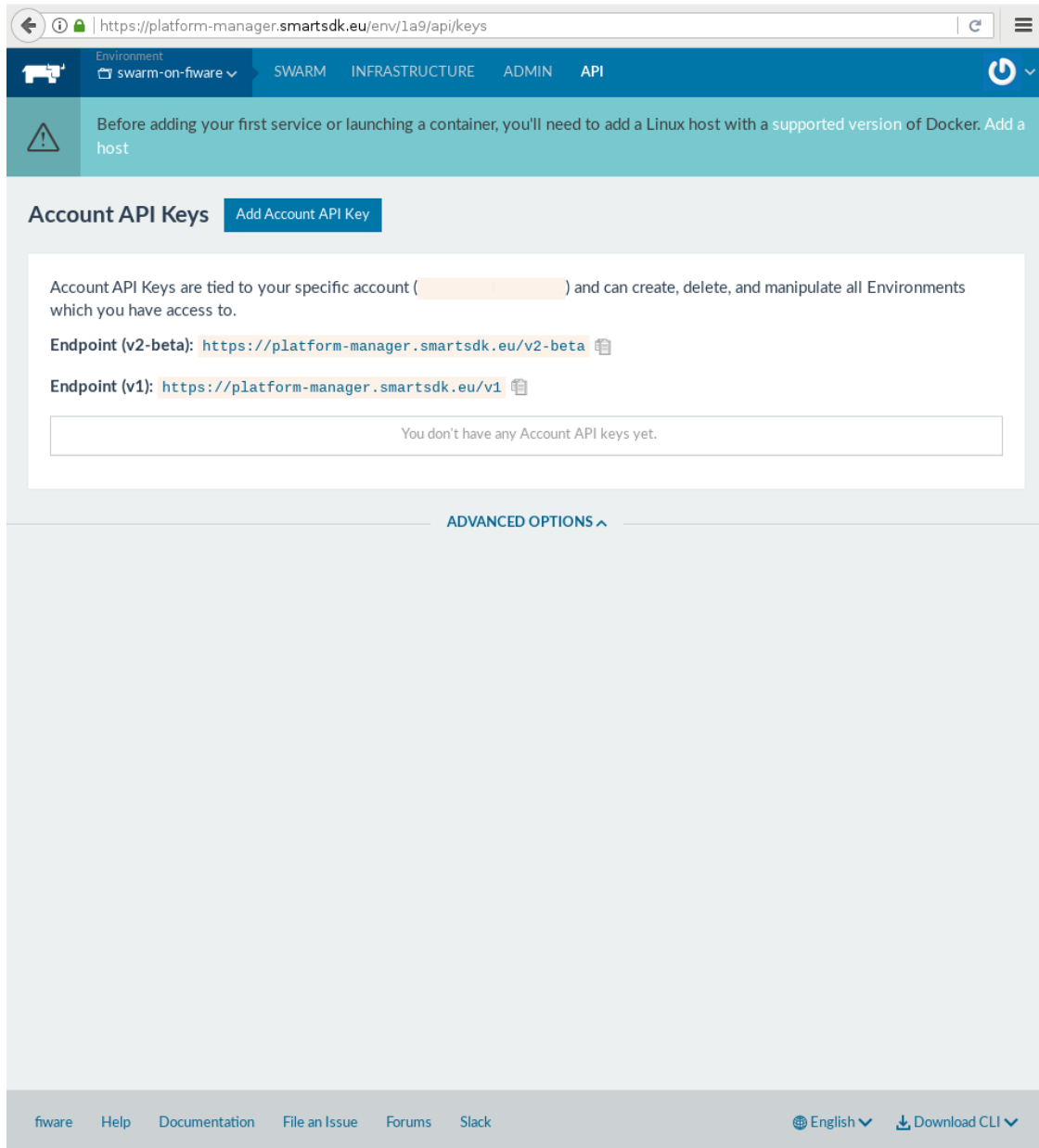
Download them from the link at the right bottom corner of the interface “Download CLI”.



*Figure 32: Download Rancher CLI*

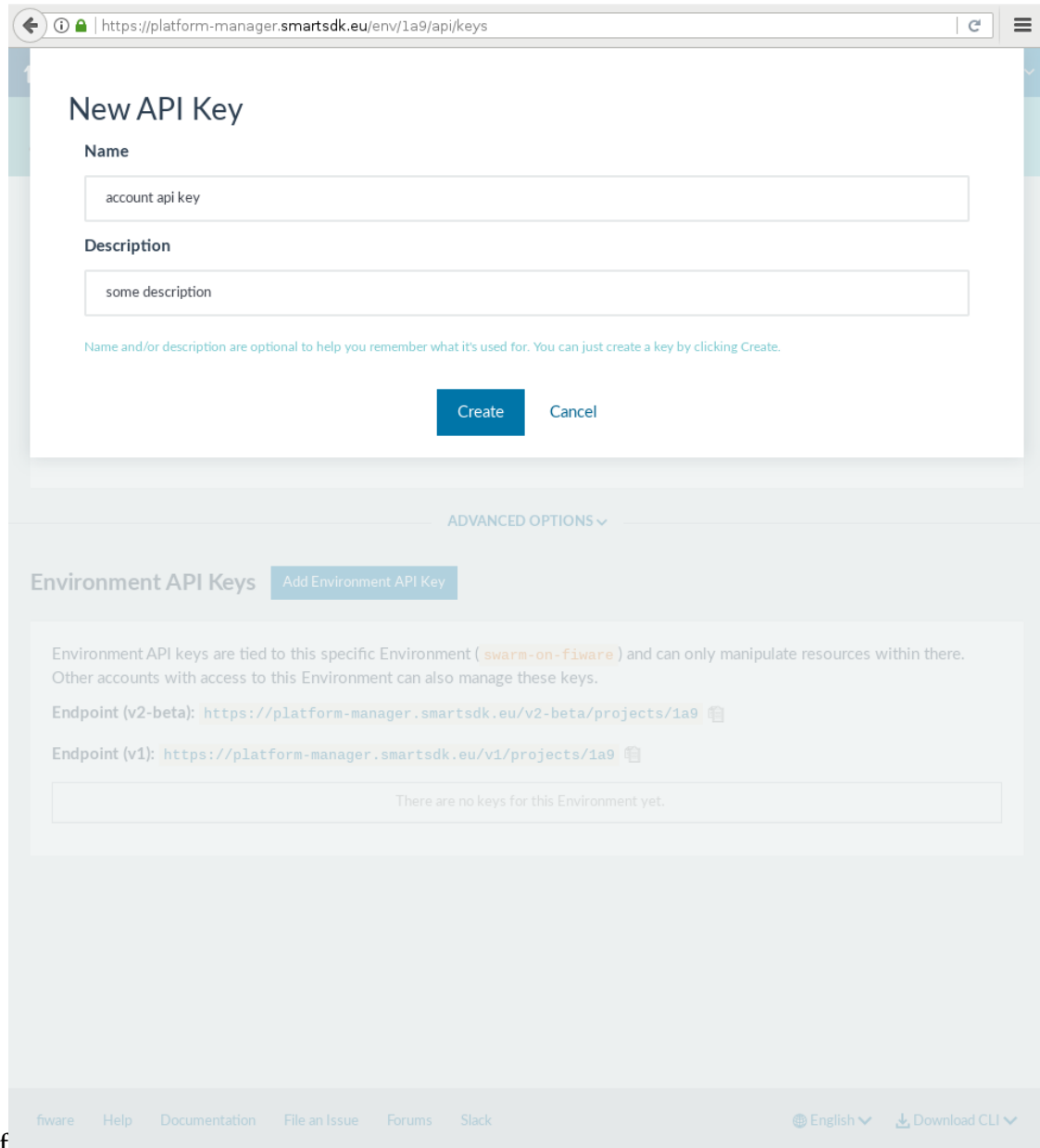
Download the account and environment API keys from the API tab. Make sure you have selected the correct environment.

An overview of the API page. Click on “Add Account API Key”.



*Figure 33: API Creation and download page*

Fill the name and description for the account API key



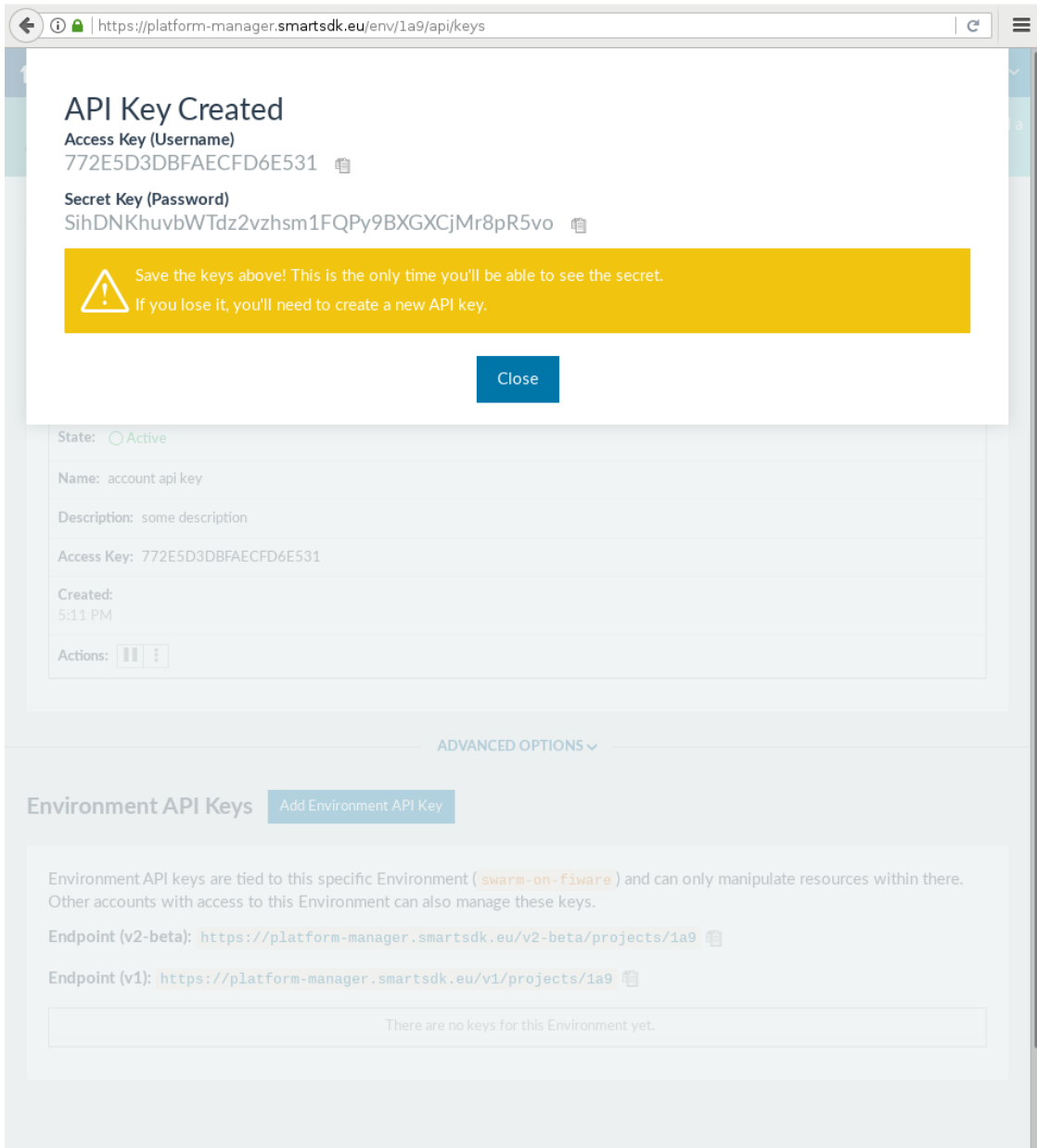
The screenshot shows a web browser window with the URL `https://platform-manager.smartsdk.eu/env/1a9/api/keys`. The page title is "New API Key". It contains two input fields: "Name" with the value "account api key" and "Description" with the value "some description". Below these fields is a note: "Name and/or description are optional to help you remember what it's used for. You can just create a key by clicking Create." At the bottom of the form are two buttons: "Create" (in blue) and "Cancel".

Below the form is a section titled "Environment API Keys" with a sub-header "ADVANCED OPTIONS". It includes a button "Add Environment API Key". The text explains: "Environment API keys are tied to this specific Environment ( `swarm-on-fiware` ) and can only manipulate resources within there. Other accounts with access to this Environment can also manage these keys." It lists two endpoints: "Endpoint (v2-beta): `https://platform-manager.smartsdk.eu/v2-beta/projects/1a9`" and "Endpoint (v1): `https://platform-manager.smartsdk.eu/v1/projects/1a9`". A message at the bottom of this section states: "There are no keys for this Environment yet."

The footer of the page contains links: "fiware", "Help", "Documentation", "File an Issue", "Forums", "Slack", "English", and "Download CLI".

*Figure 34: New account API key creation*

Take note of the access and secret keys in a secure place.



The screenshot shows a web browser window at <https://platform-manager.smartsdk.eu/env/1a9/api/keys>. A modal dialog titled "API Key Created" is displayed, showing the newly generated Access Key (Username) and Secret Key (Password). A yellow warning box advises saving the keys as they are only visible once. Below the modal, a table lists the API key details, including its state (Active), name, description, and creation time. An "ADVANCED OPTIONS" section is also visible, showing environment-specific API key information and a list of keys for the current environment.

API Key Created

Access Key (Username)  
772E5D3DBFAECFD6E531

Secret Key (Password)  
SihDNKhuvbWTdz2vzhsm1FQPy9BXGXCjMr8pR5vo

Save the keys above! This is the only time you'll be able to see the secret.  
If you lose it, you'll need to create a new API key.

Close




State: ☒ Active

Name: account api key

Description: some description

Access Key: 772E5D3DBFAECFD6E531

Created:  
5:11 PM

Actions:   

ADVANCED OPTIONS

Environment API Keys [Add Environment API Key](#)

Environment API keys are tied to this specific Environment ( **swarm-on-fiware** ) and can only manipulate resources within there.  
Other accounts with access to this Environment can also manage these keys.

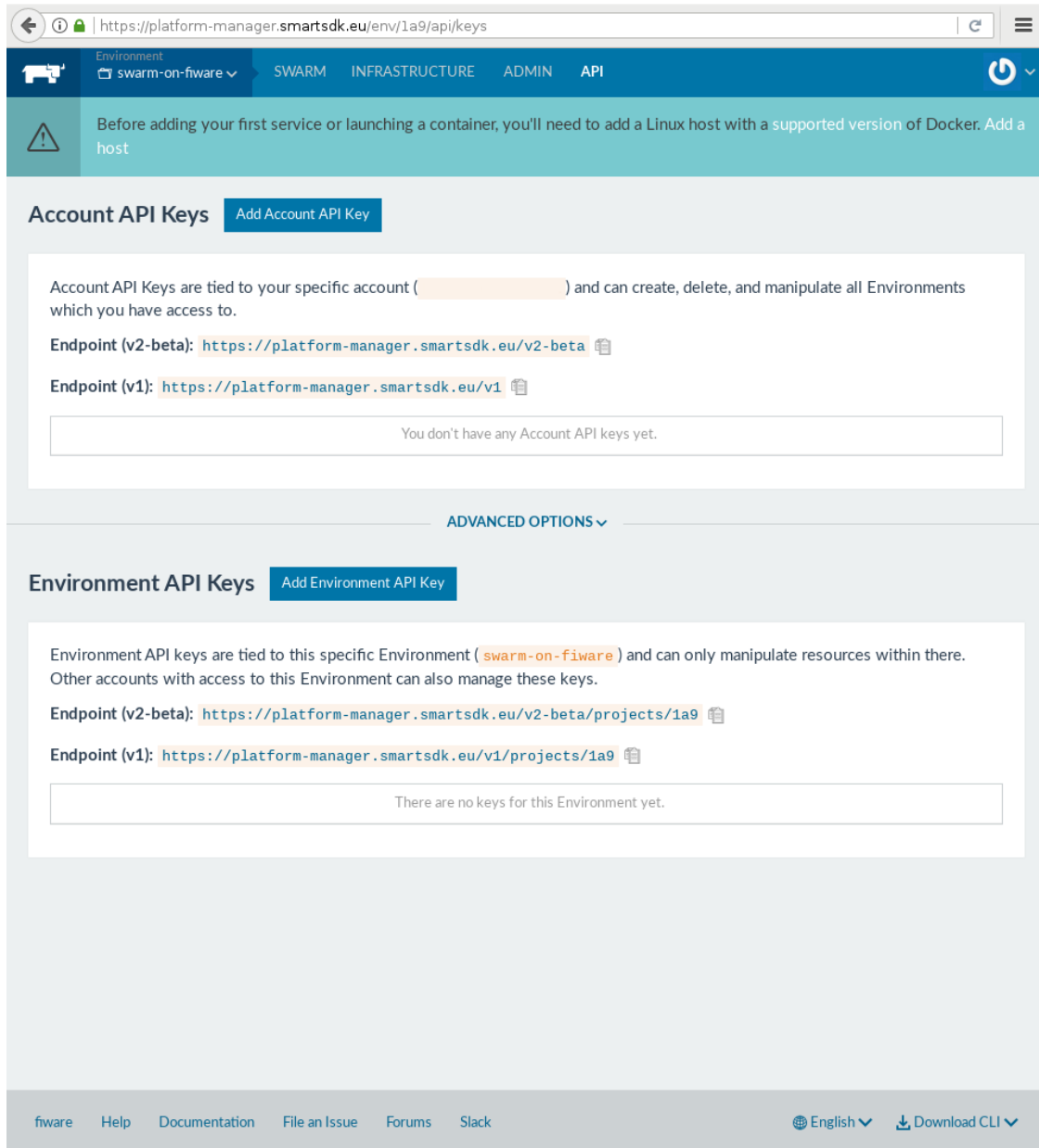
Endpoint (v2-beta): <https://platform-manager.smartsdk.eu/v2-beta/projects/1a9>

Endpoint (v1): <https://platform-manager.smartsdk.eu/v1/projects/1a9>

There are no keys for this Environment yet.

Figure 35: Account key tokens

Fill the name and description for the account API key



Environment  
swarm-on-fiware ▾ SWARM INFRASTRUCTURE ADMIN API

Before adding your first service or launching a container, you'll need to add a Linux host with a supported version of Docker. Add a host

### Account API Keys [Add Account API Key](#)

Account API Keys are tied to your specific account ( ) and can create, delete, and manipulate all Environments which you have access to.

Endpoint (v2-beta): <https://platform-manager.smartsdk.eu/v2-beta>

Endpoint (v1): <https://platform-manager.smartsdk.eu/v1>

You don't have any Account API keys yet.

[ADVANCED OPTIONS ▾](#)

### Environment API Keys [Add Environment API Key](#)

Environment API keys are tied to this specific Environment ( **swarm-on-fiware** ) and can only manipulate resources within there. Other accounts with access to this Environment can also manage these keys.

Endpoint (v2-beta): <https://platform-manager.smartsdk.eu/v2-beta/projects/1a9>

Endpoint (v1): <https://platform-manager.smartsdk.eu/v1/projects/1a9>

There are no keys for this Environment yet.

fware Help Documentation File an Issue Forums Slack English ▾ Download CLI ▾

*Figure 36: New environment key creation*

Take note of the access and secret keys in a secure place.

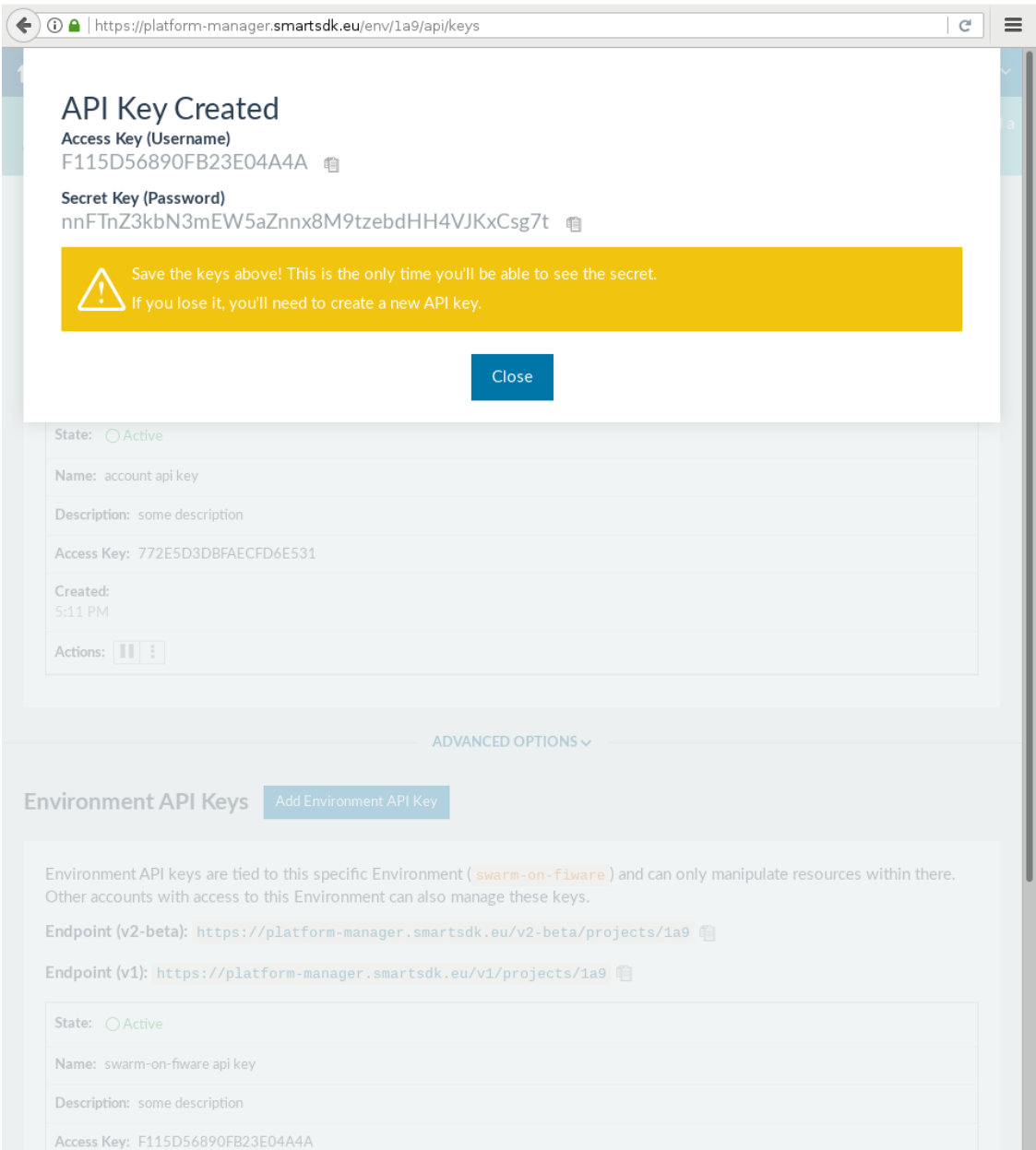


Figure 37: Environment key tokens

## 3 SMARTSDK PLATFORM USAGE

---

### 3.1 Introduction

A number of steps need to be followed in order to have a working docker swarm cluster. First, an environment template must be configured, then an environment must be added, then some hosts running docker must be added and finally docker must be configured for swarm mode on each of those hosts. Each step can be completed by choosing multiple options. For each option we will detail the pros and cons. We will spend some time especially into detailing the solutions or the workarounds that works well on a [FIWARE Lab](https://www.fiware.org/lab/)<sup>11</sup> installation. Most of the workarounds and custom configurations were integrated in the templates provided. This documentation is provided as a reference.

### 3.2 Environment Templates

The SmartSDK Platform uses environment templates in order to offer some configured templates with default values. Users can either choose one of the default templates or start the creation of a new one.

Each template contains a predefined set of services and configuration for the environment. For example you may want to add to the template, or simply reconfigure, the “Rancher IPsec” overlay network, the “Rancher NFS” or the “Portainer.io” web user interface.

The SmartSDK Platform allow the creation of templates for “Docker Swarm”.

Proper attention must be dedicated for the configuration of the:

- ➔ The Number of Swarm Managers
- ➔ Rancher IPsec plugin MTU (FIWARE Lab)
- ➔ (Optional) Rancher NFS plugin

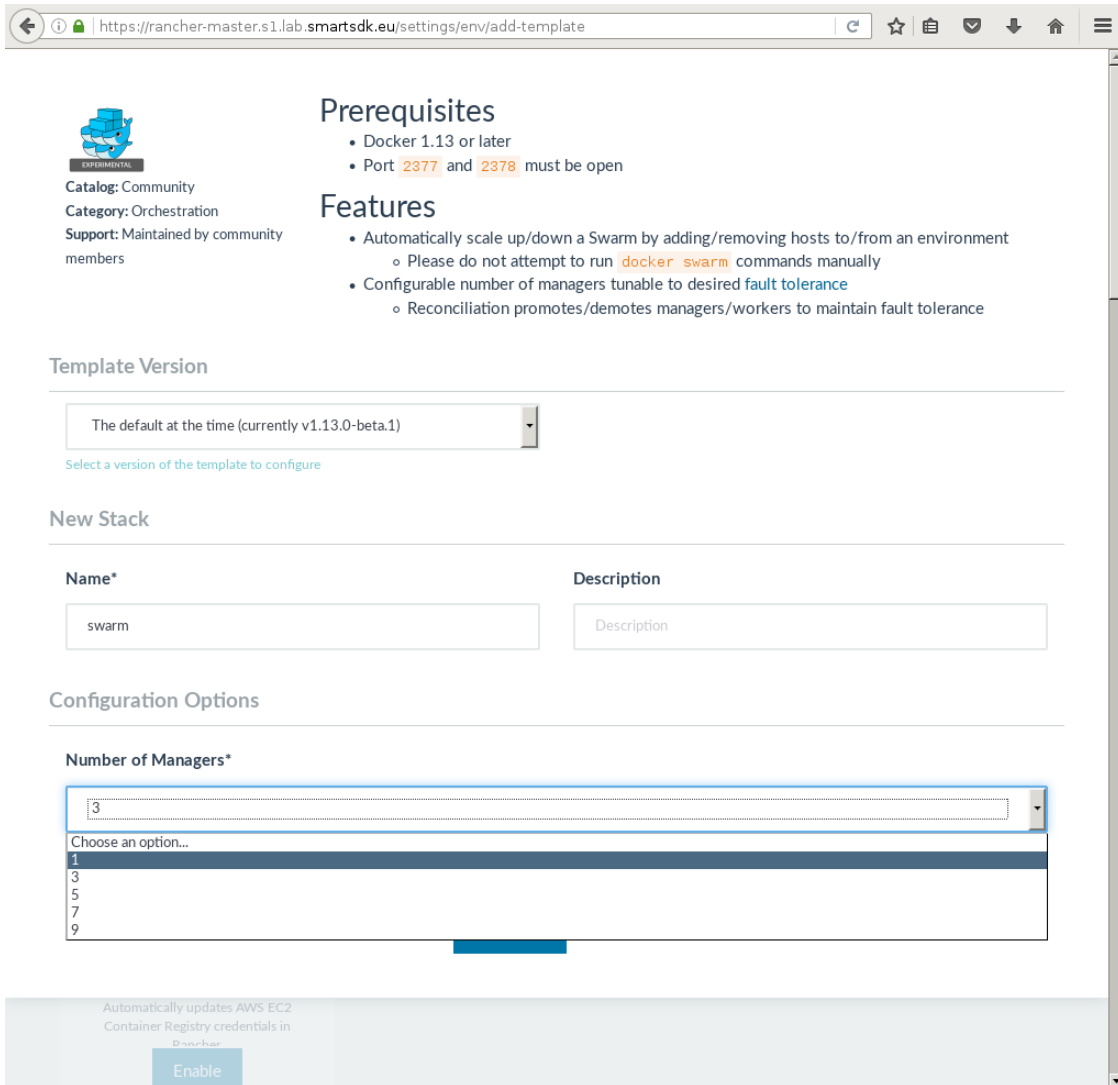
#### 3.2.1 The Number of Swarm Managers

The Number of Swarm Managers in the template will affect the high availability of the swarm mode. The default number is 3, but can be lowered to 1 for simple installation for evaluation purposes where the high availability of the managers is not needed. See the screenshot at *Figure 38: Setting the manager number in environment template settings*.

---

<sup>11</sup> <https://www.fiware.org/lab/>





https://rancher-master.s1.lab.smartsdk.eu/settings/env/add-template

**Prerequisites**

- Docker 1.13 or later
- Port 2377 and 2378 must be open

**Features**

- Automatically scale up/down a Swarm by adding/removing hosts to/from an environment
  - Please do not attempt to run `docker swarm` commands manually
- Configurable number of managers tunable to desired **fault tolerance**
  - Reconciliation promotes/demotes managers/workers to maintain fault tolerance

**Template Version**

The default at the time (currently v1.13.0-beta.1)

Select a version of the template to configure

**New Stack**

Name\* swarm

Description Description

**Configuration Options**

Number of Managers\*

3

Choose an option...

1

3

5

7

9

Automatically updates AWS EC2 Container Registry credentials in Docker

Enable

Figure 38: Setting the manager number in environment template settings

### 3.2.2 Rancher IPsec plugin MTU (Fiware LAB)

**How to find out the MTU of your host.** If your provider does not offer any documentation regarding the default MTU you can search the MTU value by yourself.

In order to find out the MTU of the device connected to the default gateway (which usually is the one that allow also local area network connectivity) of your host, connect to it and issue the following commands:

```
# Find out the device that is the default gateway
DEFAULT_GW_DEV=$(ip route | awk '/^default/ {print $NF; exit}')
# Find out the MTU of the default gateway
DEFAULT_GW_MTU=$(ip addr show "${DEFAULT_GW_DEV}" | grep -oP '(?<=mtu )[\d-9]*')
printf "%s\n" "${DEFAULT_GW_MTU}"
```

If the value is lower than the common value of **1500** bytes, you should take additional care because the overlay network created to allow the communication between the swarm cluster nodes assumes the default value of **1500** bytes.

A wrong configuration of the MTU can prevent proper communication between nodes and make the system totally unusable.

### 3.2.3 Configure the Rancher IPsec plugin MTU

The MTU for the Rancher IPsec plugin must be lower or equal to the one of the host (found out following the previous section).

For example on the Spain2 FIWARE Lab node the MTU is currently **1400** bytes.

### 3.2.4 (Optional) Rancher NFS plugin

It is possible to add to the environment template an NFS server in order to have a shared storage service available to the SmartSDK Platform Manager. The NFS server must be reachable from the swarm nodes. The parameters to configure are:

- ➔ the IP address of the server
- ➔ the exported path

## 3.3 Environment

Starting from a template the user can instantiate an Environment. In order to have a fully functional environment, a number of hosts equal or greater than the number of swarm managers selected in the template must be added to the environment.

## 3.4 Host requirements

Host requirements varies from version to version. In general the exact requirements are listed in the template. The minimal setup will be satisfied with the following:

- ➔ A modern Linux distribution
- ➔ Installation of docker version 17.12.0-ce
- ➔ Port 2377 and 2378 must be open between the hosts

## 3.5 Hosts on the FIWARE Lab

It is possible, and encouraged, to add nodes running on a project in the FIWARE Lab. We assume that the entire project is dedicated to the environment.

## 3.6 Setup the OpenStack project for hosting a SmartSDK platform environment

Before the host creation, the OpenStack project needs to be configured with proper access rules and images.

The following settings are working settings but not the best secure setup. For example you may want to restrict incoming connection to management ports only from a well known subset of IPs.

## 3.7 OpenStack client Setup

- ➔ Install **python-openstackclient**, in order to have the tool called **openstack**

- ➔ Load your local **openrc** file in order to have the setting for your account loaded

### 3.7.1 Install python-openstackclient

To install the OpenStack command line client, on a modern ubuntu-based Linux distribution you need to issue the command:

```
sudo apt install --yes python3-openstackclient
```

This will install the distribution supplied client, that from time to time may be an outdated version. You can install the latest OpenStack client using various methods. See for example: Install modern openstackclient with pip.

### 3.7.2 Load the OpenStack client settings

To load your settings and credential you need to set some well known OpenStack related variables. One of the most common way to do it is to source the **openrc** file.

```
. openrc
```

## 3.8 Start with a clean OpenStack and docker-machine environment

This may cause data loss! Double check the source of the correct **openrc** file.

- ➔ Clean up servers (also known as hosts):

```
openstack server list -f value -c ID | xargs -trn1 openstack server delete  
openstack server list
```

- ➔ Clean up volumes:

```
openstack volume list -f value -c ID | xargs -trn1 openstack volume delete  
openstack volume list
```

- ➔ Clean up security groups:

This will not work with python-openstackclient 2.3.0 shipped with ubuntu 16.04. See Install modern openstackclient with pip for a workaround.

```
openstack security group list -f value -c ID | xargs -trn1 openstack  
security group delete  
openstack security group rule list -f value -c ID default | \  
  xargs -trn1 openstack security group rule delete  
openstack security group set default --description 'empty default'  
openstack security group list
```

- ➔ Clean up keypair:

```
openstack keypair list -f value -c Name | xargs -trn1 openstack keypair  
delete  
openstack keypair list
```

You may also want to cleanup:

- ➔ Clean up floating IP reservation
- ➔ Clean up snapshots

- ➔ Clean up storage containers
- ➔ Clean up images
- ➔ Clean up flavors

### 3.9 Add proper security group roles

The configuration of the networking for the hosts that belong to the Rancher cluster offers a lot of options. In this section we will setup security roles for a group of hosts that is started on a generic OpenStack installation. This allows the binding of a public IP for each host in order to allow the direct connectivity with the Rancher server.

- ➔ Allow cluster communication among Rancher nodes:

```
openstack security group create rancher-cluster \
  --description "Security group among Rancher nodes"
```

```
openstack security group rule create rancher-cluster \
  --protocol tcp --dst-port 22:22 --remote-group rancher-cluster
```

- ➔ Allow IPsec according to the [documentation](#) <sup>12</sup> :

```
openstack security group rule create rancher-cluster \
  --protocol udp --dst-port 500:500 --remote-group rancher-cluster
openstack security group rule create rancher-cluster \
  --protocol udp --dst-port 4500:4500 --remote-group rancher-cluster
```

- ➔ Allow access to health-check:

```
openstack security group rule create rancher-cluster \
  --protocol tcp --dst-port 80:80 --remote-group rancher-cluster
```

- ➔ Allow access to docker-engine and [docker-swarm](#) <sup>13</sup> daemons:

```
# Port 2376 2377 2388
```

```
openstack security group rule create rancher-cluster \
  --protocol tcp --dst-port 2376:2378 --remote-group rancher-cluster
```

- ➔ If you use docker machine for the public network, the ports must be publicly available:

```
openstack security group rule create rancher-cluster \
  --protocol tcp --dst-port 2376:2378
```

- ➔ Allow access for [swarm ingress network](#) <sup>14</sup> :

```
openstack security group rule create rancher-cluster \
  --protocol tcp --dst-port 7946:7946 --remote-group rancher-cluster
openstack security group rule create rancher-cluster \
  --protocol udp --dst-port 7946:7946 --remote-group rancher-cluster
openstack security group rule create rancher-cluster \
  --protocol udp --dst-port 4789:4789 --remote-group rancher-cluster
```

<sup>12</sup> <https://docs.rancher.com/rancher/v1.0/en/rancher-ui/infrastructure/hosts/custom/#security-groupsfirewalls>

<sup>13</sup> <https://docs.docker.com/engine/swarm/swarm-tutorial/#open-protocols-and-ports-between-the-hosts>

<sup>14</sup> <https://docs.docker.com/engine/swarm/ingress/>

➔ (Optional) Allow NFSv4 port access to cluster security group:

```
openstack security group rule create rancher-cluster \
--protocol tcp --dst-port 2049:2049 --remote-group rancher-cluster
```

### 3.10 Docker Machine

Docker Machine enables the fast deployment of new hosts with docker installed and ready to use. Docker machine relies on specific components called “machine drivers”, in order to interface with the underlying cloud. The “machine drivers” are pluggable components. OpenStack is already supported. The FIWARE Lab is supported by using the OpenStack native driver, or by using the already mentioned Machine driver and User Interface Plugin for FIWARE Lab Nodes.

#### 3.10.1 Resolve dependencies for docker-machine

docker-machine is a young and fast moving project. Chances are that your distribution is shipping and outdated version, if any. In order to satisfy the requirements, even from a stripped bare image.

➔ Install curl

```
sudo apt install --yes curl
```

#### 3.10.2 Install the docker-machine

➔ You may want to have a look to the official [documentation](#)<sup>15</sup>.

➔ Install docker-machine:

```
MACHINE_VERSION="v0.14.0"
```

```
MACHINE_BASE_URL="https://github.com/docker/machine/releases/download"
OS_NAME="$(uname -s)"
OS_ARCH="$(uname -m)"
MACHINE_URL="${MACHINE_BASE_URL}/${MACHINE_VERSION}/docker-
machine/${OS_NAME}-${OS_ARCH}"
curl -s -L "${MACHINE_URL}" \
> /tmp/docker-machine && \
chmod +x /tmp/docker-machine && \
sudo cp /tmp/docker-machine /usr/local/bin/docker-machine
```

➔ Show the docker machine version:

```
docker-machine version
```

#### 3.10.3 Start with a clean docker-machine environment

This may cause data loss! Perform the following steps only if you want to destroy all the hosts created with docker-machine.

➔ Clean up local docker-machine:

```
docker-machine ls -q -f Name | xargs -trn1 docker-machine rm --force
docker-machine ls
```

<sup>15</sup> <https://docs.docker.com/machine/install-machine/#installing-machine-directly>

### 3.10.4 Setup docker-machine from the command line interface

- ➔ Set docker machine parameters:

```
export
MACHINE_DOCKER_INSTALL_URL='https://releases.rancher.com/install-
docker/17.12.sh'
export MACHINE_DRIVER='openstack'
```

- ➔ Define the parameters related to the security groups previously defined in the section Add proper security group roles:

```
export OS_SECURITY_GROUPS='external,rancher-cluster'
```

- ➔ Define variables related to the underling OpenStack installation. The following defaults are also used on most nodes of the FIWARE Lab nodes:

```
# Usually the name is 'default'
export OS_DOMAIN_NAME='default'
# This is the usual network names on the nodes of FIWARE Lab, check
also with
# openstack network list --column Name
export OS_NETWORK_NAME='node-int-net-01'
export OS_FLOATINGIP_POOL='public-ext-net-01'
```

- ➔ Define the variables relative to the images and flavor. Usually those are specific of a node. See List Available Images in an OpenStack Project and List Available Flavors in an OpenStack Project in order to find suitable values. The supported and tested values are:

```
export OS_IMAGE_NAME='base_ubuntu_16.04'
export OS_FLAVOR_NAME='m1.medium'
```

- ➔ The user used for ssh connection is image specific, usually it takes the name of the Linux distribution or your cloud provider should have some specific documentation.

- Usual value for ubuntu images:

```
export OS_SSH_USER='ubuntu'
```

- ➔ To create a specific key-pair to OpenStack and tell **docker-machine** to use it, issue the following commands:

```
openstack keypair create --public-key "~/.ssh/id_rsa_deployer.pub"
deployer
export OS_KEYPAIR_NAME="deployer"
export OS_PRIVATE_KEY_FILE=~/.ssh/id_rsa_deployer
```

- ➔ Or, on the contrary, to use automatically generated keys (a different one for each host), you may want to unset the specific environment variables:

```
unset OS_PRIVATE_KEY_FILE
unset OS_KEYPAIR_NAME
```

## 3.11 Setup security groups

- ➔ Allow access from external:

```
openstack security group create external \
--description "Allow external access, for ssh, http, https, proxy"
```

```

openstack security group rule create external \
  --protocol tcp --dst-port 22:22
openstack security group rule create external \
  --protocol tcp --dst-port 80:80
openstack security group rule create external \
  --protocol tcp --dst-port 443:443
openstack security group rule create external \
  --protocol tcp --dst-port 8080:8080

```

- ➔ If the master is provisioned with docker-machine, you must also add a security rule for the docker-machine connection (it would be better to restrict a bit the source IP addresses):

```

openstack security group rule create external \
  --protocol tcp --dst-port 2376:2378

```

### 3.12 Download and install Rancher CLI

To deploy Compose stack recipes we need **rancher**. Recipes version 3.x are not fully supported, but there is a known procedure to get a working deploy. See Deployment using docker stack deploy.

```

cd
wget -c https://releases.rancher.com/cli/v0.6.4/rancher-linux-amd64-
v0.6.4.tar.gz
tar xvzf rancher-linux-amd64-v0.6.4.tar.gz
cd rancher-v0.6.4/
sudo cp rancher /usr/local/bin/
rancher --version

```

### 3.13 Create API keys

```

read -d "" APIKEY_RQ_DATA <<EOF
{
  "type": "apikey",
  "accountId": "1a1",
  "name": "name_test",
  "description": "description_test",
  "created": null,
  "kind": null,
  "removeTime": null,
  "removed": null,
  "uuid": null
}
EOF

RESPONSE="$(curl 'https://platform-manager.smartsdk.eu/v2-beta/apikey' \
  --2.0 \
  -H 'Host: platform-manager.smartsdk.eu' \
  -H 'Accept: application/json' \
  -H 'Content-Type: application/json' \
  -H 'x-api-action-links: actionLinks' \
  -H 'x-api-no-challenge: true' \
  -d "${APIKEY_RQ_DATA}")"

```

```
)"
echo $RESPONSE | jq .
```

In order to access a controlled environment you need to provide the API keys to Rancher.

Note: the API key need to be related to the environment, even when there is an account related API that may have more power (admin account).

If you delete the default enabled environment the first one will get the ID of 17a.

Complete with the values available in the Rancher web interface the following variables:

### 3.14 Write CLI configuration

```
export RANCHER_URL=https://platform-manager.smartsdk.eu
export RANCHER_ENVIRONMENT=1a7
export RANCHER_ACCESS_KEY=REPLACE_VALUE
export RANCHER_SECRET_KEY=REPLACE_VALUE
```

Use the variables to create a configuration file:

```
mkdir -p "${HOME}/.rancher"
cat <<EOF > "${HOME}/.rancher/cli.json"
{
  "accessKey": "${RANCHER_ACCESS_KEY}",
  "secretKey": "${RANCHER_SECRET_KEY}",
  "url": "${RANCHER_URL}/schemas",
  "environment": "${RANCHER_ENVIRONMENT}"
}
EOF

cat "${HOME}/.rancher/cli.json"
```

Now we are ready to use the command line client to send command to our Rancher environment.

### 3.15 Provision of Rancher hosts using machine drivers

The simplest way to create and connect Rancher hosts to the Rancher server is to use the so called machine drivers.

The machine driver requires some very specific parameters, most of the time you can supply the parameters by using environmental variables or command line parameters.

You should be able to find out the parameters from your OpenStack cloud provider by yourself by using the `openstackclient`.

The parameters are quite a lot and the first time it is easy to get lost. Read this section carefully. We use a script to rename variables that do not need to be understood in order to use the machine drivers.

#### 3.15.1 Set host parameters

Set Docker version:

```
export MACHINE_DOCKER_INSTALL_URL='https://releases.rancher.com/install-docker/17.12.sh'
```



To list the available files use the Google Cloud Storage API:  
<https://releases.rancher.com/?delimiter=&prefix=install-docker>

**Reasonable parameters for FIWARE Lab.** A working environment for the **Spain2** node of the **Fiware Lab**<sup>16</sup>:

```
export MACHINE_DRIVER='fiwarelab'
export ENGINE_OPT='mtu=1400'
export FIWARELAB_DOMAIN_NAME='default'
export FIWARELAB_SEC_GROUPS='rancher-cluster,external'
export FIWARELAB_NET_NAME='node-int-net-01'
export FIWARELAB_FLAVOR_NAME='m1.medium'
export FIWARELAB_IMAGE_NAME='base_ubuntu_16.04'
export FIWARELAB_SSH_USER='ubuntu'
export FIWARELAB_FLOATINGIP_POOL='public-ext-net-01'
```

➔ Export OpenStack variables to be seen by FIWARE Lab driver,

note that we are forced to do fancy stuff because of partial support for values with spaces by env and set:

```
NAMES_VALUES=$(
  env | \
  grep '^OS_' | \
  sed -e "s:==:'" -e "s:$:'" `: # add quotes in dumb way` | \
  sed -e 's/^OS_/FIWARELAB/' `: # rename the variables` | \
  tr '\n' ' '
)
```

```
eval export "${NAMES_VALUES}"
unset NAMES_VALUES
```

```
# If you have multiple regions active, you need to specify one, in
# order to not get the multiple possible endpoint match error
export FIWARELAB_REGION="Spain2"
```

```
set | egrep '^(OS|FIWARELAB)_'
```

**Reasonable parameters for a generic OpenStack cloud:**

```
export MACHINE_DRIVER='openstack'
export OPENSTACK_NETWORK_NAME='vlab'
export OPENSTACK_FLAVOR_NAME='m1.medium'
export OPENSTACK_IMAGE_NAME='GNU/Linux Ubuntu Server 16.04 x86_64 (Default
user: ubuntu)'
export OPENSTACK_SSH_USER='ubuntu'
export OPENSTACK_SEC_GROUPS='rancher-cluster,external'
export OPENSTACK_FLOATINGIP_POOL='vlab_external'
```

➔ Export OpenStack variables to be seen by Rancher (see Rancher bug [#7647](https://github.com/rancher/rancher/issues/7647)<sup>17</sup>), note that we

<sup>16</sup> <https://cloud.lab.fiware.org/>

<sup>17</sup> <https://github.com/rancher/rancher/issues/7647>

are forced to do fancy stuff because of partial support for values with spaces by env and set:

```
NAMES_VALUES=$(
  env | \
  grep '^OS_' | \
  sed -e "s:=:='" -e "s:$:'" `: # add quotes in dumb way` | \
  sed -e 's/^OS_/OPENSTACK_/' `: # rename the variables` |
  tr '\n' ' '
)

eval export "${NAMES_VALUES}"
unset NAMES_VALUES
```

```
# If you have multiple regions active, you need to specify one, in
# order to not get the multiple possible endpoint match error
export OPENSTACK_REGION="Spain2"
```

```
set | egrep '^(OS|OPENSTACK)_'
```

### 3.15.2 Create and access hosts

Before creating an host, double check the configured Rancher environment with **rancher config**

➔ Create host:

```
rancher hosts create rancher-node-01
```

➔ List the hosts status

```
rancher hosts ls
```

➔ Host, once up, is accessible via:

```
rancher ssh rancher-node-01
```

➔ NOTE: the rancher CLI has some minor annoyances. The **rm** subcommand is not scoped like the **create** one, so you need to issue the command as **rancher rm \$HOSTID** instead of using the more human friendly name.

After adding at least 3 nodes (you can change the number by following The Number of Swarm Managers), you should wait a few minutes before being able to access the cluster, in order to give it the time to startup.

Now you are ready to follow the detailed guide to DEPLOY SMARTSDK RECIPES ON SMARTSDK PLATFORM.

## 4 SMARTSDK PLATFORM ADVANCED USAGE

---

### 4.1 User management integrated with FIWARE Lab OAuth

By using a custom <sup>18</sup> build <sup>19</sup> of Rancher it is possible to use the OAuth authentication supplied by the FIWARE Lab. The use of the FIWARE Lab OAuth endpoint simplifies the user management on the Platform and offers an integrated user experience. This component is developed outside SmartSDK and is documented here for completeness. In the SmartSDK project we updated the component to work with the latest Rancher version supported.

---

<sup>18</sup> <https://github.com/smartsdk/rancher>

<sup>19</sup> <https://hub.docker.com/r/smartsdk/platform-manager/>

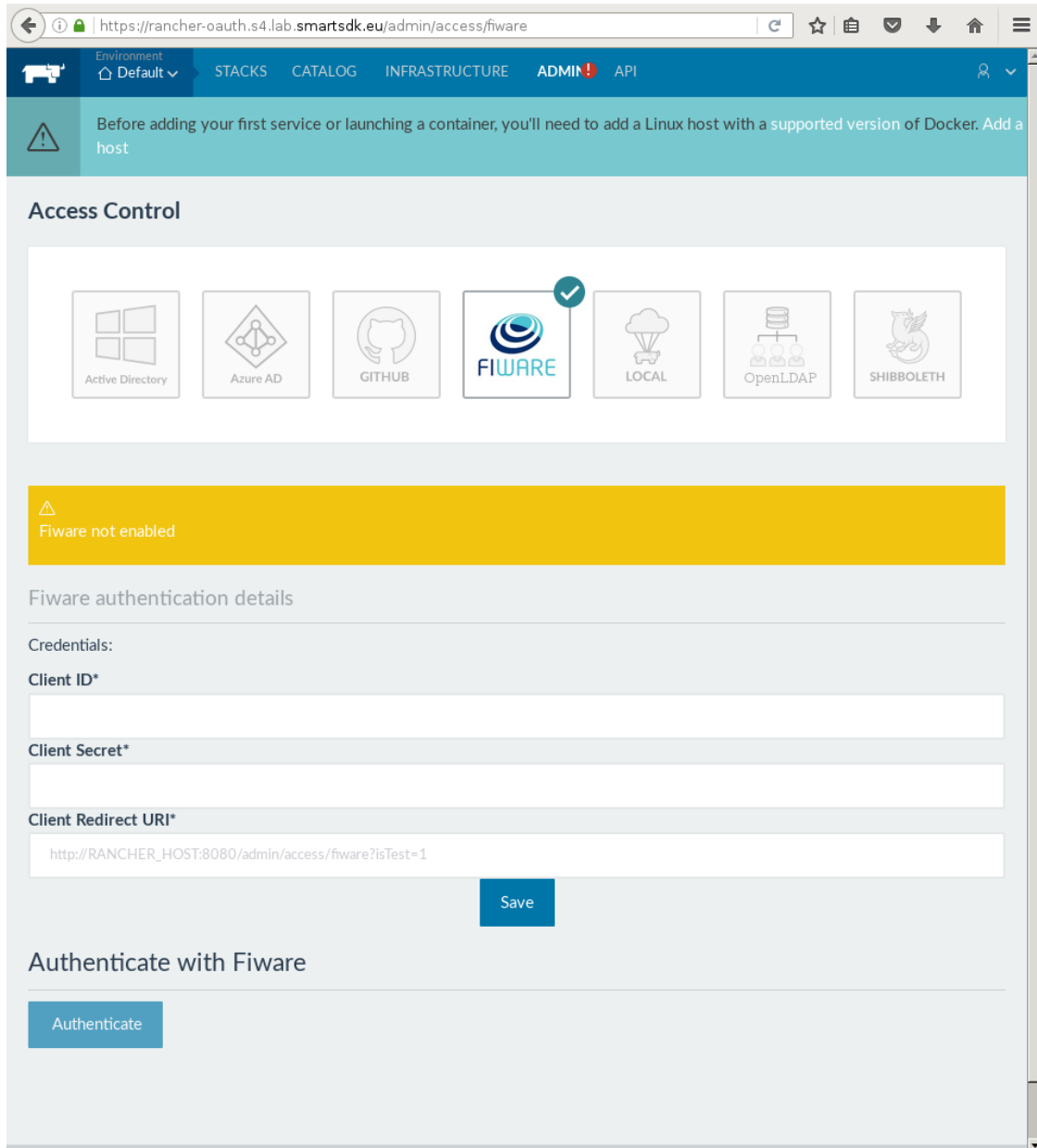


Figure 39: Screenshot of the Access control Setup for FIWARE Lab OAuth

## 4.2 Enable the FIWARE Lab OAuth

Unfortunately, to our knowledge, the FIWARE Lab OAuth does not support multiple source and callback URLs, which are required to register the application by using the Rancher web interface. However, it is possible to complete the registration by using the Rancher APIs by following the procedure described next.

➔ Remember the SmartSdk Platform Manager URL:

**BASE\_URL="https://platform-manager.smartsdk.eu"**

➔ In the [FIWARE Lab](https://account.lab.fiware.org/)<sup>20</sup> create a new application.

Assuming the URL of the SmartSDK Platform Manger is **\${BASE\_URL}**, the mandatory data are:

- URL: **\${BASE\_URL}/login**
- Callback URL: **\${BASE\_URL}/**

If you use the wrong URLs, the FIWARE Lab interface will show a warning, then an error and will not let you complete the login procedure.

Take note of the generated credentials “OAuth2 Credentials”:

- Client ID
- Client Secret

Put the values in the following variables in order to use the subsequent code snippets:

```
CLIENT_ID=REPLACE_VALUE
CLIENT_SECRET=REPLACE_VALUE
```

➔ Install prerequisites:

```
sudo apt install --yes curl jq
```

➔ Prepare the json to create new admin account keys:

```
cat <<EOD > post-apikey-data
{
  "type": "apikey",
  "accountId": "1a1",
  "name": "admin-api-name",
  "description": "admin-api-description",
  "created": null,
  "kind": null,
  "removeTime": null,
  "removed": null,
  "uuid": null
}
EOD
```

➔ Get the response and extract the Access Key (username) and Secret Key (password):

```
API_PATH="v2-beta/accounts/${ACCOUNT_ID}"
API_URL="${BASE_URL}/${API_PATH}"
RESPONSE="$(curl -s \
  "${API_URL}"
  -H 'Accept: application/json' \
  -H 'Content-Type: application/json' \
  -d @post-apikey-data)"
```

```
CATTLE_ACCESS_KEY="$(printf '%s' "${RESPONSE}" | jq --raw-output
```

<sup>20</sup> <https://account.lab.fiware.org/>

```
.publicValue)"
CATTLE_SECRET_KEY="$(printf '%s' "${RESPONSE}" | jq --raw-output
.secretValue)"
```

```
echo $CATTLE_ACCESS_KEY
echo $CATTLE_SECRET_KEY
echo $RESPONSE
```

- We inferred that the first person that login with the FIWARE OAuth will get the Account Id value of **1a7**:

```
ACCOUNT_ID=1a7
```

- Prepare the json for the access control with FIWARE Lab:

```
API_PATH="v1-auth/config"
API_URL="${BASE_URL}/${API_PATH}"
REDIRECT_URI="${BASE_URL}/"

cat <<EOD > post-data-oauth
{
  "type":"config",
  "provider":"fiwareconfig",
  "enabled":true,
  "accessMode":"unrestricted",
  "allowedIdentities":[],
  "fiwareconfig": {
    "clientId": "${CLIENT_ID}",
    "clientSecret": "${CLIENT_SECRET}",
    "redirecturi": "${REDIRECT_URI}"
  }
}
EOD
```

- Enable the access control with FIWARE Lab:

```
RESPONSE="$(curl -s \
-H "Content-Type: application/json" \
-d @post-data-oauth \
"${API_URL}")"
echo $RESPONSE | jq .
```

- Now the access control is enabled and it is possible to interact with Rancher only after authenticating with the admin API keys or by using the FIWARE Lab.
- Login using the browser to confirm that the FIWARE Lab authentication works. Please note that it is normal that the FIWARE Lab takes 10-15 seconds to reply in each step.
- Now, using the admin account keys we promote the first user as an admin:

```
API_PATH="v2-beta/accounts/${ACCOUNT_ID}"
API_URL="${BASE_URL}/${API_PATH}"
RESPONSE="$(curl -s \
-u "${CATTLE_ACCESS_KEY}:${CATTLE_SECRET_KEY}" \
-X PUT \
-H 'Accept: application/json' \
-H 'Content-Type: application/json' \
```

```
-d '{"kind": "admin"}' \
"${API_URL}"
)"
echo $RESPONSE | jq .
```

➔ Using the web interface logout and login to see the new “ADMIN” tab.

### 4.3 Machine driver and User Interface Plugin for FIWARE Lab Nodes

The default graphical user interface of SmartSDK Platform Manager requires filling about 20 fields in order to start a docker swarm environment on a generic OpenStack project. The ICCLab (Cloud Computing Lab) at ZHAW Zurich University of Applied Sciences developed a plugin to simplify the configuration, available on [on github](#)<sup>21</sup>. This component was developed outside SmartSDK project. In the SmartSDK project we [updated](#)<sup>22</sup> the component to work with the latest supported Rancher version.

### 4.4 Using a VPN for overcoming NAT issues

It is possible to create an environment with Rancher agents that are not associated to unique public IPs (i.e. connecting to remote Rancher server from a natted network).

In order to satisfy the Rancher requirement (every agent need to have a different IP) we will set up a VPN.

This unfortunately can not be easily automated with Rancher machine drivers.

The overall procedure is the following:

- ➔ Install a VPN server in the same subnet of the rancher-master host (o even on the same host of the rancher-master). This host must be reachable from all the other hosts (rancher-master and rancher-agents).
- ➔ Start the VPN service.
- ➔ Join the VPN with rancher-master.
- ➔ Join the VPN with any other host that will become a rancher-agent.
- ➔ Start the rancher-agents as custom hosts (most of the time you will specify the private VPN ip as the “public IP” in the terminology of the Rancher web interface, also known as **CATTLE\_AGENT\_IP**). Unfortunately the Rancher web interface makes some confusion about the requirement: what is labeled as “public” needs only to be reachable and unique, not really “public”.

One reasonable easy VPN service is **n2n**, for detailed information look at the [n2n howto](#)<sup>23</sup>.

The following snippet shows an example installation:

```
# Define some useful variables
SUPERNODE_IP=203.0.113.1
RANCHER_MASTER_IP_ON_VPN=192.0.2.1
```

<sup>21</sup> <https://github.com/icclab/ui-driver-fiwarelab>

<sup>22</sup> <https://github.com/smartsdk/ui-driver-fiwarelab>

<sup>23</sup> <http://blog.rot13.org/2011/10/n2n-connect-your-networks-using-p2p-vpn.html>

```

RANCHER_MASTER_PORT=443
MTU=1300
VPN_PORT=1194
# Infer RANCHER_HOST_ENV_TOKEN from the long command line interface
# from the rancher add host interface (it is the same for all the hosts)
RANCHER_HOST_ENV_TOKEN=
# install n2n
sudo apt install n2n
# start the server on port 1194
sudo supernode -l "${VPN_PORT}"
# The NODE_IP_ON_VPN must be different for each host and for the
# rancher-master the same of RANCHER_MASTER_IP
NODE_IP_ON_VPN=192.0.2.2
# Set up a long enough shared secret
SHARED_SECRET="REPLACE ME WITH A LONG ASCII TEXT"
# on the rancher master and on each rancher-agents node join the server
nohup sudo edge -c vpn4rancher -d vpn4rancher -k "${SHARED_SECRET}" \
    -l "${SUPERNODE_IP}:${VPN_PORT}" -M "${MTU}" -a "${NODE_IP_ON_VPN}" &
# on each node join the rancher-server (with a modified cut and paste
# from the rancher add host interface)
sudo docker run -e CATTLE_AGENT_IP="${NODE_IP_ON_VPN}" -d --privileged \
    -v /var/run/docker.sock:/var/run/docker.sock \
    -v /var/lib/rancher:/var/lib/rancher rancher/agent:v1.2.1 \

"http://${RANCHER_MASTER_IP}:${RANCHER_MASTER_PORT}/v1/scripts/${RANCHER_HOST_ENV_TOKEN}"

```

#### 4.5 Provision of Rancher hosts using custom hosts

This is the procedure to use when you are not able to use machine drivers, for example for lack of public IPs.

**“Add hosts to Rancher using custom hosts”** allows to specify which IP addresses hosts has to use to communicate with each other.

This is done by specifying the `CATTLE_AGENT_IP` environment variable when launching the Rancher/agent container.



## 5 DEPLOY SMARTSDK RECIPES ON SMARTSDK PLATFORM

Note that there are MTU issues with swarmkit in the FIWARE lab. See Advanced analysis of the issues related to non-standard MTU usage.

### 5.1 Deployment using docker stack deploy

When an environment is managed in docker swarm mode, the application deployment can be managed by passing Compose v3.x files to a swarm manager node.

The latest Rancher server (v1.6.15) doesn't provide an easy way to do this. A multi-step procedure follows.

1. Be sure to have docker 17.12-ce installed on your host.
2. Find a node labeled as manager:

```
rancher hosts
```

ID	HOSTNAME	STATE	CONTAINERS	IP
LABELS				
1h11	rancher-node-01.novalocal	active	14	
192.168.242.90	swarm=manager			
1h12	rancher-node-02.novalocal	active	12	
192.168.242.91	swarm=manager			
1h13	rancher-node-03.novalocal	active	12	
192.168.242.93	swarm=manager			
1h14	rancher-node-04.novalocal	active	12	
192.168.242.92				

3. Ask a manager to deploy the application:

```
rancher --host 1h11 docker stack deploy --compose-file docker-stack.yml testApp
```

Note that the deployed application stacks will not be seen nor managed by Rancher as stacks but only as independent containers.

Look at the examples in the next sections for a guidance in order to have an application deployed.

### 5.2 Test deployments

#### 5.2.1 Swarm deployment with replication (docker stack)

The following Compose file creates 3 replica containers, reachable from any of the swarm nodes.

The docker-stack.yml:

```
cat <<EOF > docker-stack.yml
version: '3'
services:
  whoami:
    image: jwilder/whoami
```

```

ports:
  - "80:8000"

deploy:
  replicas: 3
networks:
  default:
    driver: bridge
    driver_opts:
      com.docker.network.driver.mtu: ${DOCKER_MTU:-1500}
EOF

```

To launch the application follow the example:

```

# We need to set the variable for this recipe (default is 1500)
export DOCKER_MTU=1400
# Just a trick to find the first available manager
SWARM_MGR=$(rancher hosts ls | awk '/swarm=manager/ { print $1; exit}')
export SWARM_MGR
rancher --host "${SWARM_MGR}" docker stack deploy --compose-file docker-
stack.yml whoami

```

### 5.2.2 Deploy smartsdk-recipes using the CLI

All the recipes developed available in the repository <https://github.com/smartsdk/smartsdk-recipes> are deployable both using the graphical user interface as shown in PLATFORM-MANAGER USAGE and by using the CLI as shown in the previous sub-section.

## 6 SMARTSDK PLATFORM INSTALLATION

We will start with a host with a docker installation of the version 17.12-ce and deploy the **rancher-master**.

For your convenience see also how to Add the docker group to the current user.

Set some useful variables

You want to set the **RANCHER\_HOSTNAME** to a fully qualified host name. If it is reachable from the Internet it will get a proper SSL certificate signed by [Letsencrypt](https://letsencrypt.org/)<sup>24</sup>. For testing purposes you can use <http://xip.io/> or <http://nip.io/> with a public floating IP.

Note that by using those test services you may not be able to get certificates because of a rate/total certificate limit on Letsencrypt:

*ACME server returned an error: urn:acme:error:rateLimited :: There were too many requests of a given type :: Error creating new cert :: Too many certificates already issued for: nip.io*

Set Rancher version, server host name, email where to send certificate renewal alerts and MTU:

```
# Auto devel build
export RANCHER_IMAGE="smartsdk/platform-manager-auto-build"
# Manual build
export RANCHER_IMAGE="smartsdk/platform-manager"
# Auto devel version
export RANCHER_VERSION="v1.6-smartsdk-dev"
# Final release version
export RANCHER_VERSION="v1.6.15-smartsdk"
export RENEWAL_EMAIL="user@smartsdk.eu"
export RANCHER_HOSTNAME="platform-manager.smartsdk.eu"
export DOCKER_MTU="1400"
```

```
# To use with the browser
export RANCHER_URL="https://${RANCHER_HOSTNAME}"
```

Create the Compose file to deploy a Rancher server accessible through a TLS termination proxy:

```
cat <<EOF > docker-stack.yml
version: '3'
services:
  nginx-proxy:
    image: jwilder/nginx-proxy
    environment:
      - DEFAULT_HOST=${RANCHER_HOSTNAME}
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
      - /srv/conf/nginx-proxy/certs:/etc/nginx/certs:ro
      - /srv/conf/nginx-proxy/vhost.d:/etc/nginx/vhost.d
```

<sup>24</sup> <https://letsencrypt.org/>

```

- /srv/conf/nginx-proxy/httpdocs:/usr/share/nginx/html
labels:
- "com.github.jrcs.letsencrypt_nginx_proxy_companion.nginx_proxy"

lets:
  image: jrcs/letsencrypt-nginx-proxy-companion
  environment:
    - NGINX_PROXY_CONTAINER=nginx-proxy
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock:ro
    - /srv/conf/nginx-proxy/certs:/etc/nginx/certs:rw
    - /srv/conf/nginx-proxy/vhost.d:/etc/nginx/vhost.d
    - /srv/conf/nginx-proxy/httpdocs:/usr/share/nginx/html
  depends_on:
    - nginx-proxy

rancher:
  image: ${RANCHER_IMAGE}:${RANCHER_VERSION}
  ports:
    - "8080:8080"
  environment:
    - VIRTUAL_HOST=${RANCHER_HOSTNAME}
    - VIRTUAL_PORT=8080
    - LETSENCRYPT_HOST=${RANCHER_HOSTNAME}
    - LETSENCRYPT_EMAIL=${RENEWAL_EMAIL}
    - LETSENCTYPT_TEST=false
  volumes:
    - /srv/db-data:/var/lib/mysql:rw

networks:
  default:
    driver_opts:
      com.docker.network.driver.mtu: ${DOCKER_MTU:-1400}
EOF

Enable swarm mode on the node:

docker swarm init

Launch server and proxy:

docker stack deploy --compose-file docker-stack.yml platform-manager

Look at the log:

RANCHER_ID=$(docker service ls --quiet --filter name=platform-
manager_rancher)
docker service logs "${RANCHER_ID}"

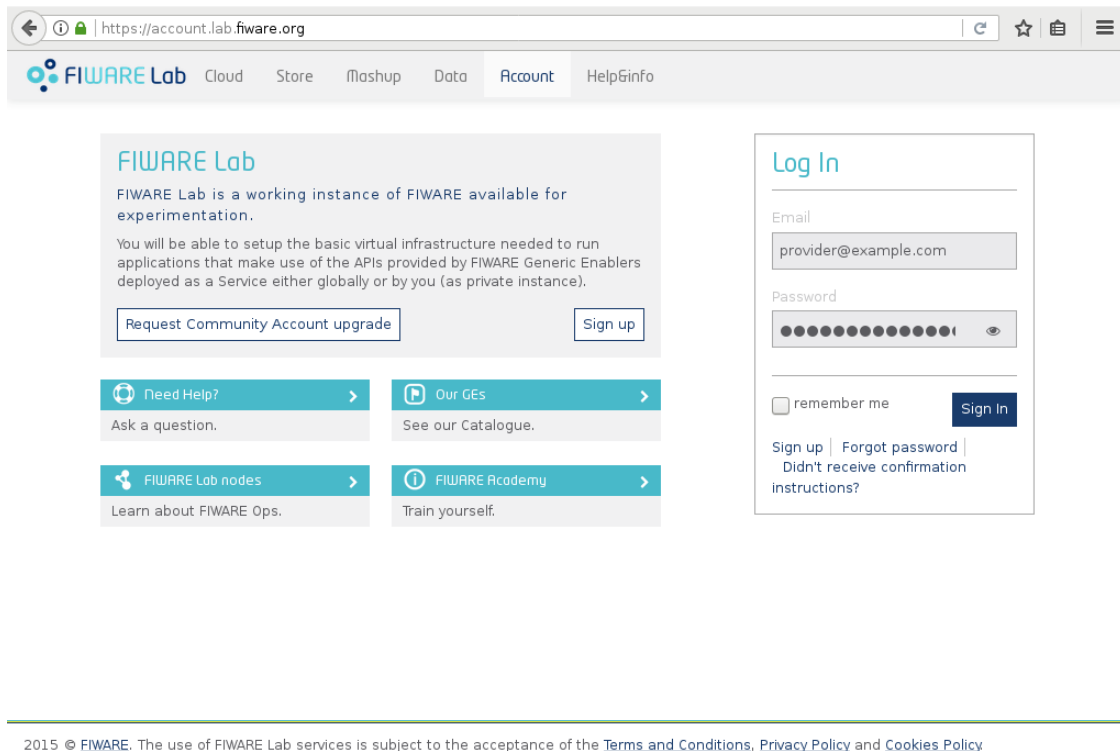
```

## 6.1 Register the SmartSDK Platform on FIWARE Lab

In order to authenticate on your instance of the SmartSDK Platform Manager using the FIWARE

OAuth2 service, you will need to perform the following steps.

First, login to the [account site of FIWARE Lab](https://account.lab.fiware.org)<sup>25</sup>.

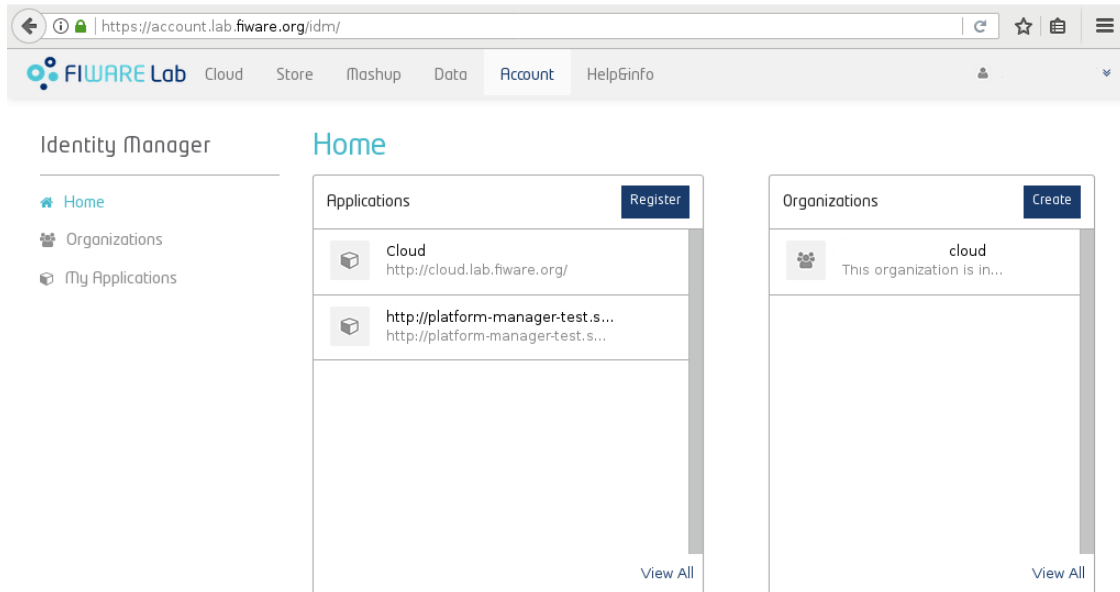


2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#).

*Figure 40: Login on the account of FIWARE Lab*

You will see a summary page:

<sup>25</sup> <https://account.lab.fiware.org/>



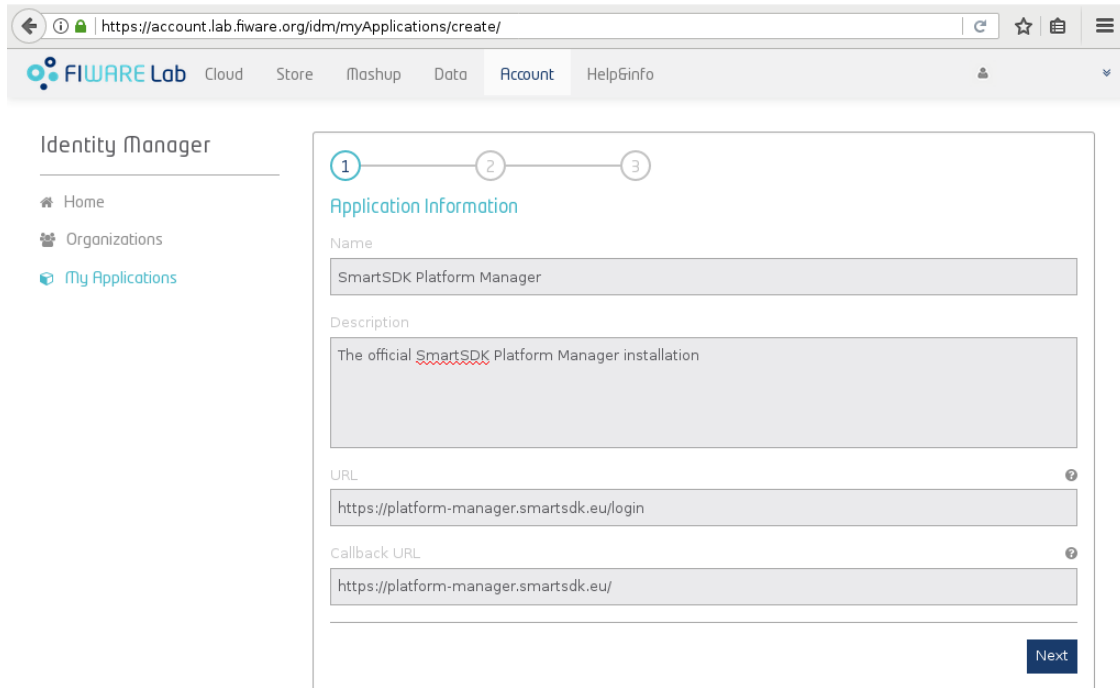
2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

*Figure 41: Summary on the account of FIWARE Lab*

Click on the “Register” button you can find on top of the “Applications” panel.

You will be redirected to the Application Creation Wizard. Insert the application information.

Note: the **URL** must be the base URL of your instance plus **login**, and the **Callback URL** must be the base URL of your instance plus a trailing **/**.



Identity Manager

- Home
- Organizations
- My Applications

Application Information

Name

SmartSDK Platform Manager

Description

The official SmartSDK Platform Manager installation

URL

https://platform-manager.smartsdk.eu/login

Callback URL

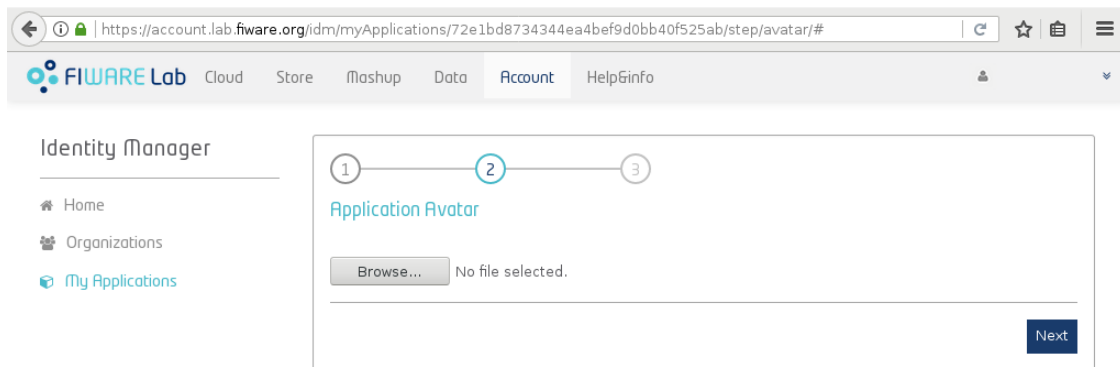
https://platform-manager.smartsdk.eu/

Next

2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

*Figure 42: Application Info on FIWARE Lab*

Then upload the (optional) logo.



Identity Manager

- Home
- Organizations
- My Applications

Application Avatar

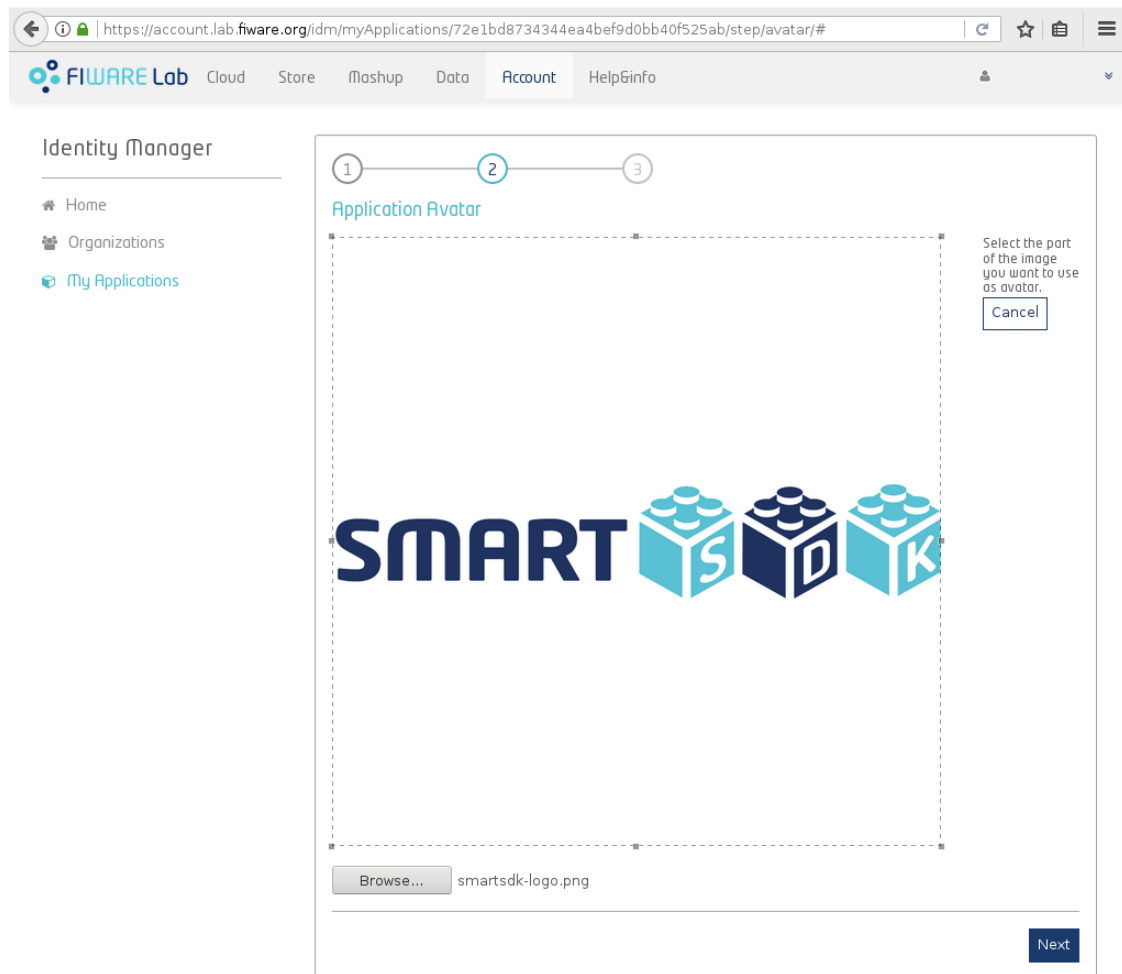
Browse... No file selected.

Next

2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

*Figure 43: Upload optional logo on FIWARE Lab*

You can resize it during the upload procedure.

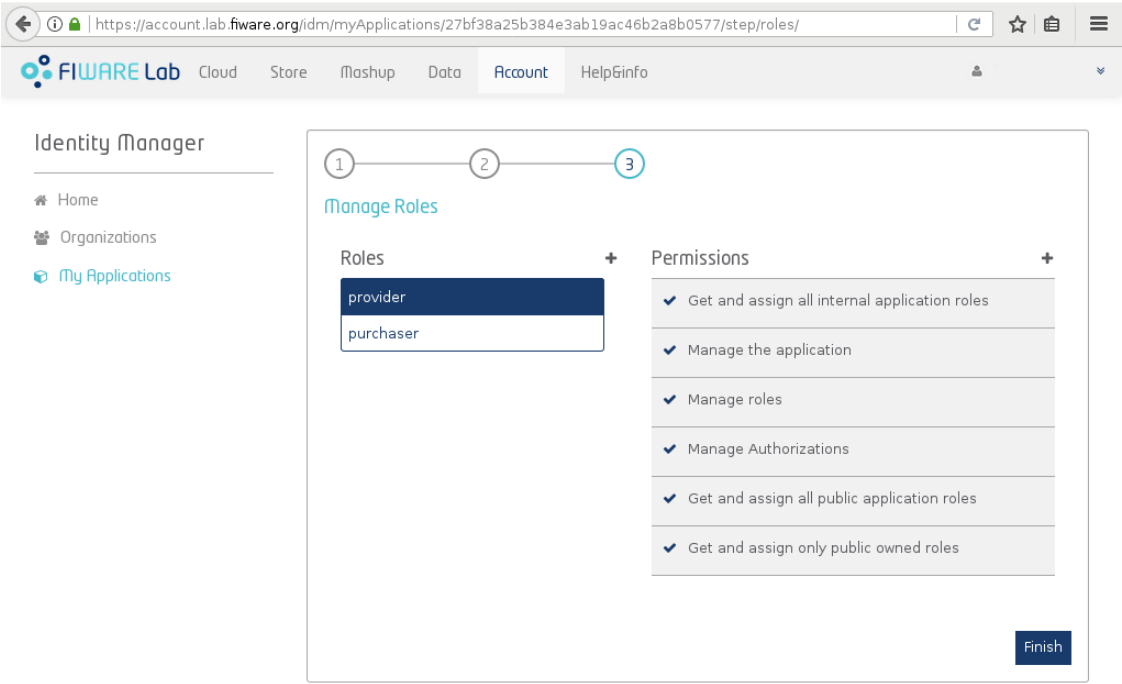


2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#).

*Figure 44: Resize logo on FIWARE Lab*

After the logo upload select the **provider Roles** for your instance.

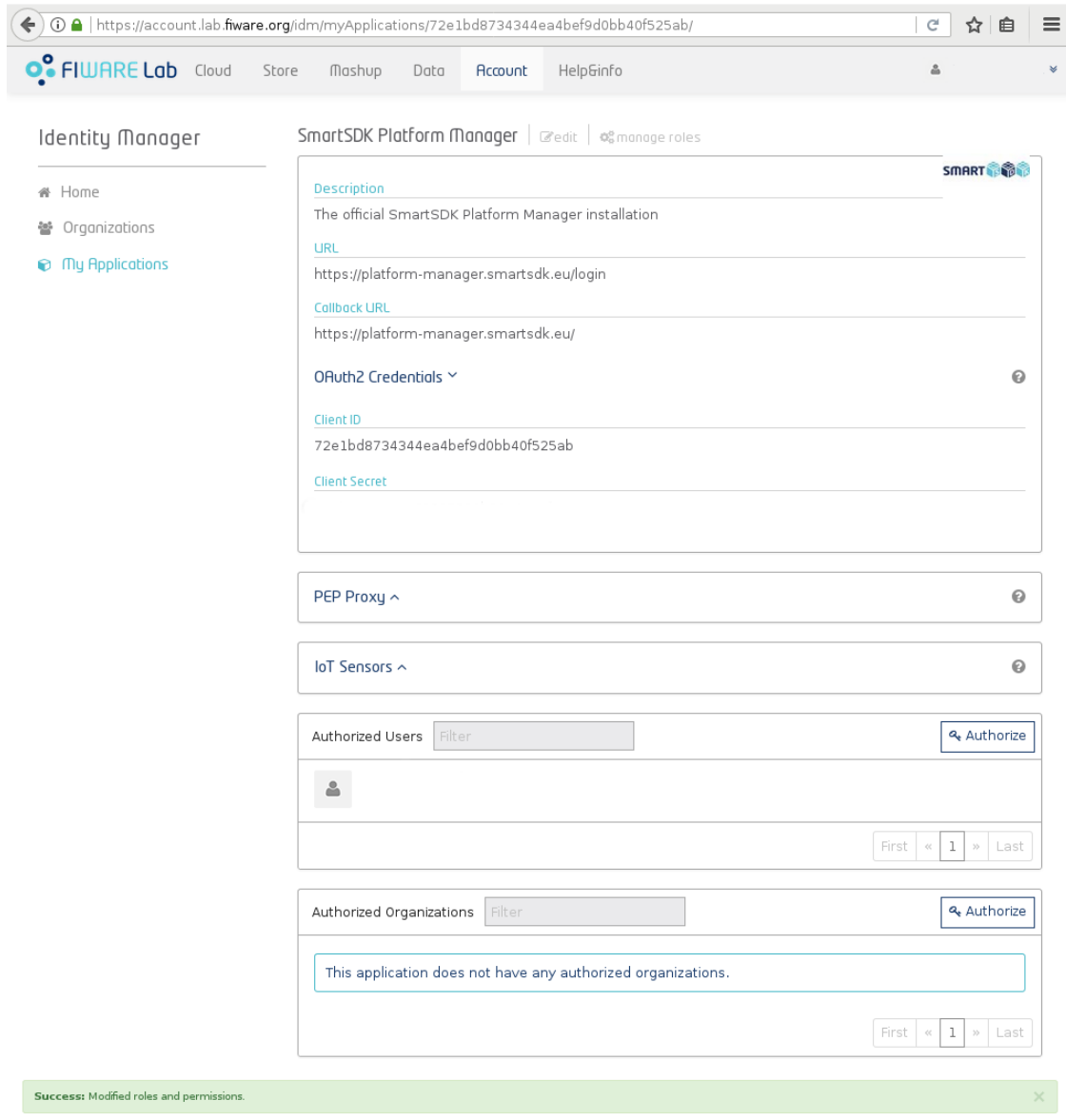




2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

Figure 45: Select app Roles on FIWARE Lab

At the end take note of the OAuth2 Credentials: **Client ID** and **Client Secret**.



The screenshot shows the FIWARE Lab account page for the SmartSDK Platform Manager. The browser address bar shows the URL: <https://account.lab.fiware.org/idm/myApplications/72e1bd8734344ea4bef9d0bb40f525ab/>. The page has a navigation bar with links: Cloud, Store, Mashup, Data, Account, and Help & Info. The main content area is titled "SmartSDK Platform Manager" and includes a "manage roles" link. The configuration details are as follows:

- Description:** The official SmartSDK Platform Manager installation
- URL:** <https://platform-manager.smartsdk.eu/login>
- Callback URL:** <https://platform-manager.smartsdk.eu/>
- OAuth2 Credentials:**
  - Client ID:** 72e1bd8734344ea4bef9d0bb40f525ab
  - Client Secret:** (empty field)
- PEP Proxy:** (empty field)
- IoT Sensors:** (empty field)
- Authorized Users:** (empty list with a "Filter" input and an "Authorize" button)
- Authorized Organizations:** (empty list with a "Filter" input and an "Authorize" button)

A success message at the bottom states: "Success: Modified roles and permissions."

2015 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the [Terms and Conditions](#), [Privacy Policy](#) and [Cookies Policy](#)

Figure 46: Recap of app on FIWARE Lab

The OAuth2 Credentials **Client ID** and **Client Secret** need to be passed to the instance of the Platform App in order to register it with the FIWARE Lab.

## 6.2 Install and configure the FIWARE Lab Rancher UI driver

The SmartSDK Platform, in order to offer an easy way to start new hosts for the user environment using the FIWARE Lab, has to be configured properly.

Here we document a visual step-by-step guide tailored to our installation.

Got to the “Admin” tab and click on “Machine Drivers”.

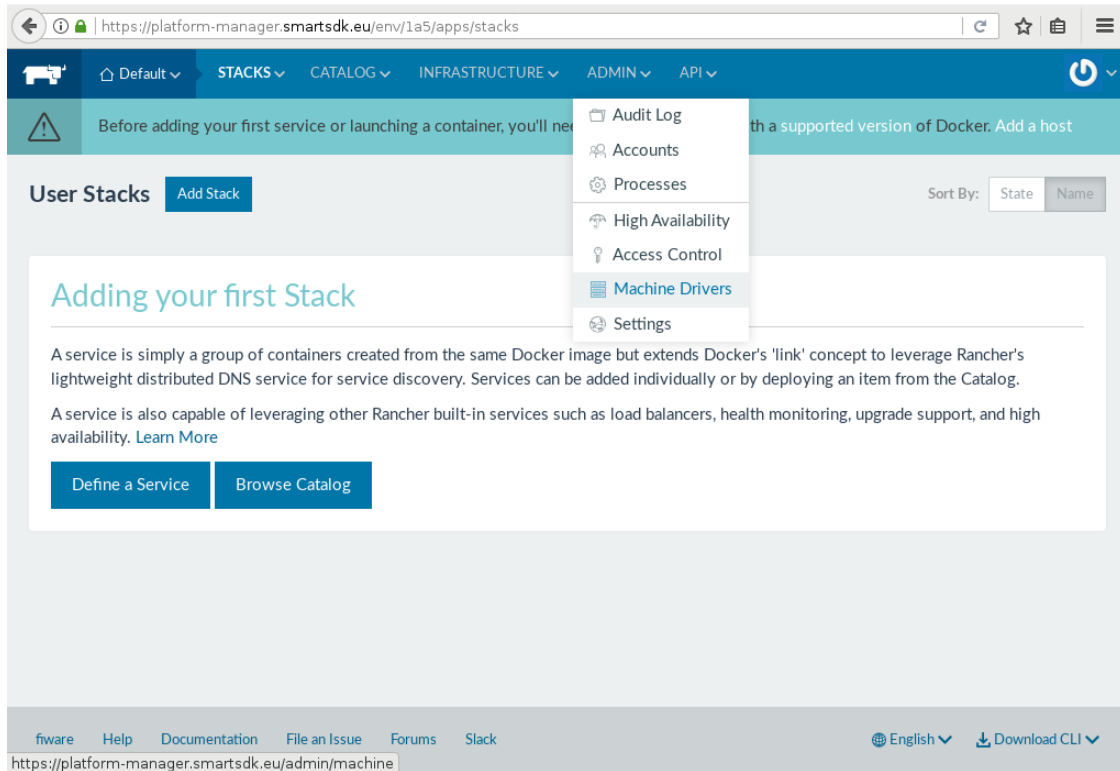


Figure 47: Machine drivers menu

The page displays the already pre-configured drivers. You can safely enable or disable the ones you may want to use. In order to add the FIWARE Lab Rancher UI driver click on the “Add Machine Driver” button.

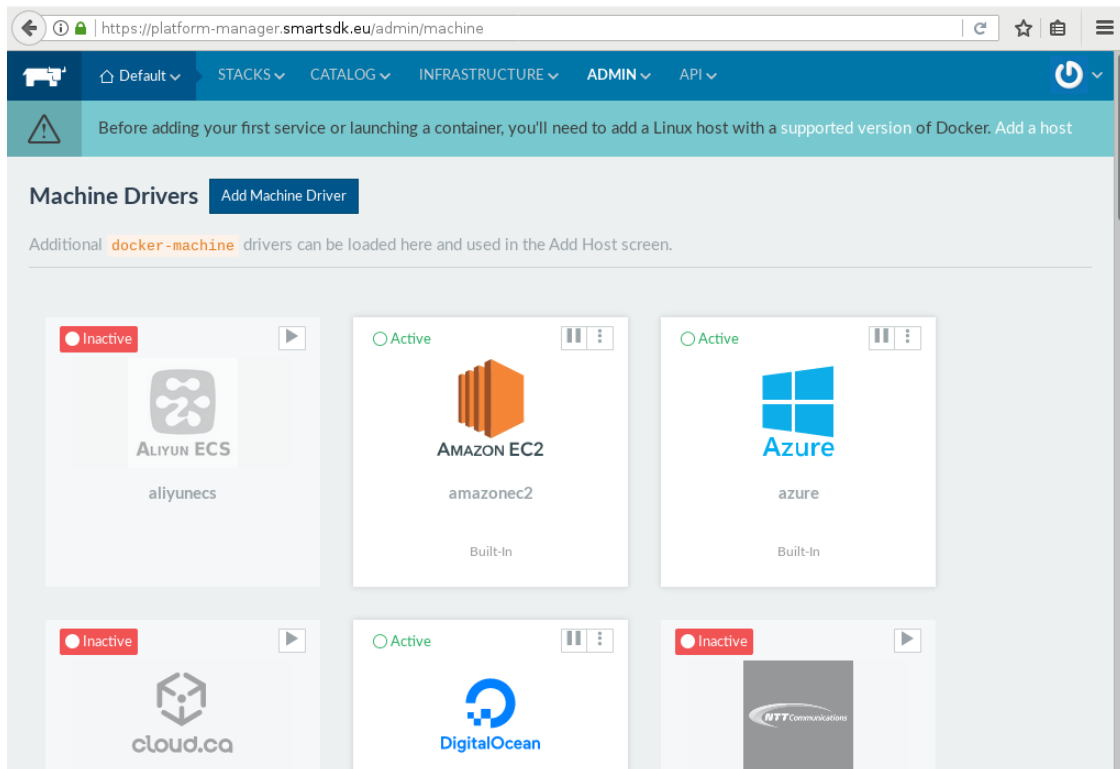
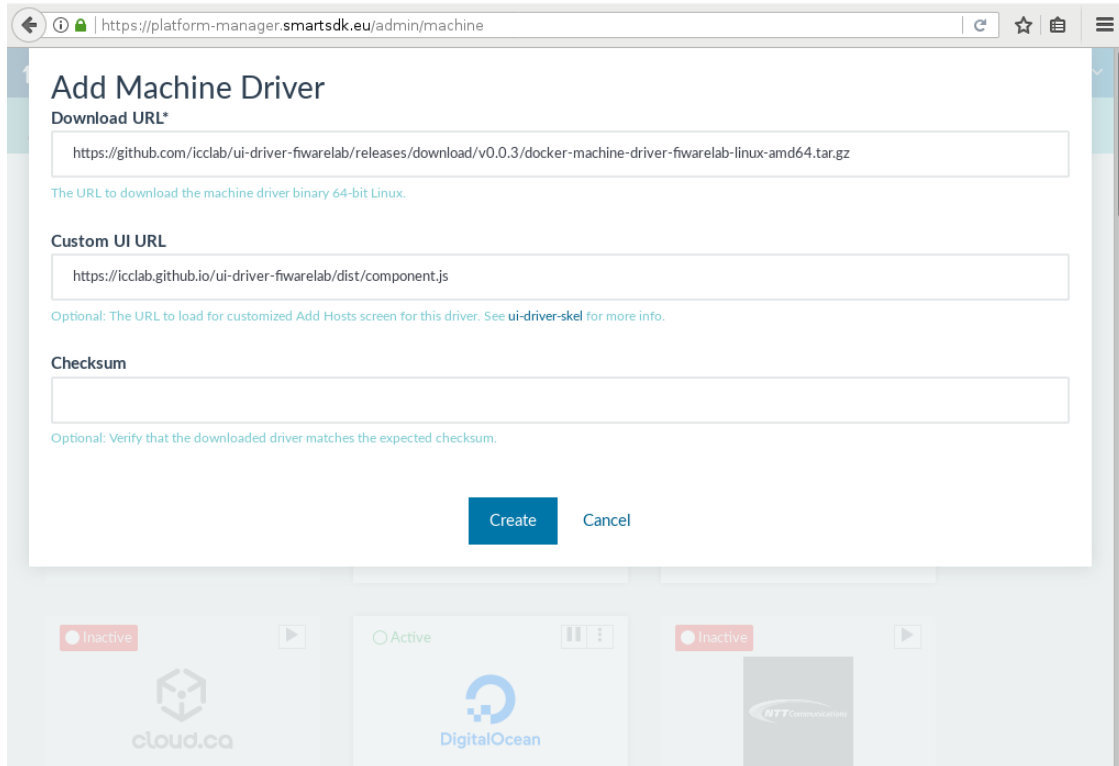



Figure 48: Enable the FIWARE Lab Rancher UI driver

Fill the required fields with the appropriate parameters. You will find the latest ones in the [documentation](#)<sup>26</sup>.



The screenshot shows a web browser window with the URL `https://platform-manager.smartsdk.eu/admin/machine`. The main heading is "Add Machine Driver". There are three input fields: "Download URL\*" containing `https://github.com/icclab/ui-driver-fiwarelab/releases/download/v0.0.3/docker-machine-driver-fiwarelab-linux-amd64.tar.gz`, "Custom UI URL" containing `https://icclab.github.io/ui-driver-fiwarelab/dist/component.js`, and "Checksum". Below each field is a small explanatory text: "The URL to download the machine driver binary 64-bit Linux.", "Optional: The URL to load for customized Add Hosts screen for this driver. See ui-driver-skel for more info.", and "Optional: Verify that the downloaded driver matches the expected checksum." At the bottom of the form are "Create" and "Cancel" buttons. Below the form, a list of existing machine drivers is visible, including "cloud.ca" (Inactive), "DigitalOcean" (Active), and "RTT Communications" (Inactive).

Figure 49: Add new Machine Driver

After the creation of the Driver you need to activate it by clicking the play  symbol.

<sup>26</sup> <https://github.com/smartsdk/ui-driver-fiwarelab>

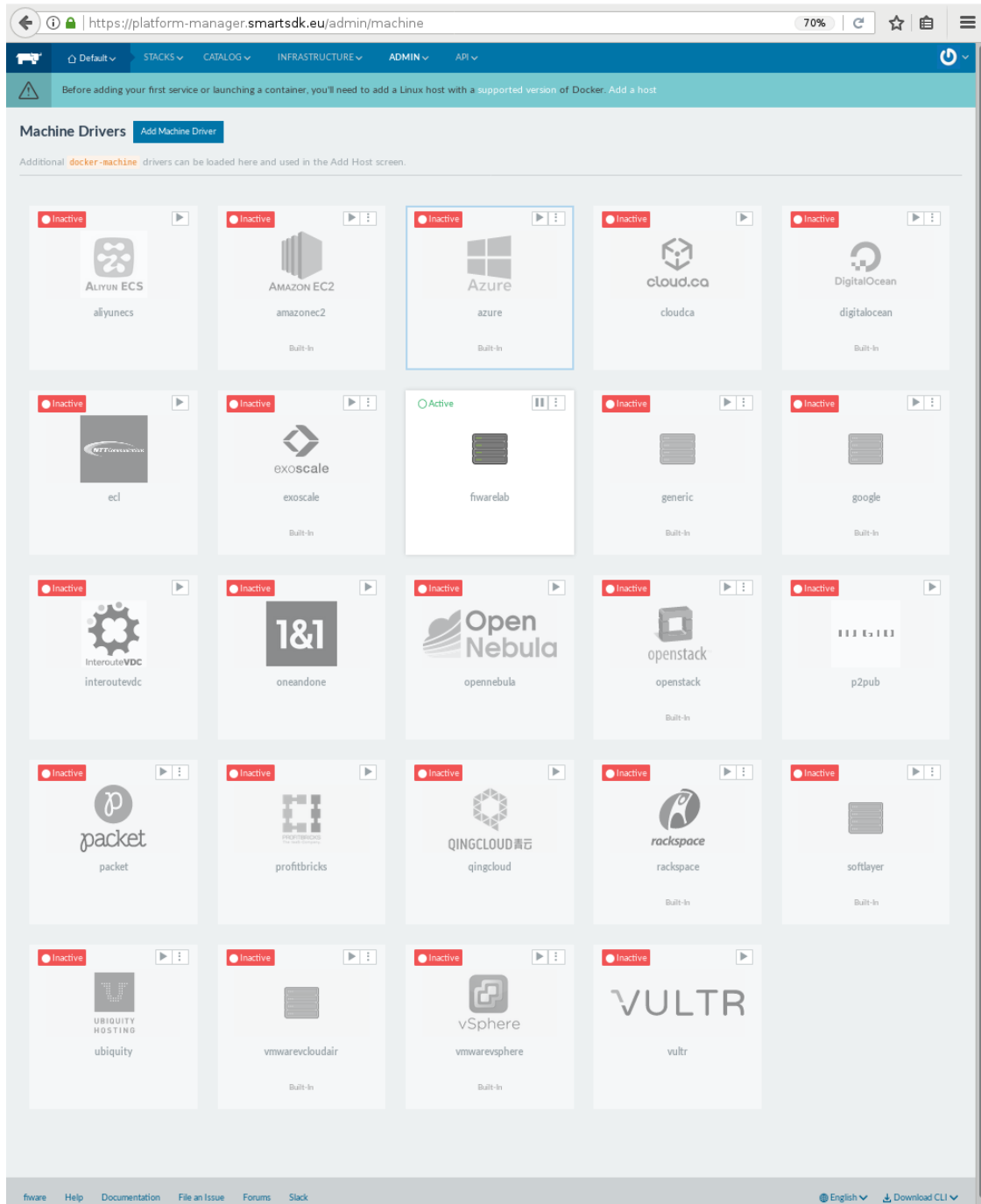


Figure 50: Driver list

Now you need to configure some advanced settings.

In the “Admin” tab click the “Settings” label.

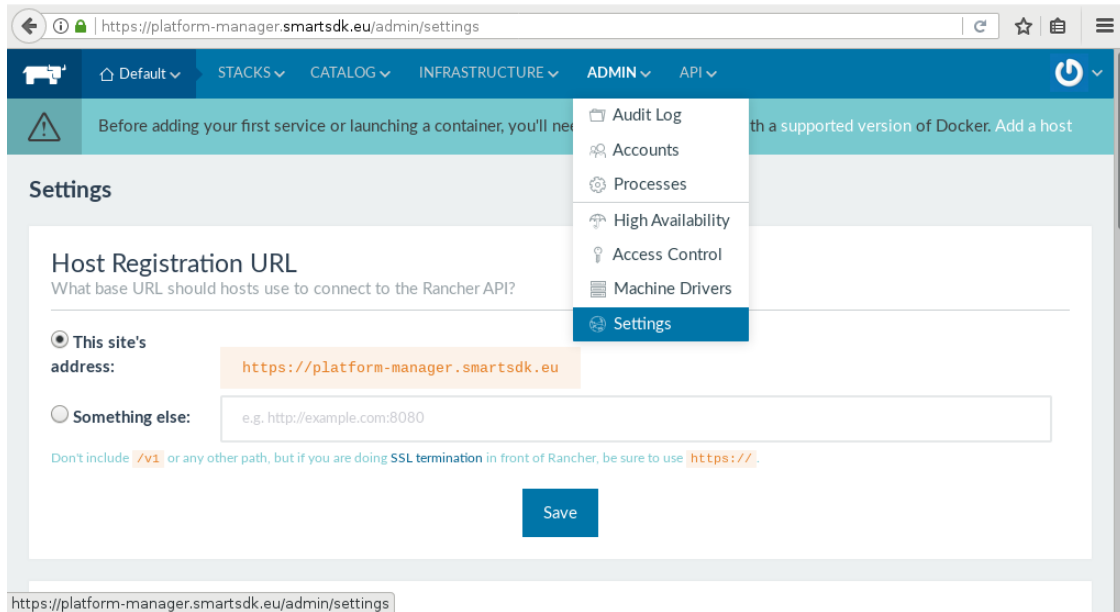


Figure 51: Admin Settings

Scroll down the page until you see the “Advanced Settings” section and click on the long disclaimer text.

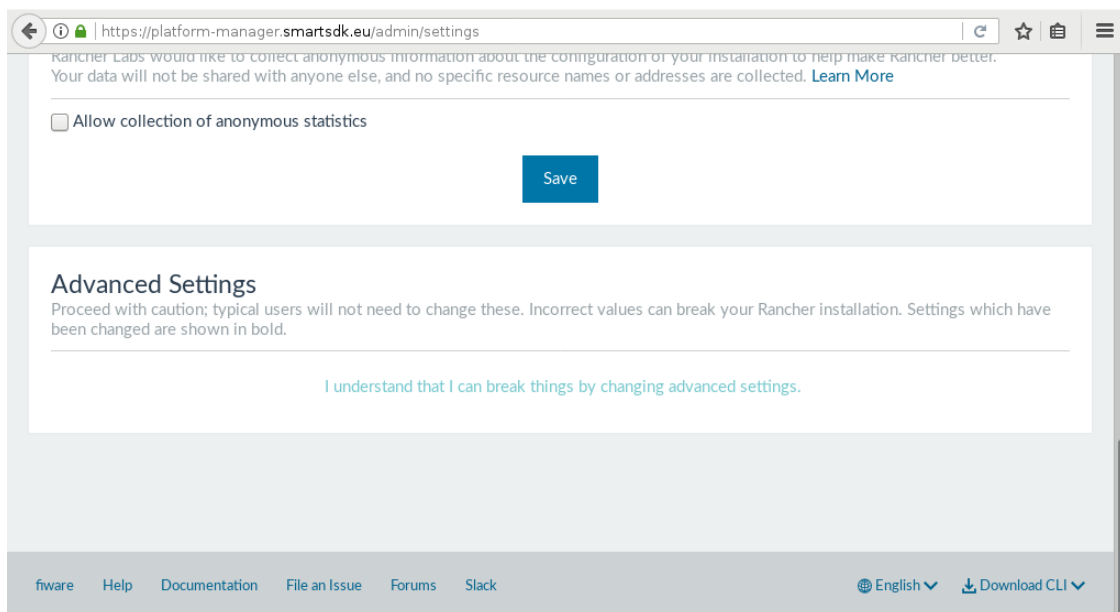


Figure 52: Admin advanced settings

You need to configure the Platform Manager so to proxy the API endpoints towards the OpenStack regions you want your users to have access.

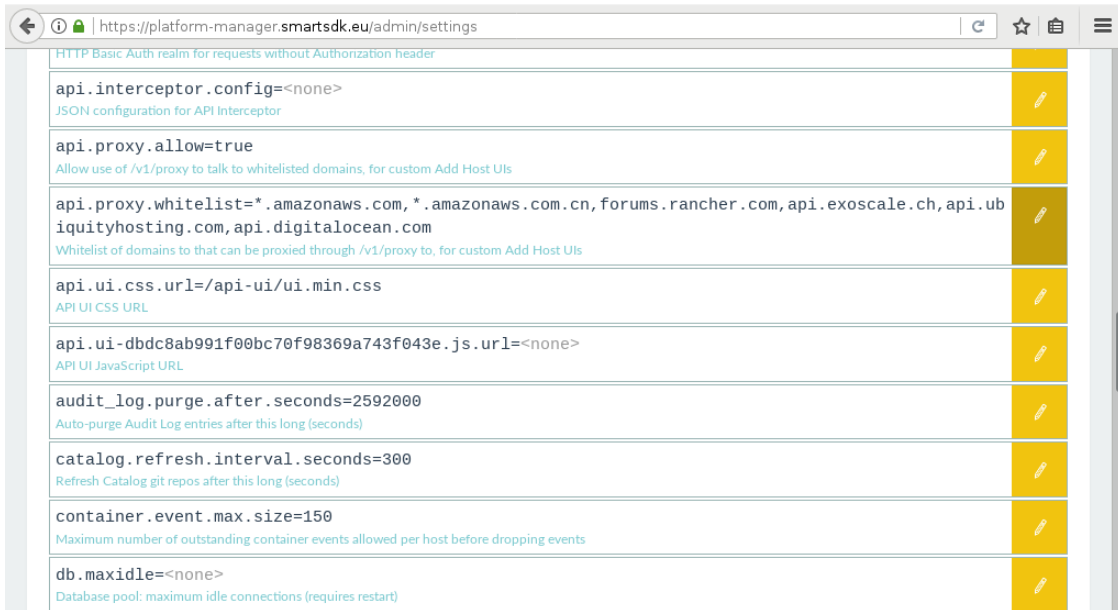


Figure 53: Admin Advanced Setting edit

For example, for the “Spain2” and “Mexico” regions, you need to set **api.proxy.whitelist** with the following addresses:  
**cloud.lab.fiware.org:4730,130.206.112.3:8774,130.206.112.3:9696.**

Please note that these addresses may change in the future. There is no authoritative list of the endpoints published. Usually you can discover this list by looking at the error of the browser during the “add host” procedure, detailed in [Add host\(s\)](#).

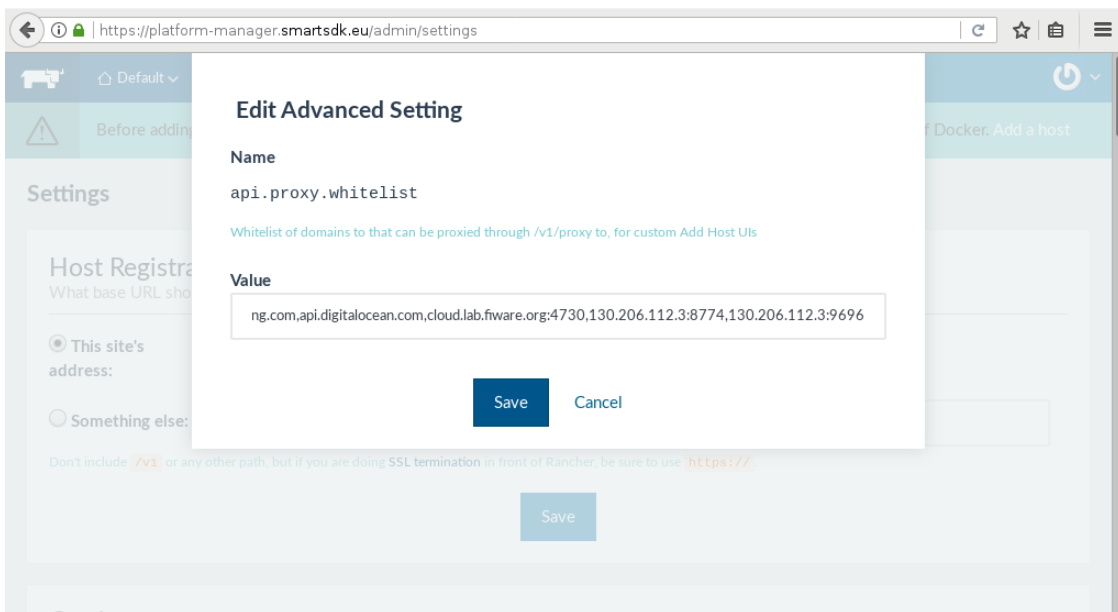


Figure 54: Setting api.proxy.whitelist

Now every user of the SmartSDK Platform can add new hosts to their environments using the just installed interface.

### 6.3 Edit Docker Swarm Settings

It is possible that your Docker Swarm Cluster will be deployed on hosts that have lower than the de-

facto standard MTU size of 1500 bytes.

This could cause unexpected connection problems (usually packets larger than the MTU will be silently dropped).

To overcome this problem configure the IPsec VPN to have an MTU value of 1400 bytes.

Then, on the Template Environment, select for editing the “Default Swarm template”

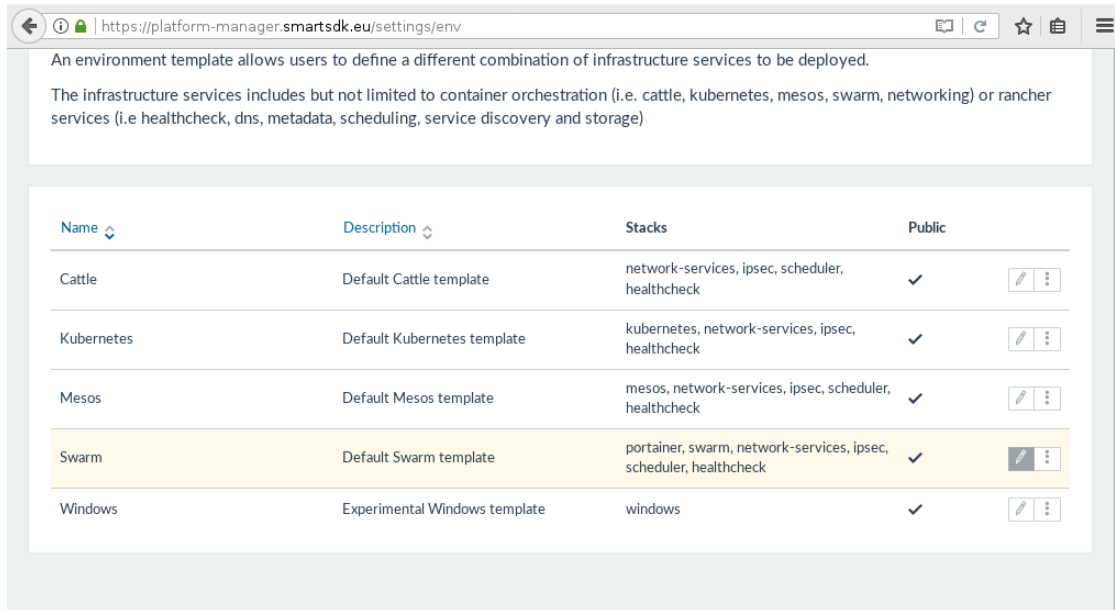


Figure 55: Select “Default Swarm template”

Click on the “Edit Config” button for “Rancher IPSEC”:

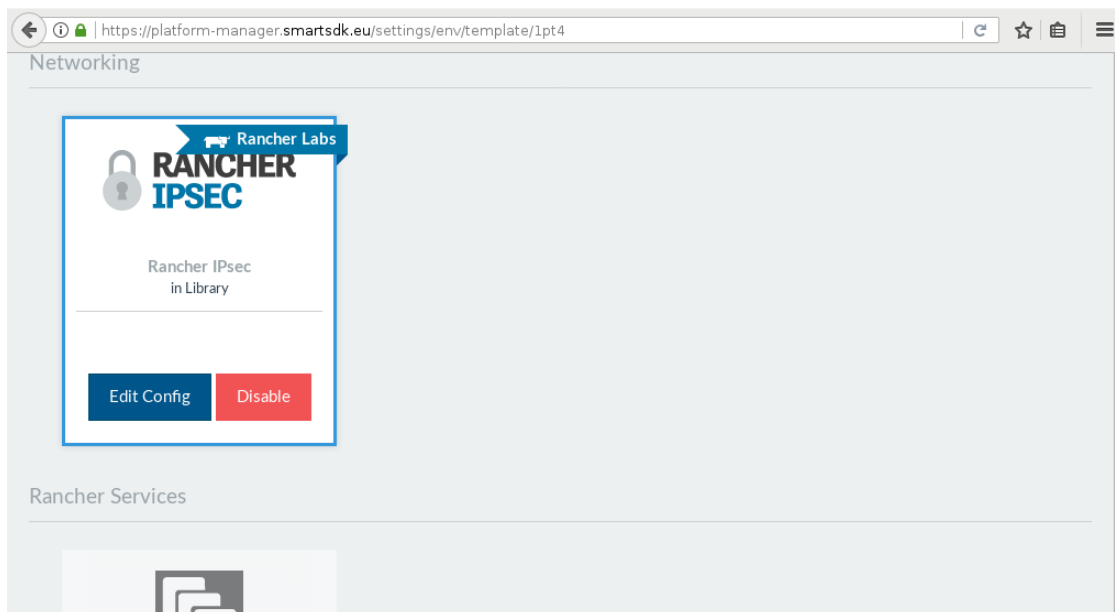


Figure 56: Edit config of Rancher IPSEC

Set the value of “MTU for the network” to 1400, and save it by clicking “Configure”.



ipsec

Description

Configuration Options

Docker Bridge\*

docker0

Name of Docker Bridge. Default is 'docker0'

Subnet\*

10.42.0.0/16

The subnet to use for the managed IPSEC network.

MTU for the network\*

1400

Adjust the MTU for the network, according to your needs. Ex: GCE(1460), AWS(1500), etc

Enable Debug Logs\*

☐ True

☒ False

This will enable very verbose debug logs.

Configure

Cancel

Figure 57: End of Template configuration

## 7 CONCLUSION

---

The SmartSDK Platform Manager is already working, installed and properly configured on a testing project on the FIWARE Lab.

At the current state of the project is possible to use the SmartSDK Platform Manager to deploy SmartSDK recipes.

All the source code newly developed or forked and adapted is hosted under the [SmartSDK project on github](https://github.com/smartsdk)<sup>27</sup>.

- ➔ A brief description of each repository follows.
- ➔ <https://github.com/smartsdk/rancher> Custom build of Rancher with settings for using FIWARE Lab auth and templates modules. The docker image is automatically build and published on docker hub: <https://hub.docker.com/r/smartsdk/platform-manager-auto-build/>. The final release is published on docker hub: <https://hub.docker.com/r/smartsdk/platform-manager/>
- ➔ <https://github.com/smartsdk/cattle> Custom build of Rancher cattle with settings for using FIWARE Lab auth.
- ➔ <https://github.com/smartsdk/ui> Custom build of Rancher ui with settings for using FIWARE Lab auth.
- ➔ <https://github.com/smartsdk/rancher-auth-service> Custom build of rancher-auth-service with settings for using FIWARE Lab auth.
- ➔ <https://github.com/smartsdk/guided-tour>: A short guide to use SmartSDK platform. The platform-manager documentation is build also on readthedocs.io: <https://guided-tour-smartsdk.readthedocs.io/en/latest/platform/swarmcluster/>.
- ➔ <https://github.com/smartsdk/guided-tour-builder> For a custom docker image to build the guided-tour. Published also on docker hub: <https://hub.docker.com/r/smartsdk/guided-tour-builder/>.
- ➔ <https://github.com/smartsdk/smartsdk-recipes> Contains recipes to use different FIWARE Generic Enablers to develop FIWARE-based applications.
- ➔ <https://github.com/smartsdk/docker-machine-driver-fiwarelab> The docker machine driver for FIWARE Lab, to be used by ui-driver-fiwarelab.
- ➔ <https://github.com/smartsdk/ui-driver-fiwarelab> The User Interface for the docker machine driver for FIWARE Lab.
- ➔ <https://github.com/smartsdk/fiwarelab-swarm-catalog> The custom catalog for the Rancher environment templates, includes the “Fiware Swarm”.
- ➔ <https://github.com/smartsdk/fiwarelab-machine-catalog> The custom catalog for the rancher machine driver, includes the “docker machine driver for FIWARE Lab”.

---

<sup>27</sup> <https://github.com/smartsdk>

## APPENDIX A

### A.1 How Docker Swarm networking works

Docker swarm allows to create overlay networks that connect containers on different swarm nodes.

The default swarm network driver uses VXLAN secured with point-to-point IPSEC tunnels to provide L2 networking among containers. IPsec requires that nodes can directly reach each other with UDP traffic (no natting).

### A.2 How exposing services through load balancers works

Docker swarm uses internal load balancers that expose ports on the swarm nodes.

Incoming requests on swarm nodes are forwarded by the node load balancer to one of the replicated containers through the swarm ingress network.

The incoming request will always be forwarded to a running container, even those who arrives on nodes on which the container is not running.

### A.3 Issues with Rancher hosts for natted machines

It is possible to use Rancher machine drivers to start virtual machines that, after the setup are not reachable from within the Rancher overlay network.

For example, if you do not have any more free floating IP you can still start virtual machines with the openstack driver by unsetting **OPENSTACK\_FLOATINGIP\_POOL**:

```
unset OPENSTACK_FLOATINGIP_POOL
```

then creating new hosts:

```
rancher hosts create rancher-node-02  
rancher hosts create rancher-node-03
```

and list them:

```
rancher host
```

ID	HOSTNAME	STATE	CONTAINERS	IP
1h27	rancher-node-01.novalocal	active	14	
130.206.126.142	swarm=wait_leader			
1h30	rancher-node-02.novalocal	active	11	
130.206.122.186	swarm=manager			
1h31	rancher-node-03.novalocal	active	11	
130.206.122.186	swarm=manager			

Note that the two new host have the same IP, that is the IP used by OpenStack to do the NAT for hosts that do not have a floating IP. An SSH connection as well as the establishment of the overlay network with them is impossible.

```
rancher --debug ssh 1h30
```

```
ssh: connect to host 130.206.122.186 port 22: Connection refused
```

To overcome this issue it is possible to follow the guide at [Using a VPN for overcoming NAT issues](#).

## A.4 Advanced analysis of the issues related to non-standard MTU usage

The MTU for the network interfaces of the Spain2 FIWARE Lab VMs differs from the standard one of **1500** bytes.

This requires to explicitly specify it for every newly created network that uses the Linux bridge driver, otherwise packets could be corrupted by the network stack. Interfaces connected on a bridge need to have all the same MTU (see [here](#)<sup>28</sup>).

The predefined `docker0` and `docker_gwbridge` are both affected, as they use the Linux bridge driver.

The MTU of the **docker0** bridge network can be set by passing the `--mtu=${DOCKER_MTU}` value to the docker daemon.

The **docker\_gwbridge** bridge network used in swarm is also affected. It is used as the default gateway for containers created in swarm mode.

It is created automatically when the swarm is initialized and picks the default(non-configurable!) MTU.

The MTU can be set by (re)creating it with desired parameters initializing the swarm node (see [here](#)<sup>29</sup>):

```
docker network create -o com.docker.network.bridge.enable_icc=false \
-o com.docker.network.bridge.enable_ip_masquerade=true \
-o com.docker.network.driver.mtu=${DOCKER_MTU} \
docker_gwbridge
```

Also **every container** using bridge networks, has to be started by specifying the MTU to assign to the containers interfaces. For Compose v3 declare it in the networks section (see [here](#)<sup>30</sup>):

```
networks:
  default:
    driver: bridge
    driver_opts:
      com.docker.network.driver.mtu: ${DOCKER_MTU:-1400}
```

To pass the MTU with docker-machine use the following snippet:

```
# Set the MTU equal to the one of the default gateway interface
export DOCKER_MTU=1400
docker-machine create rancher-server --engine-opt mtu="${DOCKER_MTU}"
```

*Explicitly setting an MTU value for the docker bridge avoids network issues in case the default route has an MTU different from 1500 (see [#22028](#)<sup>31</sup> and [Customize the docker0 bridge](#)<sup>32</sup>).*

## A.5 Install modern openstackclient with pip

To install the openstackclient you need to satisfy some build dependencies. Please follow the

<sup>28</sup> <https://wiki.linuxfoundation.org/networking/bridge#what-can-be-bridged>

<sup>29</sup> <https://github.com/docker/docker/issues/24906#issuecomment-235894478>

<sup>30</sup> <https://github.com/docker/docker/issues/22297#issuecomment-242934050>

<sup>31</sup> <https://github.com/docker/docker/issues/22028>

<sup>32</sup> [https://docs.docker.com/engine/userguide/networking/default\\_network/custom-docker0/](https://docs.docker.com/engine/userguide/networking/default_network/custom-docker0/)

following snippet.

```
# Enable the deb-src sources in /etc/apt/source.list
sudo sed -i.bak -e 's:# deb-src :deb-src :' /etc/apt/sources.list
sudo apt update
sudo apt install --yes virtualenvwrapper
sudo apt build-dep --yes python-openstackclient
sudo apt build-dep --yes python-netifaces

# There are issues if python-openstackclient==3.9.0 and
# python-novaclient==8.0.0 on #openstack the working suggestion by
# dtroyer was to
pip install python-novaclient==7.1.0
# and wait the fixing version python-openstackclient==3.10.0

. /etc/bash_completion.d/virtualenvwrapper
mkvirtualenv osclient
workon osclient
pip install python-openstackclient
```

## A.6 List Available Images in an OpenStack Project

```
openstack image list --column Name
```

## A.7 List Available Flavors in an OpenStack Project

```
openstack flavor list --column Name
```

## A.8 Useful script to manage Docker

- ➔ stop all docker containers
 

```
sudo docker ps -aq | xargs -tr sudo docker stop
```
- ➔ remove all docker containers
 

```
sudo docker ps -aq | xargs -tr sudo docker rm
```
- ➔ remove all docker volumes
 

```
sudo docker volume ls -q | xargs -tr sudo docker volume rm
```

## A.9 Change the MTU of all interfaces to 1400

```
ip addr show | grep ^[0-9] | awk '{print $2}' | tr -d : | xargs -I X sudo
ifconfig X mtu 1400
```

## A.10 Workaround for sudo complaint

Usually the hosts created in cloud environment do not have proper fully qualified host name. Sudo complains a bit with the following output:

```
# sudo: unable to resolve host $HOSTNAME
```

To fix the verbose sudo warning about the missing FQDN use the following:

```
echo 127.0.1.1 $(hostname) | sudo tee -a /etc/hosts
```

### A.11 Add the docker group to the current user

To avoid typing sudo every time before the **docker** command you may want to issue:

```
sudo usermod -aG docker "${USER}"
echo exit an login to make the group change effective
echo or launch another login shell
```

### A.12 Workaround to view display error on FIWARE Lab portal

In order to get the “[Connect to VM display \(view display\)](http://cloud.lab.fiware.org/vnc_display)”<sup>33</sup> working in the [FIWARE Lab portal](https://www.fiware.org/lab/)<sup>34</sup>, you need to enable 3rd party cookies. If not, you will see the error: **Failed to connect to server (code: 1006)**.

### A.13 Rancher General cleanup

```
# Select your rancher host from docker swarm masters manually
RANCHER_HOST=1h3
rancher --host "${RANCHER_HOST}" docker ps -a
rancher --host "${RANCHER_HOST}" docker ps -a -q --filter status=created | \
  xargs -r rancher --host "${RANCHER_HOST}" docker rm
rancher --host "${RANCHER_HOST}" docker ps -a -q --filter status=exited | \
  xargs -r rancher --host "${RANCHER_HOST}" docker rm
rancher --host "${RANCHER_HOST}" docker ps -a
```

---

<sup>33</sup> [http://cloud.lab.fiware.org/vnc\\_display](http://cloud.lab.fiware.org/vnc_display)

<sup>34</sup> <https://www.fiware.org/lab/>