Grant Agreement No.: 732638
Call: H2020-ICT-2016-2017
Topic: ICT-13-2016
Type of action: RIA

D3.4: Developments for the second cycle

Work package	WP 3
Task	Task 3.1-3.5
Due date	31/12/2019
Submission date	17/3/2020
Deliverable lead	Imec
Version	3
Authors	Brecht Vermeulen (imec), Wim Van der Meerssche (imec), Thijs Walcarius (imec), Albert (Yiu Quan) Su (SU), Dimitris Dechouniotis (NTUA), Costas Papadakis (NTUA), Aris Dadoukis (CERTH), Donatos Stavropoulos (CERTH), Ana Juan Ferrer (ATOS), Roman Sosa Gonzalez (ATOS), Ana Juan Ferrer (ATOS), Rowshan Jahan Sathi (TUB), Alex Willner (TUB), Lucas Nussbaum (Inria), David Margery (Inria), Cedric Crettaz (MI)
Reviewers	Peter Van Daele (imec)

Abstract	This deliverable gives an overview of the developments in WP3 during the second 18 months of the project. WP2 are normal operations developments (add testbeds, fix bugs, small features, etc). WP3 is focussing on larger new functionality.
Keywords	Developments second cycle, new functionality

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1	1/10/2019	TOC	Brecht Vermeulen (imec)
V2	22/12/2019	First version	Brecht Vermeulen (imec), Wim Van der Meerssche (imec), Thijs Walcarius (imec), Albert (Yiu Quan) Su (SU), Dimitris Dechouniotis (NTUA), Costas Papadakis (NTUA), Aris Dadoukis (CERTH), Donatos Stavropoulos (CERTH), Ana Juan Ferrer (ATOS), Roman Sosa Gonzalez (ATOS), Joaquin Iranzo Yuste (ATOS), Rowshan Jahan Sathi (TUB), Alex Willner (TUB), Lucas Nussbaum (Inria), David Margery (Inria), Cedric Crettaz (MI)
V3	10/03/2020	Final version	Brecht Vermeulen (imec)

DISCLAIMER

COPYRIGHT NOTICE

ACKNOWLEDGMENT

Co-funded by the European Union

Co-funded by the Swiss Confederation

This deliverable has been written in the context of a Horizon 2020 European research project, which is co-funded by the European Commission and the Swiss State Secretariat for Education, Research and Innovation. The opinions expressed and arguments employed do not engage the supporting parties.

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	✓
CL	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to FED4FIRE+ 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.

Co-funded by the Horizon 2020 Framework Programme of the European Union

EXECUTIVE SUMMARY

This deliverable gives an overview of the developments in WP3 during the first 18 months of the project. WP2 are normal operations developments (add testbeds, fix bugs, small features, etc). WP3 is focussing on larger new functionality.

WP3 consists out of the following tasks, which are also the sequence of sections in this deliverable:

- Task 3.1 is focussing on SLA and reputation for testbed usage
- Task 3.2 is focussing on Experiment-as-a-Service (EaaS), data retention and reproducibility of experiments
- Task 3.3 is targeting Federation monitoring and interconnectivity
- Task 3.4 works on Service orchestration and brokering
- Task 3.5 researches ontologies for the federation of testbeds
- As described in D3.3, the following extra developments have been made based on demands of users, tool developers and testbed owners:
    - New user account portal with OAuth
    - A tool (for the Fed4FIRE.eu website) to allow to more easily chose testbeds
    - An automated setup for openstack with Fed4FIRE tools

Co-funded by the Horizon 2020
Framework Programme of the European Union

TABLE OF CONTENTS

Co-funded by the Horizon 2020 Framework Programme of the European Union

Co-funded by the Horizon 2020
Framework Programme of the European Union

LIST OF FIGURES

Co-funded by the Horizon 2020 Framework Programme of the European Union

Co-funded by the Horizon 2020
Framework Programme of the European Union

LIST OF TABLES

Co-funded by the Horizon 2020
Framework Programme of the European Union

ABBREVIATIONS

FIRE	Future Internet Research and Experimentation
JSON	JavaScript Object Notation
SLA	Service Level Agreement
SLO	Service Level Objective
XML	eXtensible Markup Language
WSAG	Web Service-Agreement
API	Application Programming Interface
XML-RPC:	Extensible Markup Language Remote procedure call
REST	REpresentational State Transfer
AM	Aggregate Manager
QoS	Quality of Service
QoE	Quality of Experience
MVC	Model-View-Controller
O/RM	Object-relational mapping
GUI	Graphical User Interface
CLI	Command Line Interface
HRS	Hybrid Reputation System
KPI	Key Performance Indicator
FAHP	Fuzzy Analytic Hierarchical Process

Co-funded by the Horizon 2020
Framework Programme of the European Union

1 INTRODUCTION

This deliverable gives an overview of the developments in WP3 during the second 18 months of the project. WP2 are normal operations developments (add testbeds, fix bugs, small features, etc). WP3 is focussing on larger new functionality. D3.2 described the developments during the first 18 months of the project.

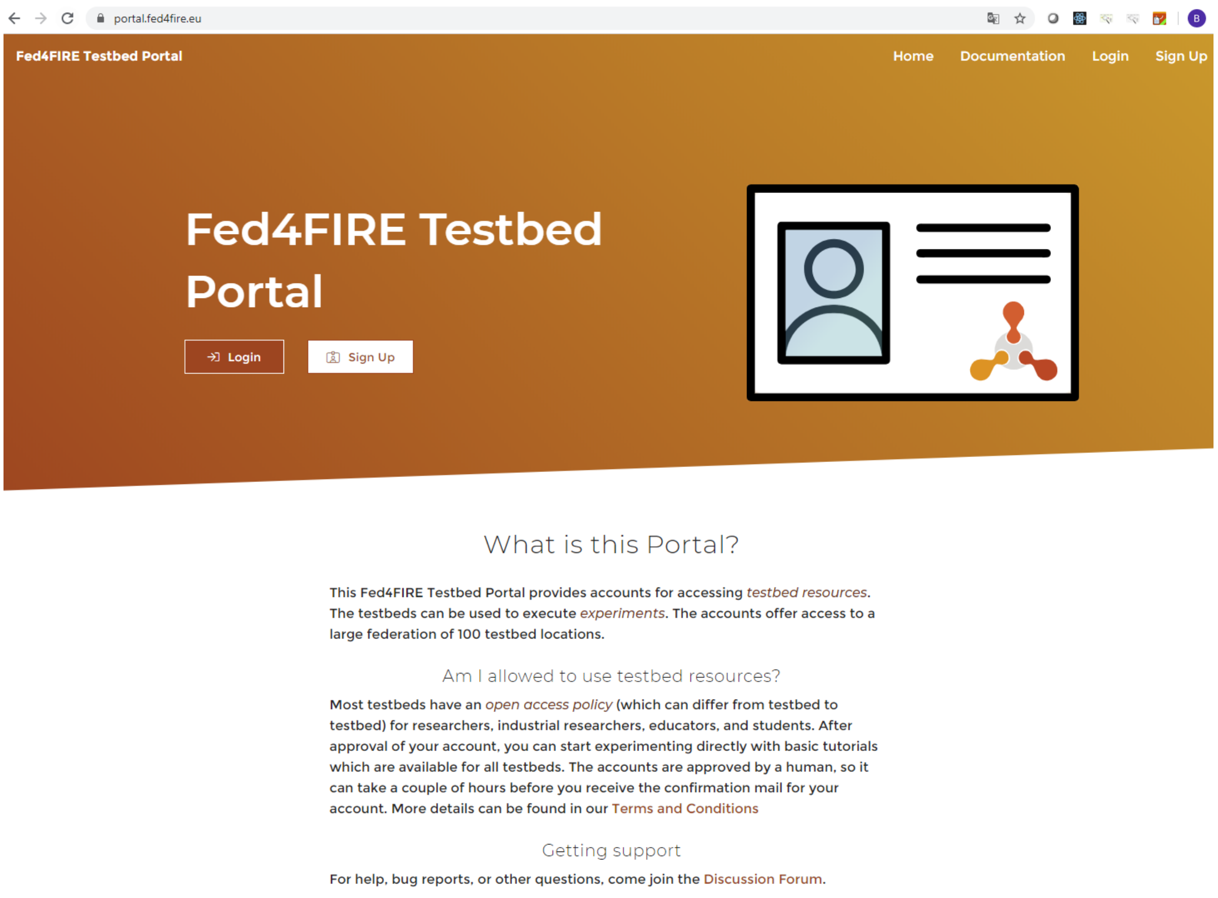
WP3 consists out of the following tasks, which are also the sequence of sections in this deliverable:

- Task 3.1 is focussing on SLA and reputation for testbed usage
- Task 3.2 is focussing on Experiment-as-a-Service (EaaS), data retention and reproducibility of experiments
- Task 3.3 is targeting Federation monitoring and interconnectivity
- Task 3.4 works on Service orchestration and brokering
- Task 3.5 researches ontologies for the federation of testbeds

Co-funded by the Horizon 2020
Framework Programme of the European Union

2 NEW USER ACCOUNT PORTAL

A new user portal for the user accounts and project registration has been brought online at https://portal.fed4fire.eu.



What is this Portal?

This Fed4FIRE Testbed Portal provides accounts for accessing *testbed resources*. The testbeds can be used to execute *experiments*. The accounts offer access to a large federation of 100 testbed locations.

Am I allowed to use testbed resources?

Most testbeds have an *open access policy* (which can differ from testbed to testbed) for researchers, industrial researchers, educators, and students. After approval of your account, you can start experimenting directly with basic tutorials which are available for all testbeds. The accounts are approved by a human, so it can take a couple of hours before you receive the confirmation mail for your account. More details can be found in our Terms and Conditions

Getting support

For help, bug reports, or other questions, come join the Discussion Forum.

Figure 1: New https://portal.fed4fire.eu portal

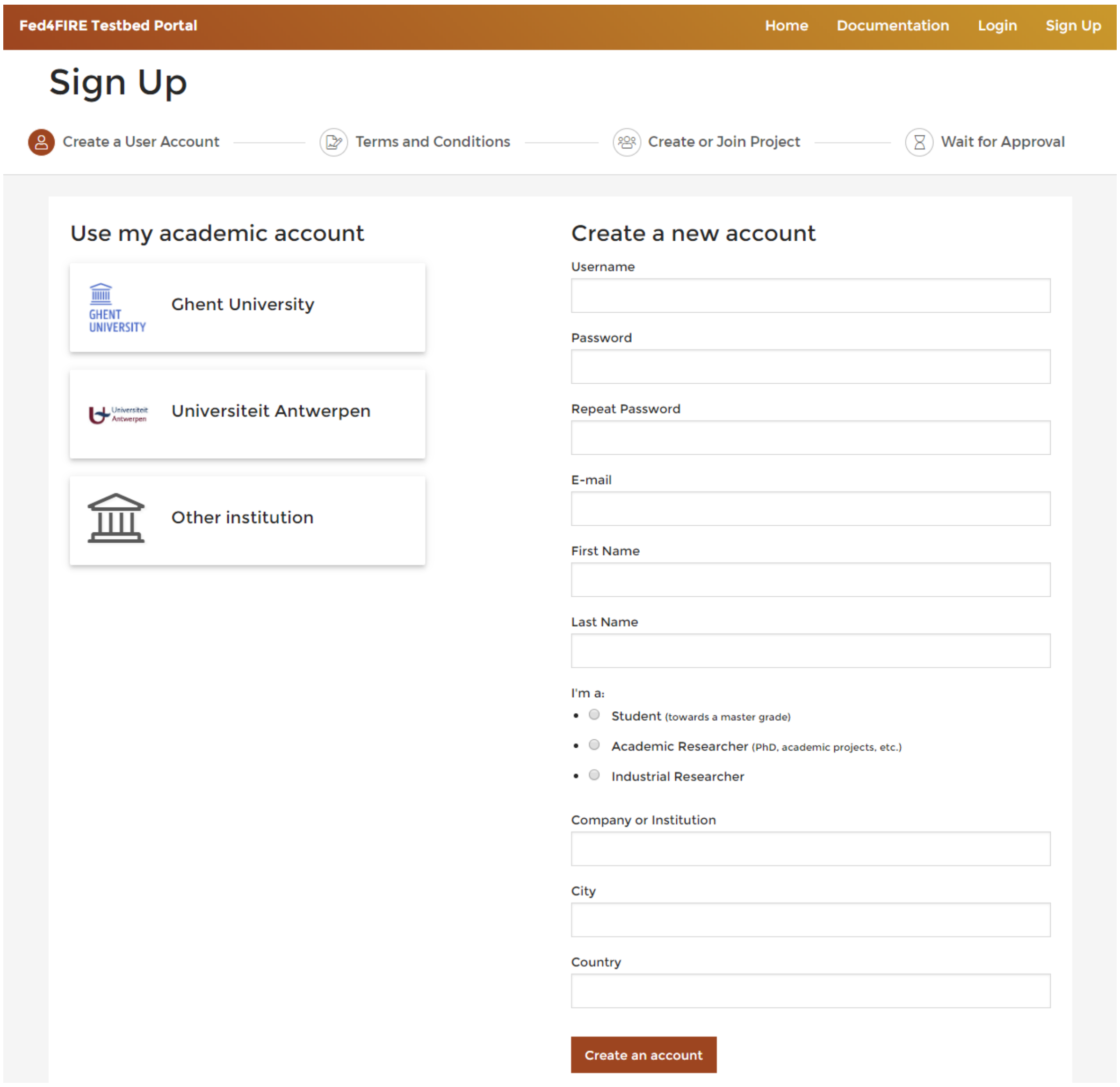
The main benefits from the new portal are the following:

- Better Fed4FIRE branding and look&feel (see above screenshot)
- More user friendly portal (e.g. to invite people to a project or for student classes)
- Better and more clear flow for approval of terms and conditions and GDPR terms
- Possibility to use edugain login for academics (=university home account)
- Extra information is gathered for the user accounts and is put in the user credential to make it possible for testbeds to allow more fine-grained access
- Logging for auditing
- Better statistics are possible on the usage of the testbeds
- OAuth API to make it easy for other (web-based) services to use the same account base

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.1 USER FRIENDLINESS AND MORE CLEAR FLOW

When signing up for a new account, the new steps are now clearly shown at the top and it is also clearly shown that you can use your academic login or create a local account. We do ask also extra information on the user type (Student, Academic researcher, industrial researcher) as this can make a difference for testbeds to accept experiments of these users (e.g. academic research can use more resources than a student, industrial researches are limited in resource use for free, etc.).



Fed4FIRE Testbed Portal — Home | Documentation | Login | Sign Up

Sign Up

Create a User Account — Terms and Conditions — Create or Join Project — Wait for Approval

Use my academic account

- Ghent University
- Universiteit Antwerpen
- Other institution

Create a new account

Username

Password

Repeat Password

E-mail

First Name

Last Name

I'm a:

- Student (towards a master grade)
- Academic Researcher (PhD, academic projects, etc.)
- Industrial Researcher

Company or Institution

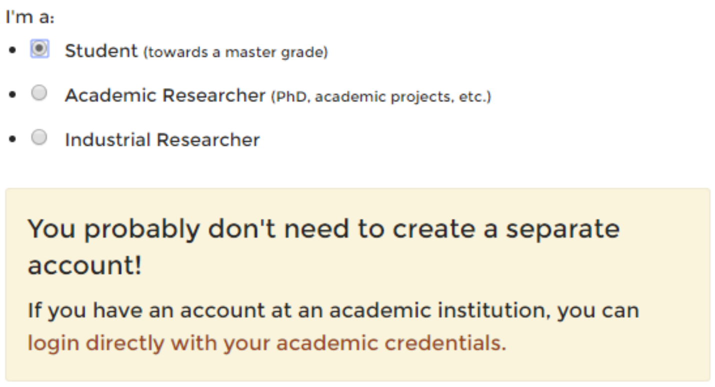
City

Country

Create an account

Figure 2: User friendly sign up form and edugain academic login

Co-funded by the Horizon 2020 Framework Programme of the European Union



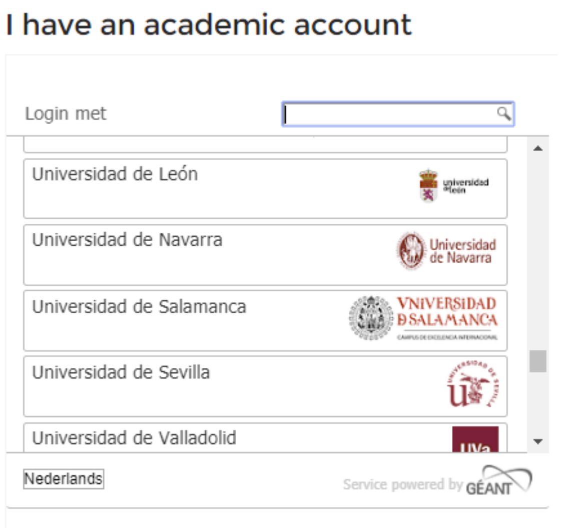
I'm a:

- [✓ Student] (towards a master grade)
- Academic Researcher (PhD, academic projects, etc.)
- Industrial Researcher

You probably don't need to create a separate account!

If you have an account at an academic institution, you can login directly with your academic credentials.

Figure 3: When selecting academic user type, it is suggested to use the edugain login



I have an academic account

Login met

Universidad de León

Universidad de Navarra

Universidad de Salamanca

Universidad de Sevilla

Universidad de Valladolid

Nederlands

Service powered by GÉANT

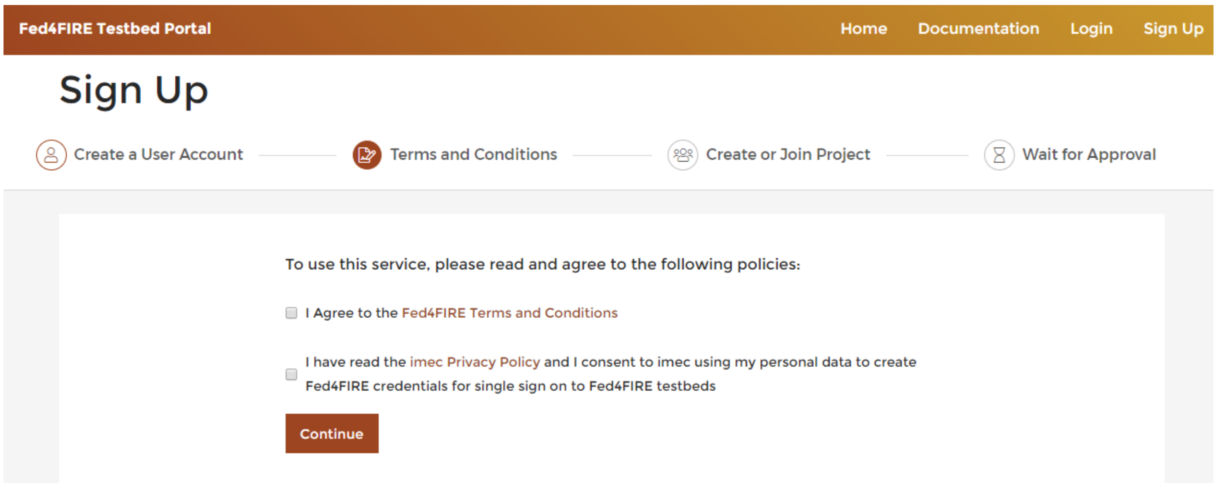
Figure 4: Edugain institute choser

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.2 TERMS AND CONDITIONS AND PRIVACY

The sign up sequence has now also a clear overview of the terms and conditions and privacy policy. The Fed4FIRE terms and conditions can be found at https://fed4fire.eu/terms.

As Fed4FIRE is no legal entity at this moment, and this new portal runs in the imec datacentre, the imec privacy policy is used (https://www.imec-int.com/en/privacy-statement).



To use this service, please read and agree to the following policies:

☐ I Agree to the Fed4FIRE Terms and Conditions

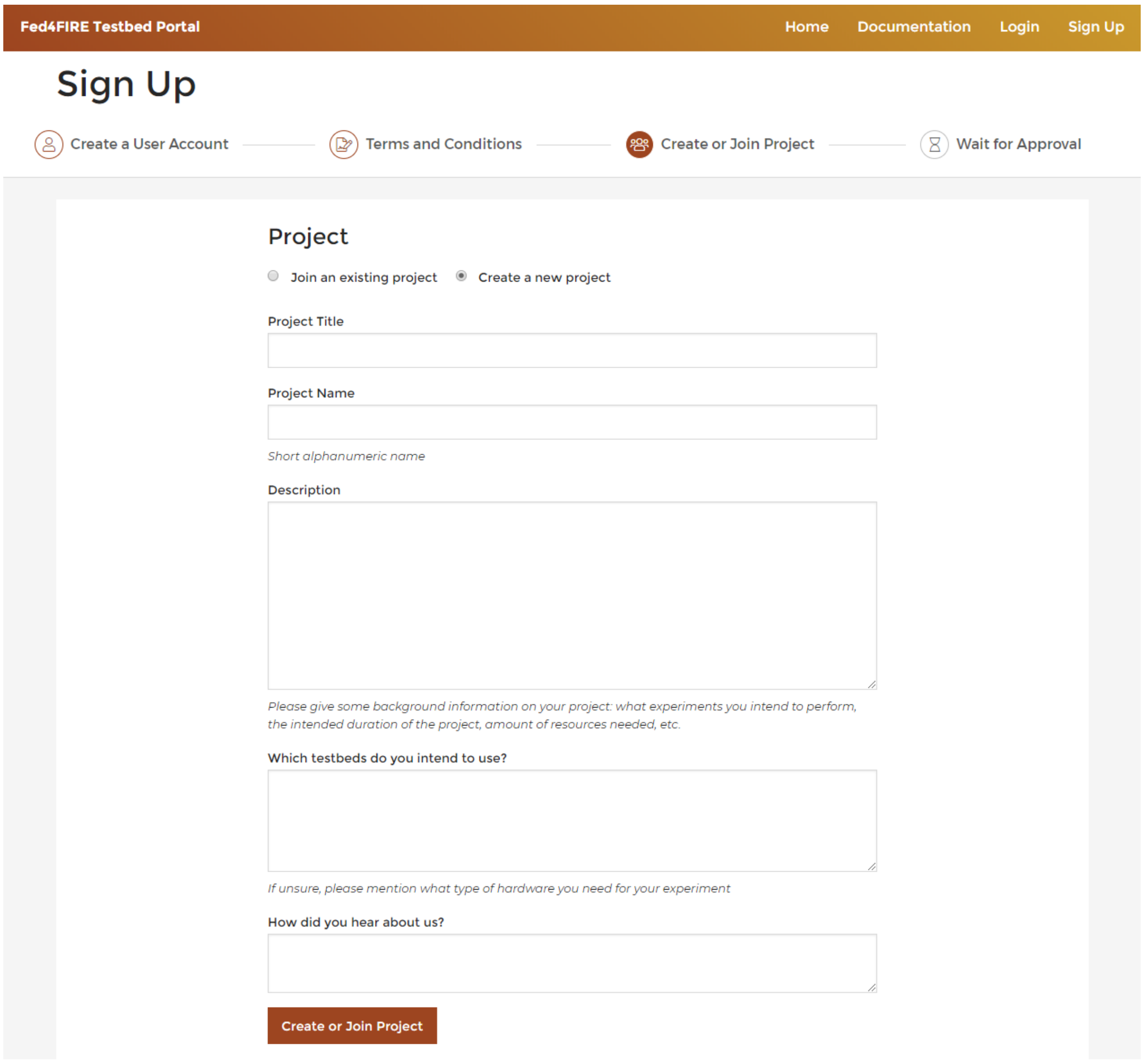
☐ I have read the imec Privacy Policy and I consent to imec using my personal data to create Fed4FIRE credentials for single sign on to Fed4FIRE testbeds

Figure 5: Clear overview of terms and conditions and privacy policy

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.3 NEW PROJECT REQUEST

If people ask for a new project, we do ask extra information: which testbeds they want to use (so we can inform the right testbed owners) and how they heard about Fed4FIRE. New projects always need to be manually approved by a portal administrator. Even if the account comes from an edugain institute, we still verify the project request manually.



Sign Up

Create a User Account — Terms and Conditions — Create or Join Project — Wait for Approval

Project

Join an existing project [✓ Create a new project]

Project Title

Project Name

Short alphanumeric name

Description

Please give some background information on your project: what experiments you intend to perform, the intended duration of the project, amount of resources needed, etc.

Which testbeds do you intend to use?

If unsure, please mention what type of hardware you need for your experiment

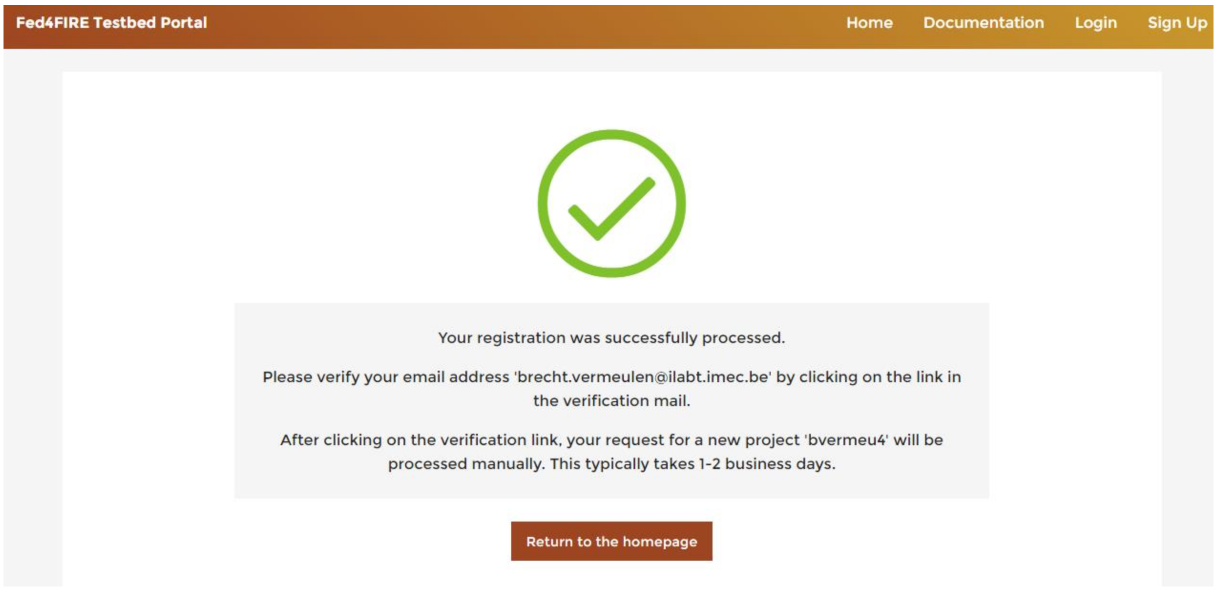
How did you hear about us?

Figure 6: New project request form asking for which testbeds to use and how they heard about Fed4FIRE

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.4 PROJECT APPROVAL

As before, the approval process is the same. The imec administrators approve manually the PIs/new project requests. After that, the PIs of the project are responsible for approving users in their projects.



Your registration was successfully processed.

Please verify your email address 'brecht.vermeulen@ilabt.imec.be' by clicking on the link in the verification mail.

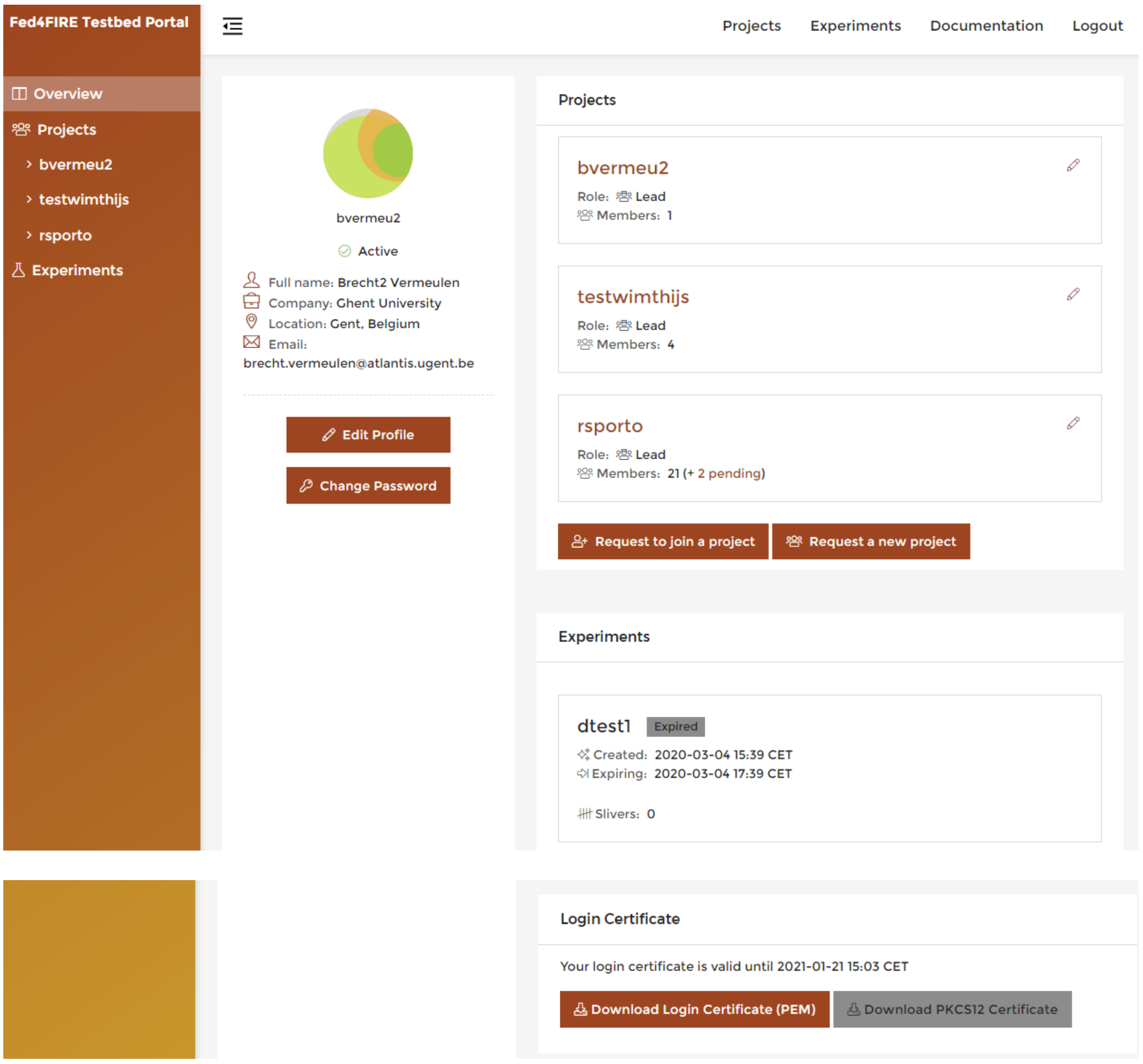
After clicking on the verification link, your request for a new project 'bvermeu4' will be processed manually. This typically takes 1-2 business days.

Figure 7: Final step: approving email confirmation and waiting for administrator approval

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.5 PORTAL DASHBOARD AFTER APPROVAL

The screenshot below shows the dashboard a user sees after login. It is a quick overview of the projects the user is member off, the user profile, the last experiments and the possibility to download the PEM certificate for use in other tools.


Fed4FIRE Testbed Portal

Projects Experiments Documentation Logout

Overview
Projects
› bvermeu2
› testwimthijs
› rsporto
Experiments

bvermeu2
Active
Full name: Brecht2 Vermeulen
Company: Ghent University
Location: Gent, Belgium
Email:
brecht.vermeulen@atlantis.ugent.be

Edit Profile
Change Password

Projects

bvermeu2
Role: Lead
Members: 1

testwimthijs
Role: Lead
Members: 4

rsporto
Role: Lead
Members: 21 (+ 2 pending)

Request to join a project
Request a new project

Experiments

dtest1 Expired
Created: 2020-03-04 15:39 CET
Expiring: 2020-03-04 17:39 CET

Slivers: 0

Login Certificate

Your login certificate is valid until 2021-01-21 15:03 CET

Download Login Certificate (PEM)
Download PKCS12 Certificate

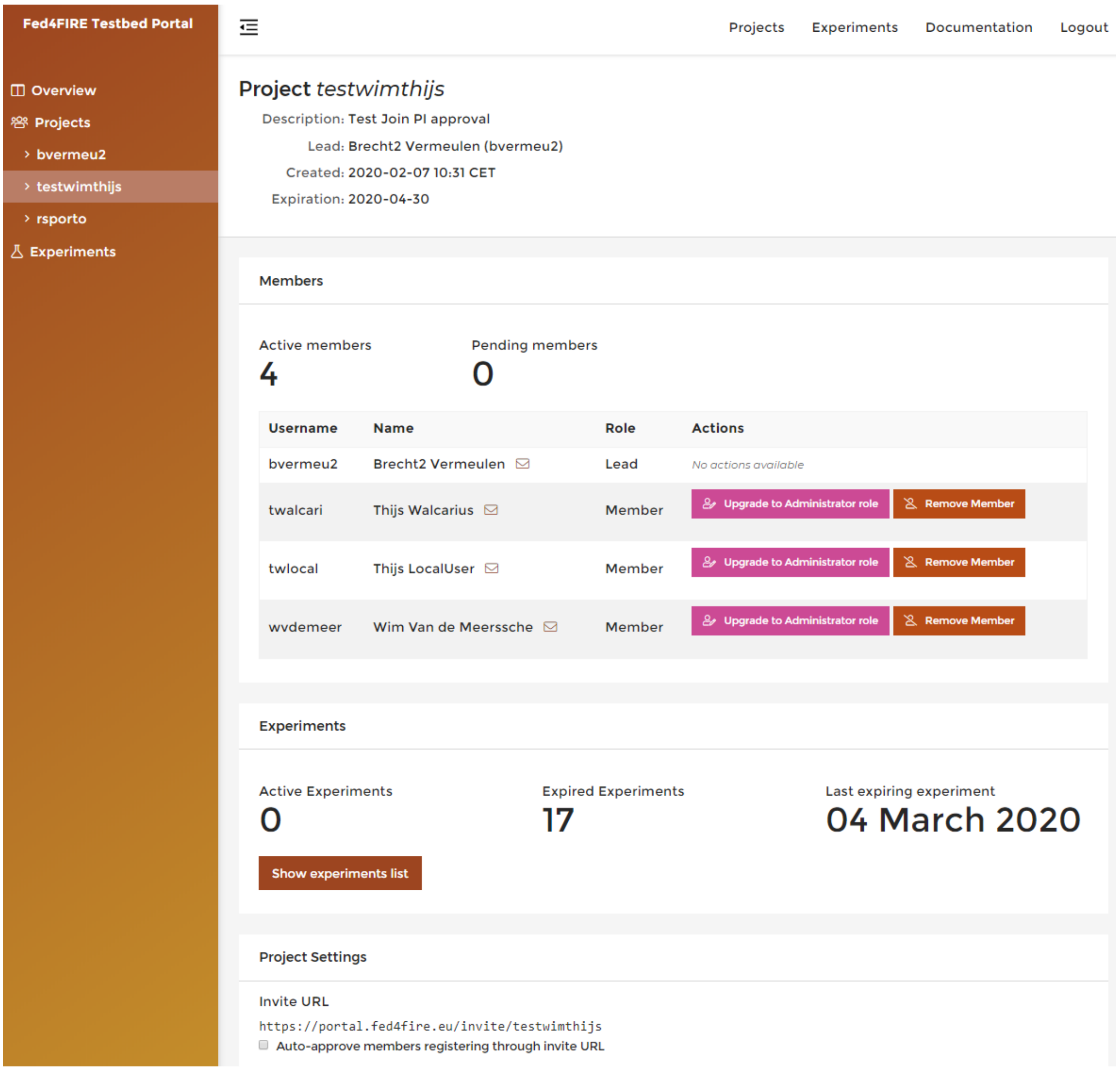
Figure 8: Dashboard after approval of the account

Co-funded by the Horizon 2020
Framework Programme of the European Union

2.6 PER PROJECT FUNCTIONALITY

The dashboard per project shows the current members and their role (lead, admin, member) and the number of experiments in that project.

It contains also an invite url to easily invite other users and it is possible to change this in an auto-approval (random) url, e.g. for class exercises where you expect many students at once.



Project *testwimthijs*

Description: Test Join PI approval
Lead: Brecht2 Vermeulen (bvermeu2)
Created: 2020-02-07 10:31 CET
Expiration: 2020-04-30

Members

Active members 4

Pending members 0

Username	Name	Role	Actions
bvermeu2	Brecht2 Vermeulen	Lead	*No actions available*
twalcari	Thijs Walcarius	Member	Upgrade to Administrator role / Remove Member
twlocal	Thijs LocalUser	Member	Upgrade to Administrator role / Remove Member
wvdemeer	Wim Van de Meerssche	Member	Upgrade to Administrator role / Remove Member

Experiments

Active Experiments 0

Expired Experiments 17

Last expiring experiment 04 March 2020

Show experiments list

Project Settings

Invite URL

`https://portal.fed4fire.eu/invite/testwimthijs`

☐ Auto-approve members registering through invite URL

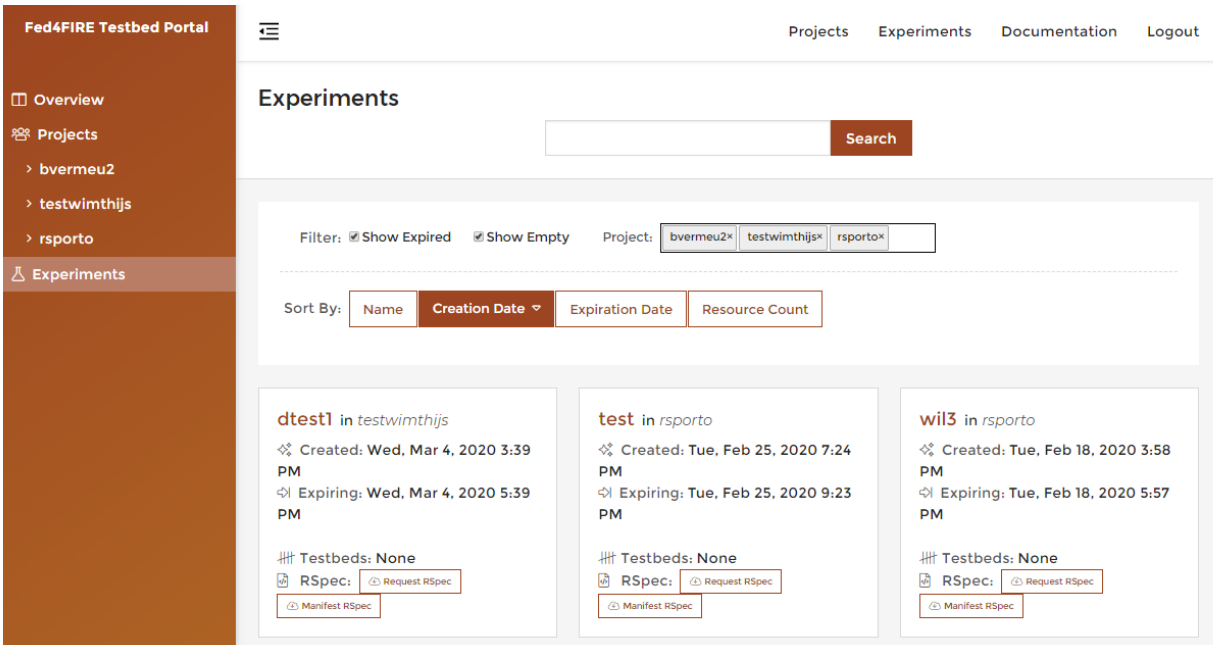
Figure 9: Dashboard after approval of the account

Co-funded by the Horizon 2020
Framework Programme of the European Union

2.7 DASHBOARD EXPERIMENT OVERVIEW

In the below screenshot you can see how the user sees the list of expired and running experiments and the following functionality is present:

- Possibility to download request and manifest rspec (to rerun an old experiment e.g.)
- Running experiments: possibility to open directly in jFed by clicking a button
- Easy filter, sort and search functionality



Experiments

Filter: ☑ Show Expired ☑ Show Empty Project: bvermeu2× testwimthijs× rsporto×

Sort By: Name | Creation Date | Expiration Date | Resource Count

dtest1 in *testwimthijs*
Created: Wed, Mar 4, 2020 3:39 PM
Expiring: Wed, Mar 4, 2020 5:39 PM
Testbeds: None
RSpec: Request RSpec Manifest RSpec

test in *rsporto*
Created: Tue, Feb 25, 2020 7:24 PM
Expiring: Tue, Feb 25, 2020 9:23 PM
Testbeds: None
RSpec: Request RSpec Manifest RSpec

wil3 in *rsporto*
Created: Tue, Feb 18, 2020 3:58 PM
Expiring: Tue, Feb 18, 2020 5:57 PM
Testbeds: None
RSpec: Request RSpec Manifest RSpec

Figure 10: Dashboard showing expired and running experiments

Co-funded by the Horizon 2020 Framework Programme of the European Union

2.8 ADMIN VIEW

The portal administrators have access to admin pages in the portal to view users, projects and experiments.

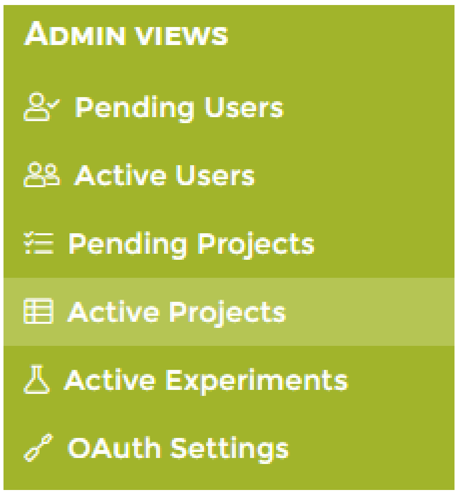
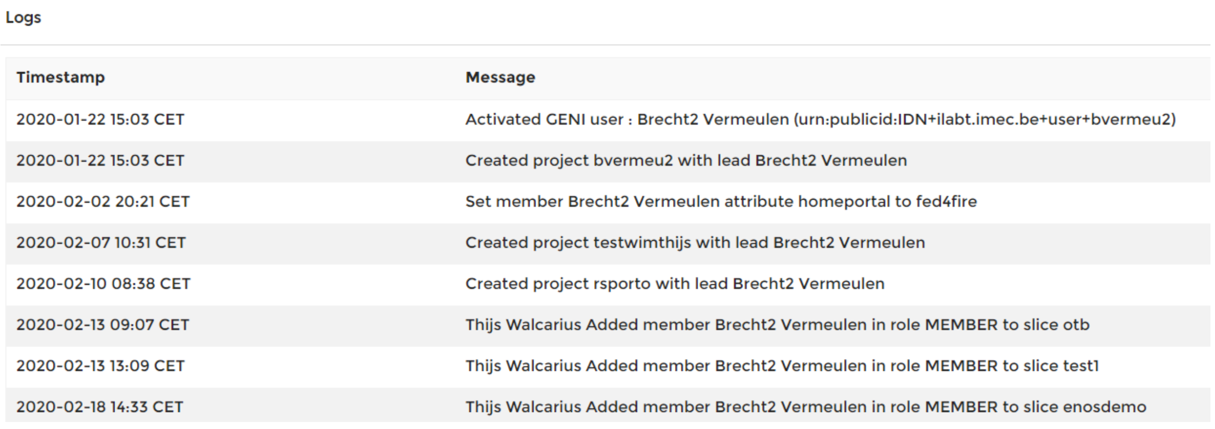


Figure 11: Admin functionality of the portal

2.9 LOGGING FOR AUDITING

In this new portal, we also have better logging of all actions so we can use it for auditing purposes. See the example below:



Logs

Timestamp	Message
2020-01-22 15:03 CET	Activated GENI user : Brecht2 Vermeulen (urn:publicid:IDN+ilabt.imec.be+user+bvermeu2)
2020-01-22 15:03 CET	Created project bvermeu2 with lead Brecht2 Vermeulen
2020-02-02 20:21 CET	Set member Brecht2 Vermeulen attribute homeportal to fed4fire
2020-02-07 10:31 CET	Created project testwimthijs with lead Brecht2 Vermeulen
2020-02-10 08:38 CET	Created project rsporto with lead Brecht2 Vermeulen
2020-02-13 09:07 CET	Thijs Walcarius Added member Brecht2 Vermeulen in role MEMBER to slice otb
2020-02-13 13:09 CET	Thijs Walcarius Added member Brecht2 Vermeulen in role MEMBER to slice test1
2020-02-18 14:33 CET	Thijs Walcarius Added member Brecht2 Vermeulen in role MEMBER to slice enosdemo

Figure 12: Logging for auditing purposes

Co-funded by the Horizon 2020 Framework Programme of the European Union

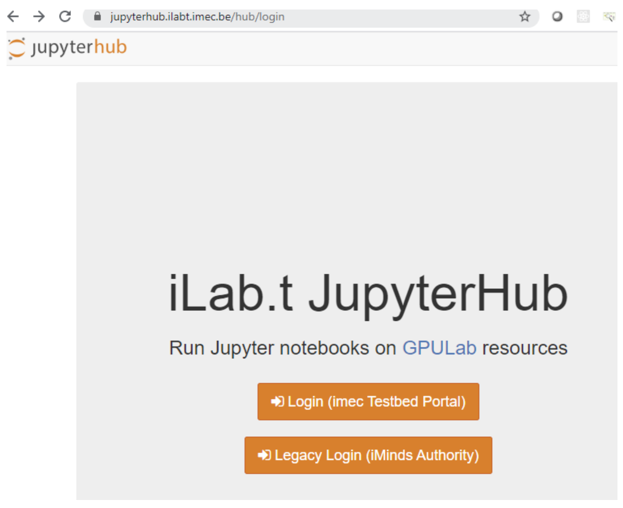
2.10 OAUTH

An important feature of the portal is the OAuth functionality. OAuth is an open standard for access delegation, commonly used as a way for Internet users to grant websites or applications access to their information on other websites but without giving them the passwords.

Generally, OAuth provides to clients a "secure delegated access" to server resources on behalf of a resource owner. It specifies a process for resource owners to authorize third-party access to their server resources without sharing their credentials. Designed specifically to work with Hypertext Transfer Protocol (HTTP), OAuth essentially allows access tokens to be issued to third-party clients by an authorization server, with the approval of the resource owner. The third party then uses the access token to access the protected resources hosted by the resource server.

The OAuth functionality is useful for all kind of web-based services, e.g. also the SLA and reputation service discussed later on.

Example of GPULab Jupyterhub: when you browse to the jupyterhub website, you can chose the authority you want to use.


jupyterhub.ilabt.imec.be/hub/login

jupyterhub

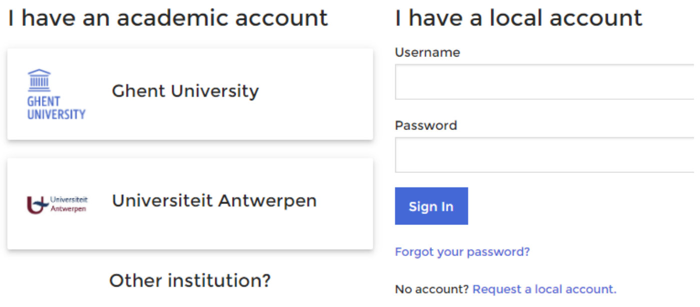
iLab.t JupyterHub

Run Jupyter notebooks on GPULab resources

Login (imec Testbed Portal)

Legacy Login (iMinds Authority)

You are redirected to the login page of the authority:


I have an academic account

Ghent University

Universiteit Antwerpen

Other institution?

I have a local account

Username

Password

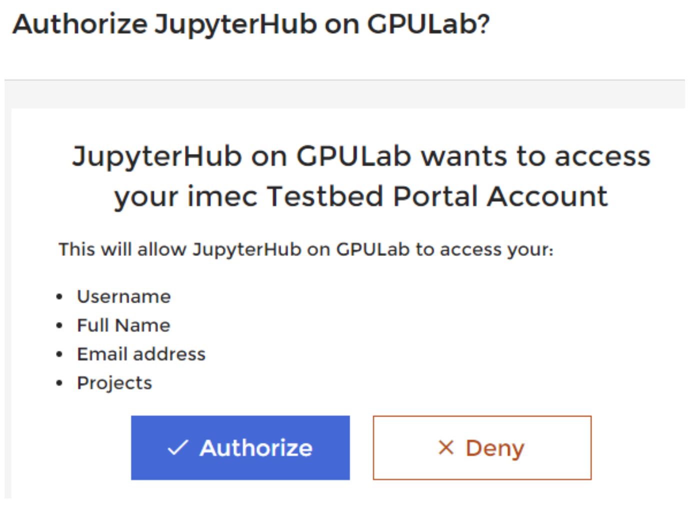
Sign In

Forgot your password?

No account? Request a local account.

Co-funded by the Horizon 2020
Framework Programme of the European Union

And after login, you are asked if you allow Jupyterhub to receive the listed information:



Authorize JupyterHub on GPULab?

JupyterHub on GPULab wants to access your imec Testbed Portal Account

This will allow JupyterHub on GPULab to access your:

- Username
- Full Name
- Email address
- Projects

Figure 13: OAuth login procedure for services.

Co-funded by the Horizon 2020 Framework Programme of the European Union

3 TESTBED CHOSER FOR FED4FIRE.EU WEBSITE

When people come to Fed4FIRE and try to find the right testbed for their experiment, or want to check if they can use Fed4FIRE for their experimentation, they use now https://www.fed4fire.eu/testbeds/.

This consists of a map with color-coded testbed labels, based on 7 testbed types (wired, wireless, 5G, IoT, openflow, cloud, big data). Further on, each testbed has a more detailed description with a link to a specific documentation website and contact email.

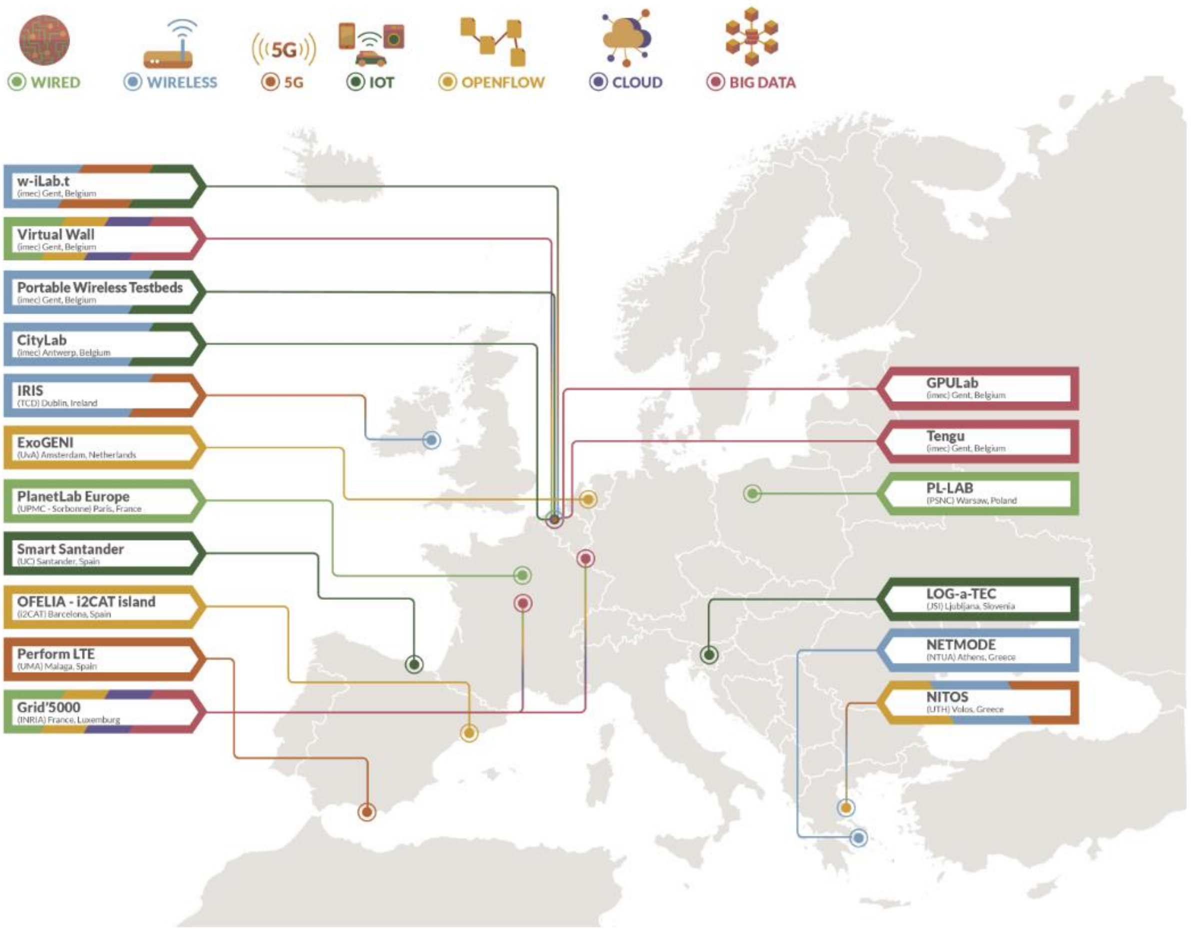


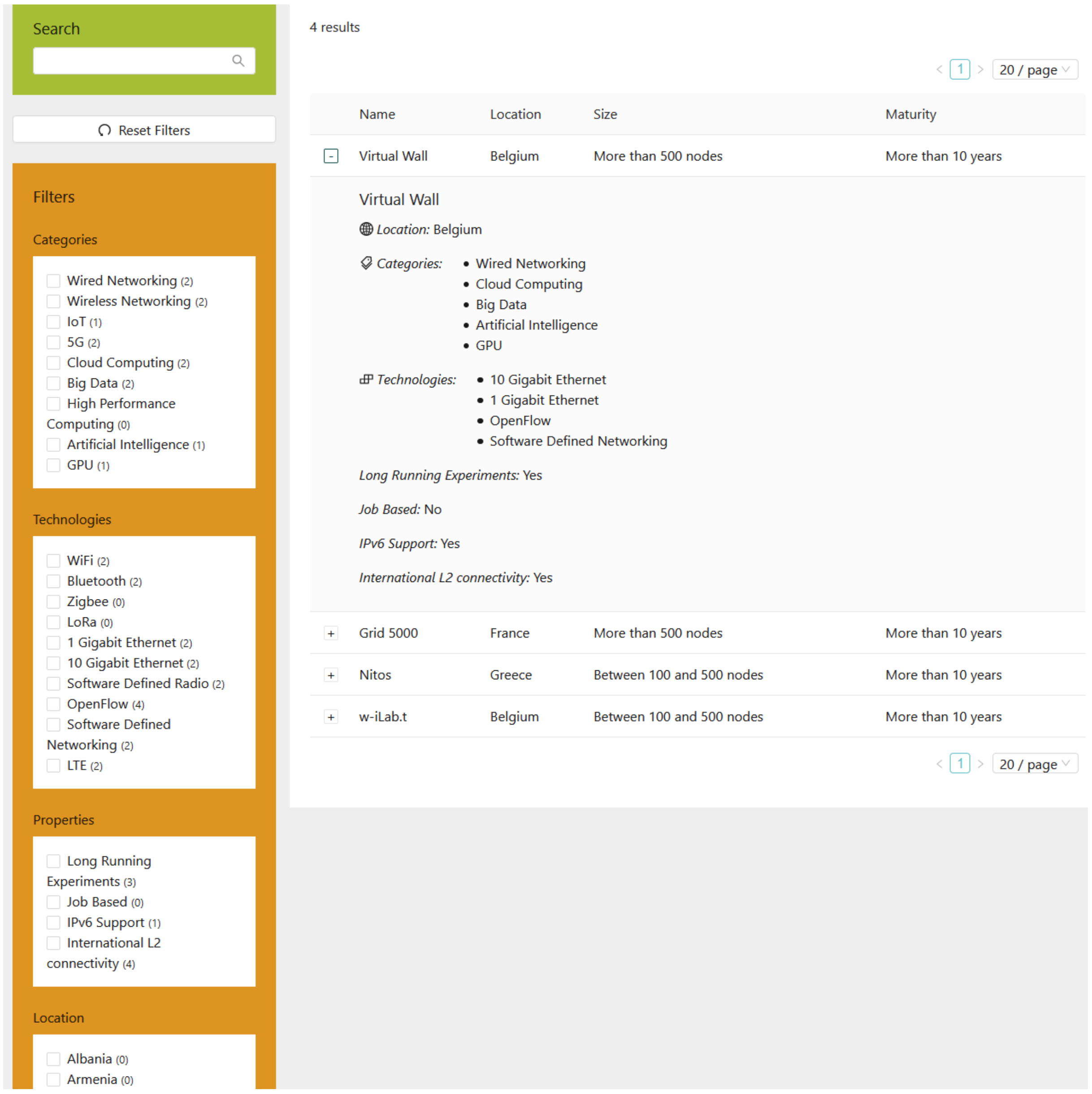
Figure 14: Current testbed map with technology labelling

However, this is too limited for some visitors/possible users. If you are looking e.g. for GPUs or for LTE or for IoT IPv6, you have to basically run through too many testbed descriptions.

Based on this simple question, it seems that ontologies could answer this, but then you need a detailed ontology for all testbeds and the queries need to be quite detailed. So we have set up something in between: not too complex to set up and maintain (adapt to user questions), helps users, and only needed for the website for now.

Co-funded by the Horizon 2020
Framework Programme of the European Union

We have come up with the below web-based tool, based on filters and search. So, now you can easily combine multiple filters and look at a basic decription and have e.g. extra info about size and maturity of the testbed.



Search

Reset Filters

Filters

Categories

- Wired Networking (2)
- Wireless Networking (2)
- IoT (1)
- 5G (2)
- Cloud Computing (2)
- Big Data (2)
- High Performance Computing (0)
- Artificial Intelligence (1)
- GPU (1)

Technologies

- WiFi (2)
- Bluetooth (2)
- Zigbee (0)
- LoRa (0)
- 1 Gigabit Ethernet (2)
- 10 Gigabit Ethernet (2)
- Software Defined Radio (2)
- OpenFlow (4)
- Software Defined Networking (2)
- LTE (2)

Properties

- Long Running Experiments (3)
- Job Based (0)
- IPv6 Support (1)
- International L2 connectivity (4)

Location

- Albania (0)
- Armenia (0)

4 results

1 20 / page

	Name	Location	Size	Maturity
-	Virtual Wall	Belgium	More than 500 nodes	More than 10 years

Virtual Wall

Location: Belgium

Categories:
- Wired Networking
- Cloud Computing
- Big Data
- Artificial Intelligence
- GPU

Technologies:
- 10 Gigabit Ethernet
- 1 Gigabit Ethernet
- OpenFlow
- Software Defined Networking

Long Running Experiments: Yes

Job Based: No

IPv6 Support: Yes

International L2 connectivity: Yes

	Name	Location	Size	Maturity
+	Grid 5000	France	More than 500 nodes	More than 10 years
+	Nitos	Greece	Between 100 and 500 nodes	More than 10 years
+	w-iLab.t	Belgium	Between 100 and 500 nodes	More than 10 years

1 20 / page

Figure 15: Testbed selection with easy filters

The input for this filter system is a very simple and easy (in comparison to e.g. ontologies) to maintain json description:

```
[
  {
    "name": "Virtual Wall",
```

Co-funded by the Horizon 2020
Framework Programme of the European Union

```
    "country": "belgium",
    "categories": [
      "wired",
      "cloud",
      "bigdata",
      "ai",
      "gpu"
    ],
    "technologies": [
      "10gb",
      "1gb",
      "openflow",
      "sdn"
    ],
    "size": "500plus",
    "maturity": "10plus",
    "properties": [    "long_running_experiments",
      "ipv6",
      "international_l2"]
```

The next steps for bringing this into production on the Fed4FIRE website:

- Add all testbeds and the relevant features
- Chose the right filters
- Make a wordpress plugin of this, so it can be used in the wordpress Fed4FIRE website
- Look and feel adapted for the website

This is planned for Q1-Q2 2020.

4 SLA AND REPUTATION SERVICE

In the Fed4FIRE+ environment, the Service Level Agreement (SLA) and the Reputation Service provide the necessary tools and mechanisms for delivering to the users a quantitative view of the trustworthiness of the federated testbeds. This service facilitates the Fed4FIRE+ users to select the appropriate testbeds in the federation according to their experiment's requirements and the testbeds provide SLA on specific QoS metrics.

The aim of adding SLA within Fed4FIRE+ is to enable testbed providers to create offerings that experimenters can accept establishing an agreement with the testbed owner. We can understand the agreement as a contract between the platform providers and the testbed users. Once the agreement has been created, it must be verified that it is being fulfilled. The information related to the execution of an experiment, i.e., if there is an agreement violation, will be send to the other components using a notification / subscription pattern.

The Reputation Service of Fed4FIRE+ aims to enhance and extend the already-developed reputation service of Fed4Fire project. The updated service will leverage Quality of Service (QoS) metrics, such as Availability, Latency etc., Quality of Experience (QoE) metrics, e.g., Usability and Documentation Readability, and SLA data in order to compute the degree of confidence of both experimenters and testbed. At the end of an experiment, the users will be prompted to give their feedback for the reserved testbeds in order to update the reputation score of the testbed and the credibility score of the experimenter. This process mitigates the effect of abnormal or malicious evaluations and guarantees that the testbeds' reputation score is fairly computed.

During the first cycle, the SLA and reputation services were developed. At the second cycle, these services will be continuously updated and will be integrated with the core testbeds of the federation.

4.1 REPUTATION DEVELOPMENTS FOR THE SECOND CYCLE

During the first cycle, the testing and execution of the experiment evaluation lifecycle was conducted through REST API calls with the Reputation Service frontend tools. One of the key features of the HRS is the credibility mechanism which allows the Reputation Service to reduce the impact of malicious users in the computation of the reputation score of each testbed. Through the testing phase, the credibility mechanism was considered inadequate in some cases. In particular, when the monitoring data value was below the SLA constraints, if the user's rating was far below the monitoring data, the credibility value was modified and in some case heightened irrationally, since the SLA agreement was breached, and the user has every right to be negative about the testbed's performance. Such misconceptions of the original

Co-funded by the Horizon 2020 Framework Programme of the European Union

credibility algorithm were corrected, and the following new credibility algorithm was developed and replaced the previous one in HRS:

```
Algorithm 1 User credibility mechanism.
Inputs: $UO,\ SD,\ MD$
Outputs: $CR,\ \widetilde{UO}$
for $\forall UO_i \in UO$ do
    $CASE1 \equiv (SD_i \geq UO_i) \wedge (SD_i \geq MD_i)$
    $CASE2 \equiv (SD_i \leq UO_i) \wedge (SD_i \leq MD_i)$
    $CASE3 \equiv (SD_i > UO_i) \wedge (SD_i \leq MD_i)$
    $CASE4 \equiv (SD_i \leq UO_i) \wedge (SD_i > MD_i)$
    $e_M = \frac{|MD_i - SD_i|}{SD_i},\ i = 1, \ldots, k$, Monitoring Relative Error
    $e_O = \frac{|UO_i - SD_i|}{SD_i},\ i = 1, \ldots, k$, Opinion Relative Error
    $e_D = \frac{|UO_i - MD_i|}{SD_i},\ i = 1, \ldots, k$, Relative Distance
    $C = [c_i]^\top,\ i = 1, \ldots, k$, Correction Vector
    if $CASE1 \vee CASE2$ then
        $c_i = 1 - e_O$
    else if $CASE3 \vee CASE4$ then
        $c_i = 1 - (e_{Mi} + e_{Oi})$
    end if
    $c_i = max(c_i, 0)$
end for
$\bar{c} = avg(c_i)$
$CR_n = \frac{(n-1)CR_{n-1} + \bar{c}}{n}$
for $\forall UO_i \in UO$ do
    $\widetilde{UO}_i = UO_i$
    if $CASE1 \vee CASE2$ then
        if $UO_i \leq MD_i$ then
            $\widetilde{UO}_i = MD_i + e_{Oi} CR_n$
        else if $UO_i < MD_i$ then
            $\widetilde{UO}_i = MD_i - e_{Oi} CR_n$
        end if
    end if
    if CASE3 then
        $\widetilde{UO}_i = MD_i - min(e_{Mi}, e_{Oi}) CR_n$
    end if
    if CASE4 then
        $\widetilde{UO}_i = MD_i + min(e_{Mi}, e_{Oi}) CR_n$
    end if
end for
```

Figure 16: HRS Credibility Mechanism

Co-funded by the Horizon 2020 Framework Programme of the European Union

The second update refers to the normalization of some QoS KPIs for the computation of the credibility score of the experimenter. Some metrics, such as response time, cannot be normalized, because there is no maximum value. In that case, for the inputs of the credibility mechanism we utilize the SLA violations to define an alternative KPI, which actually measures the fragment of time interval where no violation occurs, with the following formula,

$$KPI = 1 - \frac{violations}{samples}) \, 100.$$

4.2 SLA DEVELOPMENTS FOR THE SECOND CYCLE

The objective of the Fed4FIRE+ SLA Service is to provide the capability to testbed providers to offer QoS guarantees of their infrastructure to experimenters. The idea is that every time an experimenter initiates an experiment, an agreement is established between the testbed provider and the experimenter – in general, stating the total availability of the infrastructure during the total duration of the experiments.

The requirements for the development of this second cycle were already presented on deliverable D3.03 – Requirements and specifications for the second cycle, which are also summarized in *Requirements coverage* below.

The list of developments during the second cycle is:

- Development of assessment periods per agreement, overriding the platform assessment period (by default, one minute). The two additional assessment periods are periodic assessment (means that the agreement can be evaluated each given amount of time, e.g. 10 minutes, useful for detecting violations and proposing reactions); and assessment at the end, when the experiment has concluded (this is the assessment expected to be adopted by all testbeds). Both approaches were tested on NITOS and NETMODE testbeds;
- Violations now include information about the failing node, which provides additional information to Reputation component;
- Integration with NITOS and NETMODE testbeds tested;
- Design and start of development and integration of SLA lifecycle with jFed;
- Development of dashboard and collector.

4.2.1 SLA Service workflow

The sequence diagram below shows the involved workflow with regards to the SLA service when a user reserves an experiment. It starts when a user decides to reserve infrastructure in one of the Fed4FIRE+ testbeds. According to the user's needs, a number of testbeds can be available. Besides to the infrastructure, the testbed offers guarantees for their resources (in principle, resources availability during the experiment, but additional metrics could be added). These guarantees are stored in a document called SLA Template, which can be viewed by the user.

Co-funded by the Horizon 2020 Framework Programme of the European Union

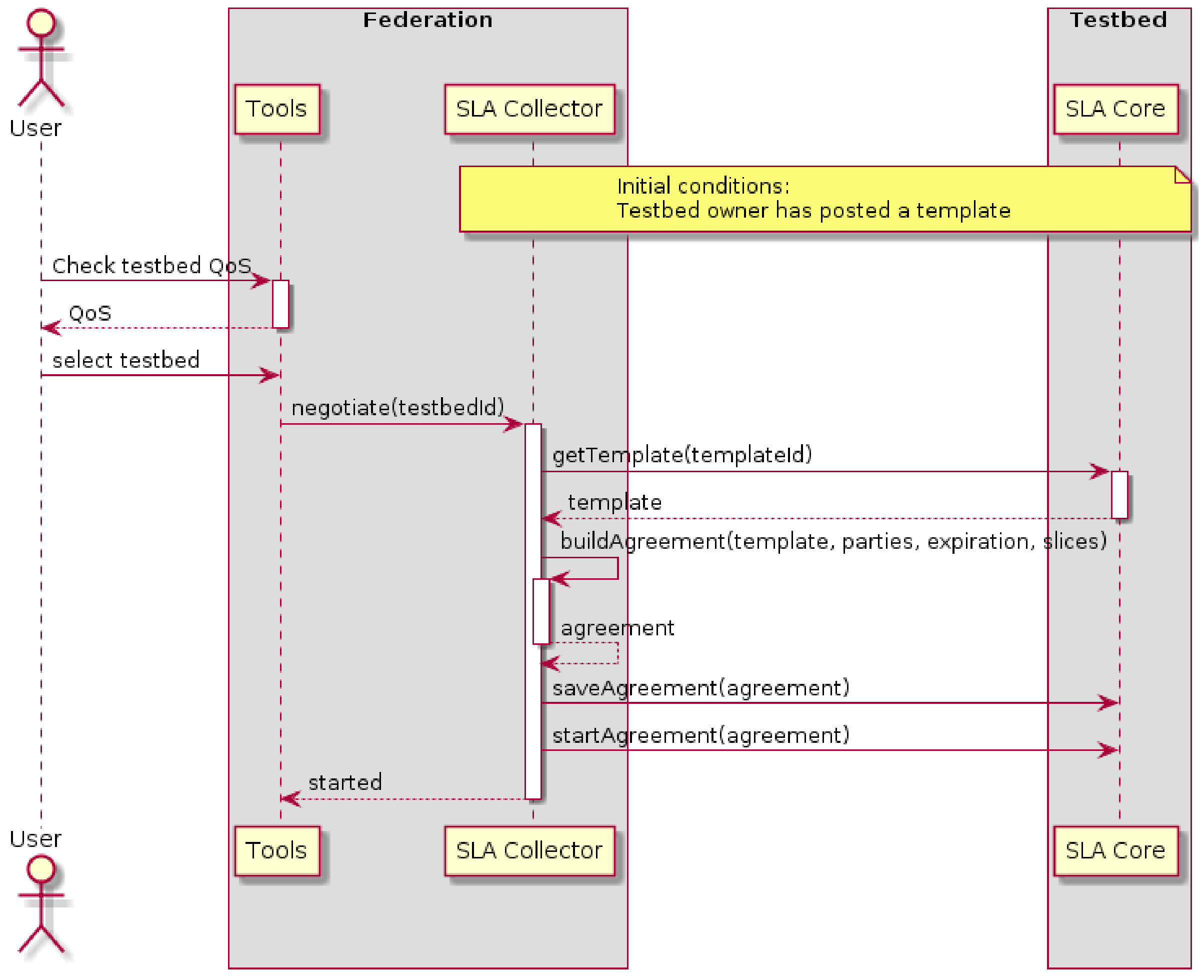


Figure 17: SLA Service Workflow

When the user selects a testbed and reserves the experiment, an agreement between the user and the testbed is created automatically by the system. This agreement, depending on the guarantees, is evaluated during the experiment or when the experiment ends. For example, the availability is evaluated after the experiment expiration.

A new requirement has been identified during the discussions to integrate the Reputation and SLA services in other testbeds. Users can extend the duration of an experiment; as such, when this happens, the federation tools must send a signal to the SLA Service in the testbed to also extend the duration of the agreement.

As a future requirement, we have identified that the SLA Service may provide alarms to the user and testbed in case one of the guarantees could not be fulfilled. For example, in the case of availability, if the availability in a period of 1 hour is lower than 98%, we can consider that a global 99% is compromised, and send an alarm in case this situation is detected.

Co-funded by the Horizon 2020 Framework Programme of the European Union

4.3 REQUIREMENTS COVERAGE

The requirements of the Reputation and SLA Service are already recorded in D3.3 and presented in the table below, which provides an explanation on the maturity achieved during the second cycle and the next steps for the third cycle. The requirements highlighted in green can be considered as completed.

ID	Title	Coverage in Second cycle
SLA_01	SLA solution must cover the whole lifecycle specified in WS-Agreement	An initial workflow for the whole lifecycle has been proposed. After discussing with testbed owners, an improved workflow has come out, which will be developed in the third cycle.
SLA_02	SLA solution Subscription mechanism	Delayed to 3rd cycle
SLA_03	SLA Dashboard	Dashboard implemented. Pending integration with federation tools.
SLA_04	Agreement creation and enactment	
SLA_05	Include node information in violation	
REP_01	Reputation Service access to monitoring and SLA data	An initial unified solution is discussed using the new federation portal. An early integration of Imec's testbeds is ongoing.

4.4 TESTBEDS INTEGRATION WITH REPUTATION AND SLA SERVICE

The core testbeds of the federation must be integrated with the Reputation and SLA service. Towards this direction, NTUA and ATOS partners documented and provided specific instructions for the integration. An experiment is uniquely defined by the slice URN, its starting and ending time and the involving testbeds. This information must be sent to the service's components in order to assess any SLA violations and update the reputation score of each involved testbed. This can be done in a unified way, in the new portal of the federation. Through this portal, SLA agreements will be conducted, and the experimenters will rate the conducted experiments. Then, the SLA and Reputation Service receive the information and requests for the agreements and the ratings respectively, the services will request the monitoring data from the testbeds involved in the experiment based on the unique slice URN.

For the computation of the reputation score of a testbed, no previous installation on the testbed is required. The only requirement is that the testbeds expose a monitoring data API for the

Co-funded by the Horizon 2020 Framework Programme of the European Union

procedure mentioned above. After the completion of the experiment, the user is prompt to evaluate all the involved testbeds and provide the ranking for the QoS and QoE metrics. After the experimenter rates the QoS and QoE metrics through the portal's GUI, the portal will send as a JSON POST request to the reputation service the user's evaluation. The JSON body section referring to the users evaluation will have the following format.

```
"user_eval":[{"Usability":"very high", "Sup Satisfaction":"very high", "Doc Readability":"very high", "Operability":"very high","Availability":100, "Response":87},{"Usability":"very high", "Sup Satisfaction":"very high", "Doc Readability":"very high", "Operability":"very high","Availability":100, "Response":55}]
```

The above example contains two evaluations, since the experiment used resources from two testbeds and the experimenter must rate each involving experiment.

The Reputation Engine will compute the new values of the testbed's reputation and the user's credibility using the evaluation received from the federation's portal and the monitoring data it will request sequentially from the involved testbeds APIs. The updated testbed's reputation score will be available on the portal and combined with the definitions of the KPIs in order to facilitate future experimenters on the resource selection and compare different testbeds based on their performance on specific metrics.

The integration of a testbed with the SLA service requires initially the installation of the SLA management module. This component includes several subcomponents, such as the Repository and the Assessment modules, that are responsible for maintaining the information about the agreements, the penalties, the violations and the templates and for assessing the QoS performance of the service based on specific KPIs. The SLA management module will collect the monitoring data either periodically or aggregated at the end of the experiment in order to assess the provided service from the monitoring data with the exact same process described previously. The only difference is the intervals of the monitoring data requests.

Imec partner works on providing the monitoring API for all Imec Testbeds, in order to begin the interconnection of the service with the core testbeds. Also, Imec will update of the portal front-end and back-end in order to expose to the experimenters reputation and SLA values to the experimenters and also connect both services with the portal and allow the users to use SLA agreements and rate the resources chosen for their experiment.

The above procedure and the testing of the new developments will be the guideline for the easy and smooth interconnection of the rest core testbeds with the SLA and Reputation Service.

Co-funded by the Horizon 2020 Framework Programme of the European Union

4.5 FUTURE WORK

The following steps of the Reputation and SLA service can be summarized as follows.

- The completion of the development and deployment of the monitoring data APIs of all core testbeds. The, the interconnection with the Reputation and SLA service will follow.
- Installation of SLA Manager component in Imec (and other) testbeds. Integration of jFed and SLA system, allowing a user to consult SLA of testbeds and create resources with an associated SLA. Adaptations in SLA dashboard and SLA collector components are expected as part of this integration.
- Implementation of the subscription mechanism, which has been deferred to 3rd cycle.
- Development of the warnings concept: measures of resources availability over a short period of time (e.g. 1 hour) that can be used as a warning for the testbed provider and the experimenter that the agreement may not be fulfilled.
- Implement the renegotiation concept, which corresponds to extending the expiration time of the agreement when an experimenter extends the duration of an experiment.

Co-funded by the Horizon 2020
Framework Programme of the European Union

5 IMPROVING REPRODUCIBILITY OF EXPERIMENTS – EXPERIMENT-AS-A-SERVICE

5.1 EXPERIMENT SPECIFICATION (ESPEC)

The Experiment Specification (ESpec) was developed as a new standard for setting up experiments. It combines various existing industry standards and leverages them to make it easier to fully setup an experiment: from requesting and provisioning the necessary testbed resources to installing software, doing the configuration management and the application deployment. This is also documented towards the users at https://jfed.ilabt.imec.be/espec .

In this way, the ESpec can be used as a base for creating "Experiments-as-a-Service", where we provide experimenters with fully automated experiments that provide an excellent starting point for doing their scientific research or education activities. See e.g. 12 automated openstack deployment in this deliverable and other examples in D3.2.

The functionality of this ESpec can also be leveraged to automate continuous testing of the Fed4FIRE+ testbed resources, and the software platforms which have been developed on it. This allows the developers of these platform to detect breaking changes from the moment they happen, which greatly simplifies debugging and decreases the effort needed to sustain these platforms.

The Experiment Specification is not a replacement for the Resource Specification (RSpec) format. Instead, it acts as a bundle (see Figure 18) for an RSpec – which defines the testbed resources that are needed for the experiment – with additional files for the software deployment and configuration. For that second part, we use Ansible: a widely used open source software that automates software provisioning, configuration management and application deployment. As Ansible connects via SSH to the servers it controls and doesn't need an "agent" to be present on these servers, it is a natural fit for controlling Fed4FIRE+ testbed servers.

The ESpec also provides the necessary glue to make Ansible work: it can generate the necessary configuration files for Ansible, like the inventory-file and an SSH private key for accessing the other servers, and upload them to the Ansible master-node.

For more details about the ESpec, see D3.2 and https://jfed.ilabt.imec.be/espec.

In this cycle the ESpec was extended with some smaller features based on user requests (e.g. possibility to use branches in the git dialog box) and bug fixes (e.g. ESpec in jFed didn't use correctly ssh proxy for some testbeds). And it was used for the Openstack deployment example discussed later on in this deliverable.

Co-funded by the Horizon 2020 Framework Programme of the European Union

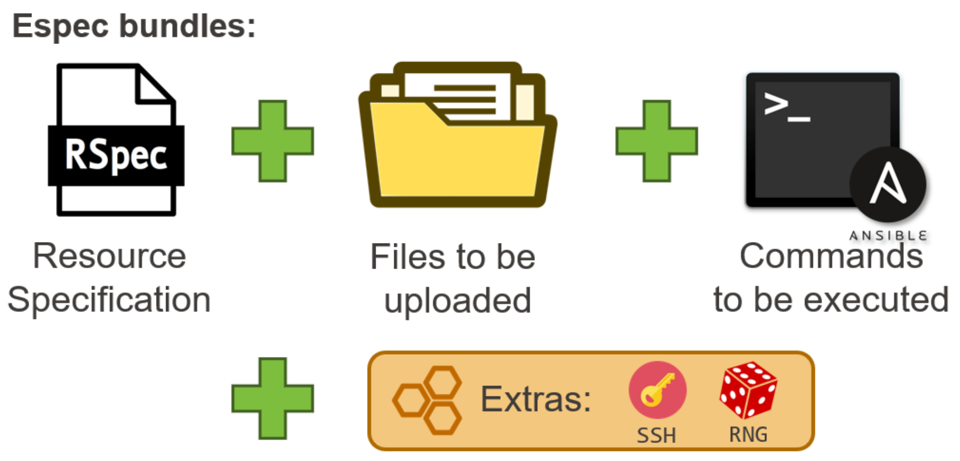


Figure 18: ESpec bundles RSpec, files to be uploaded and scripts

5.2 LIGHTWEIGHT EXPERIMENT ORCHESTRATION TOOL (EXPO)

The ESpec discussed in the previous section helps in provisioning an experiment, but is not meant/usable for the experimentation orchestration itself.

In Fed4FIRE we had OMF[1] (https://www.rubydoc.info/github/mytestbed/omf/file/README.mkd) as an experiment orchestration tool, but it is currently not supported anymore (e.g. the original website does not respond for the last couple of years: http://omf.mytestbed.net/projects/omf/wiki/Introduction.

However, there is a demand for an experiment orchestration tool, especially for wireless experiments at scale. Experimenters want to be able to turn on and off things, change parameters at fixed times, for a large number of nodes.

That's the reason we started the development of a lightweight tool ExpO (Experiment Orchestrator) to help in experiment orchestration. Lightweight means that it only contains the minimal features and a minimal number of components, so it's very simple to install and use.

Documentation and code can be found at https://gitlab.ilabt.imec.be/ilabt/expo

[1] Thierry Rakotoarivelo, Maximilian Ott, Guillaume Jourjon, and Ivan Seskar. 2010. OMF: a control and management framework for networking testbeds. SIGOPS Oper. Syst. Rev. 43, 4 (January 2010), 54–59. DOI:https://doi.org/10.1145/1713254.1713267

Co-funded by the Horizon 2020 Framework Programme of the European Union

5.2.1 About ExpO

ExpO is short for "Experiment Orchstrator". It allows you to run time-sensitive experiments over multiple machines.

ExpO consists out of two pieces of software:

- the **ExpO slave** which is present on all machines participating in the experiment, waiting for instructions on when to execute commands
- the **ExpO director** which executes experiments defined in an *Experiment Orchestration definition*

5.2.2 Installation

```
pip install git+https://gitlab.ilabt.imec.be/ilabt/expo.git

for ubuntu18:

apt-get update

apt-get install python3-pip

pip3 install git+https://gitlab.ilabt.imec.be/ilabt/expo.git
```

5.2.3 Dependencies

ExpO requires Python 3.6 or higher.

ExpO uses MQTT to communicate between the *Director* and the *Slaves*. The MQTT-server is typically installed on the same server as the *Director*. We recommend Eclipse Mosquitto as MQTT-server

```
apt-get install mosquitto
```

5.2.4 Usage

5.2.4.1 Slave

An ExpO slave must be present on each machine that takes part in the experiment

```
expo-slave -b <broker-url>
```

5.2.4.2 Director

To start an experiment, specify an experiment_id and the *Experiment Orchestrator definition*-file:

```
expo-director start <experiment_id> < my-expo-definition.yaml
```

Co-funded by the Horizon 2020 Framework Programme of the European Union

The director will first verify that all defined nodes are online. It will then configure them to register themselves in the correct groups. Once all nodes are correctly configured, it will start executing the commands.

5.2.5 The *Experiment Orchestration Definition*-file

The *Experiment Orchestration Definition* is a YAML based file.

Example:

```
version: 1.0
nodes:
  node1: [groupA]
  node2: groupB
  node3:
    - groupA
    - groupB
  node4
commands:
  - command: "iperf -s"
    groups: [groupA]

  - after: 10
    id: iperf_client
    command: "iperf -c server"
    groups: [groupB]

  - command: echo "Hello $EXAMPLE_NAME"
    args:
      chdir: /tmp
      environment:
        EXAMPLE_NAME: 'World'
    nodes: node4
```

TIP: YAML has some powerful features like Anchors and Aliases, which allows you to reference values defined elsewhere in the file. You can use these to enrich your definition file.

5.2.5.1 The file format

`version` Currently always 1.0

`nodes` List of nodes expected in the experiment. Optionally you can specify to which groups the node belongs in this experiment

`commands`: List of commands to be executed

5.2.5.2 Command format

`command`: the command to execute

`after`: number of seconds after the previous command that this command must be executed (optional, if omitted, it will be executed immediately after the previous command)

`groups`: the groups which must execute the command (optional if `nodes` has been specified)

`nodes`: the nodes which must execute the command (optional if `groups` has been specified)

`id`: a custom ID to identify MQTT-messages which relate to this command, such as the result, stdout, stderr, stdin (optional, if omitted, the index of this command in the `commands`-list is used as an id)

`stderr`: whether to stream the stderr as MQTT-messages with topic `expo/<experiment_id>/node/<node_id>/cmd/<command_id>/stderr` (defaults to True)

`stdout`: whether to stream the stdout as MQTT-messages `expo/<experiment_id>/node/<node_id>/cmd/<command_id>/stdout` (defaults to False)

`stdin`: whether to stream data sent to MQTT topic `expo/<experiment_id>/node/<node_id>/cmd/<command_id>/stdin` or `expo/<experiment_id>/group/<group_id>/cmd/<command_id>/stdin` to the stdin of the process (defaults to False)

Warning: the order in which data is streamed to the stdin of a process is indeterminate when mixing both topics

Co-funded by the Horizon 2020
Framework Programme of the European Union

6 FEDERATION MONITORING

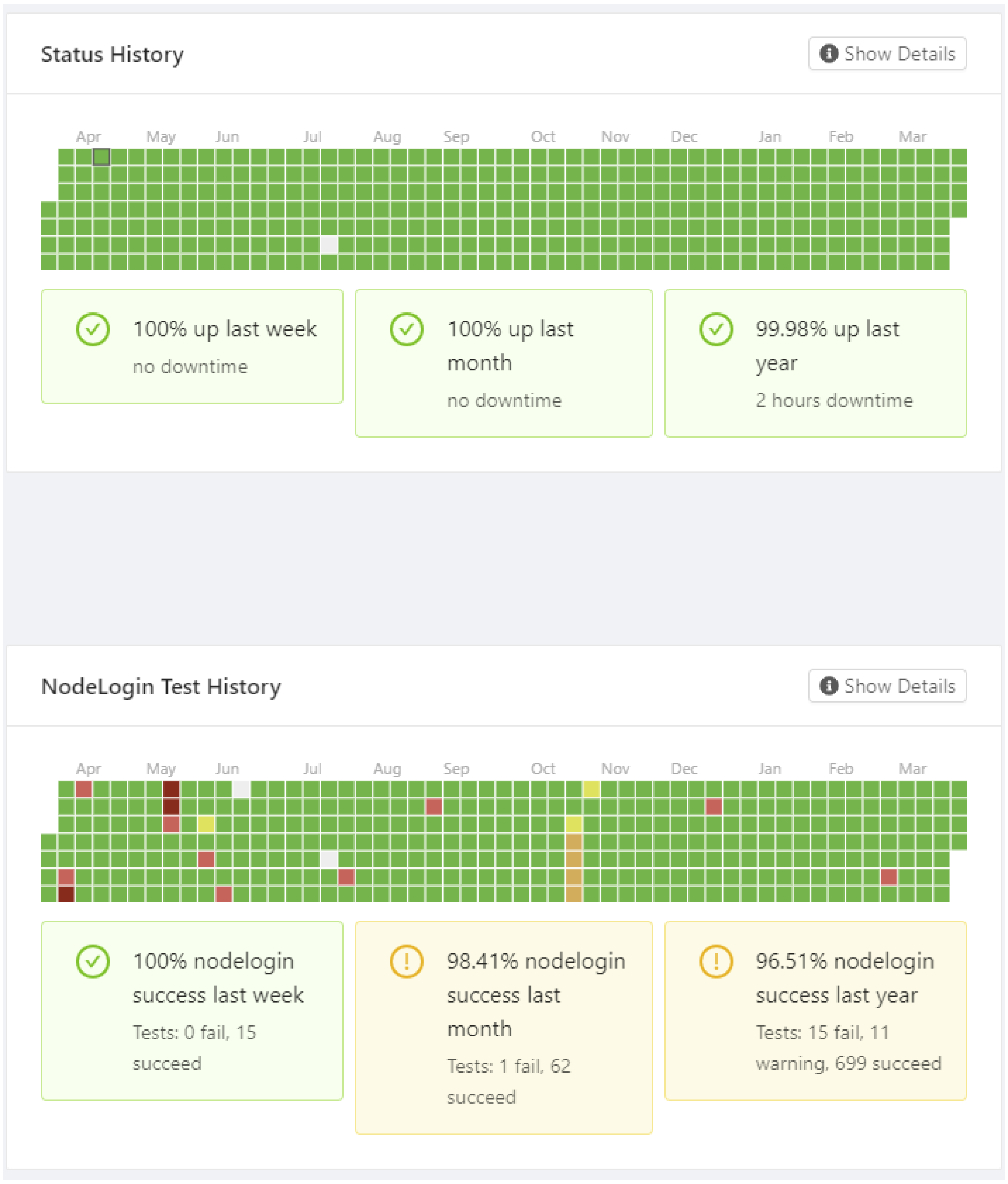
6.1 INTRODUCTION

Because the federation of testbed is a ‘loose’ federation (there is no single entity controlling all testbeds), continuous monitoring of the testbeds is key. In that way, both the testbed operators can be warned if something goes wrong and the experimenters can have an overview of which testbeds are okay to use. D3.2 has an extensive overview of the Federation Monitoring 2.0 that was put in place for cycle 1: https://fedmon.fed4fire.eu.

6.2 MORE DETAILED HISTORY OVERVIEW

We had already an overview of the last year for login tests and the API status, but now we added also percentages and the number of tests, so we have a clear view per testbed on the availability. We added also the absolute numbers, because the login tests only twice a day, so one failed test in a week has an impact of 7%.

Co-funded by the Horizon 2020 Framework Programme of the European Union



100% up last week
no downtime

100% up last month
no downtime

99.98% up last year
2 hours downtime

100% nodelogin success last week
Tests: 0 fail, 15 succeed

98.41% nodelogin success last month
Tests: 1 fail, 62 succeed

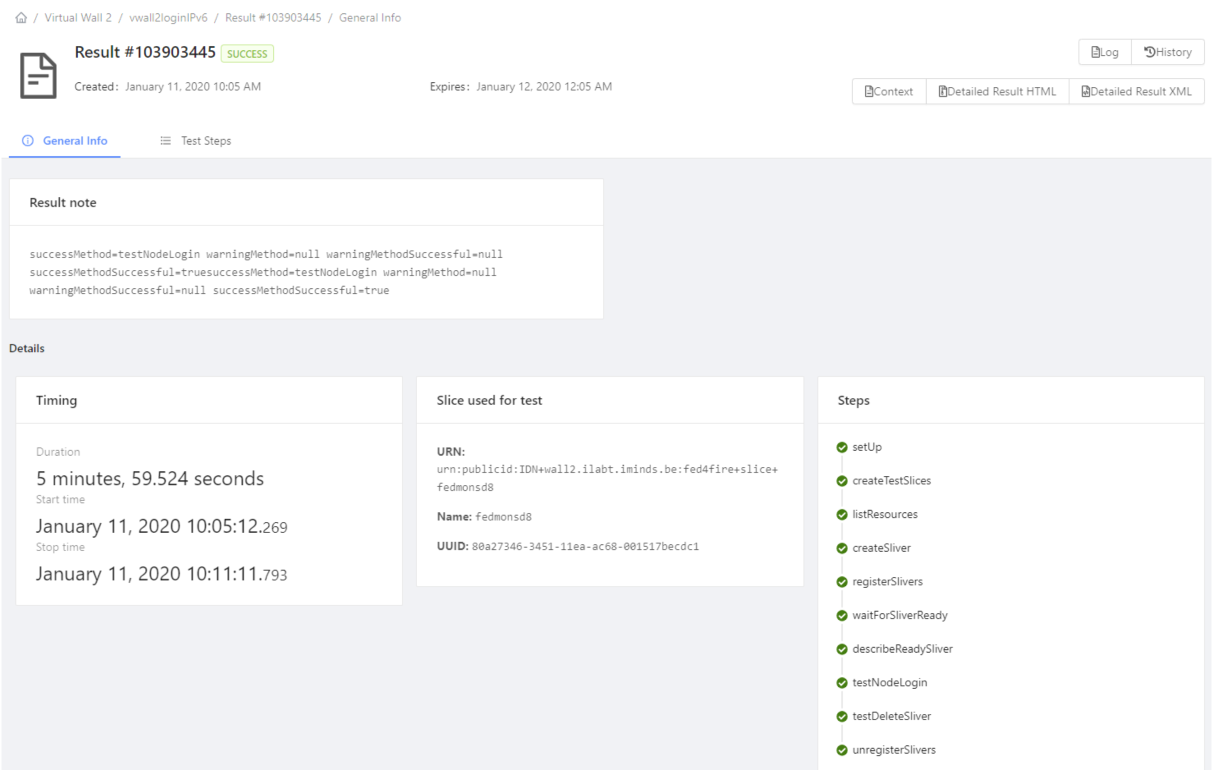
96.51% nodelogin success last year
Tests: 15 fail, 11 warning, 699 succeed

Figure 19: Detailed statistics on uptime and API availability for the last year (per testbed)

Co-funded by the Horizon 2020 Framework Programme of the European Union

6.3 IMPROVED TEST REPORTS

The test reports have been improved to give a better overview, see some examples below.



Result #103903445 SUCCESS

Created: January 11, 2020 10:05 AM

Expires: January 12, 2020 12:05 AM

Result note

```
successMethod=testNodeLogin warningMethod=null warningMethodSuccessful=null
successMethodSuccessful=truesuccessMethod=testNodeLogin warningMethod=null
warningMethodSuccessful=null successMethodSuccessful=true
```

Details

Timing

Duration
5 minutes, 59.524 seconds
Start time
January 11, 2020 10:05:12.269
Stop time
January 11, 2020 10:11:11.793

Slice used for test

URN: `urn:publicid:IDN+wall2.ilabt.iminds.be:fed4fire+slice+fedmonsd8`

Name: `fedmonsd8`

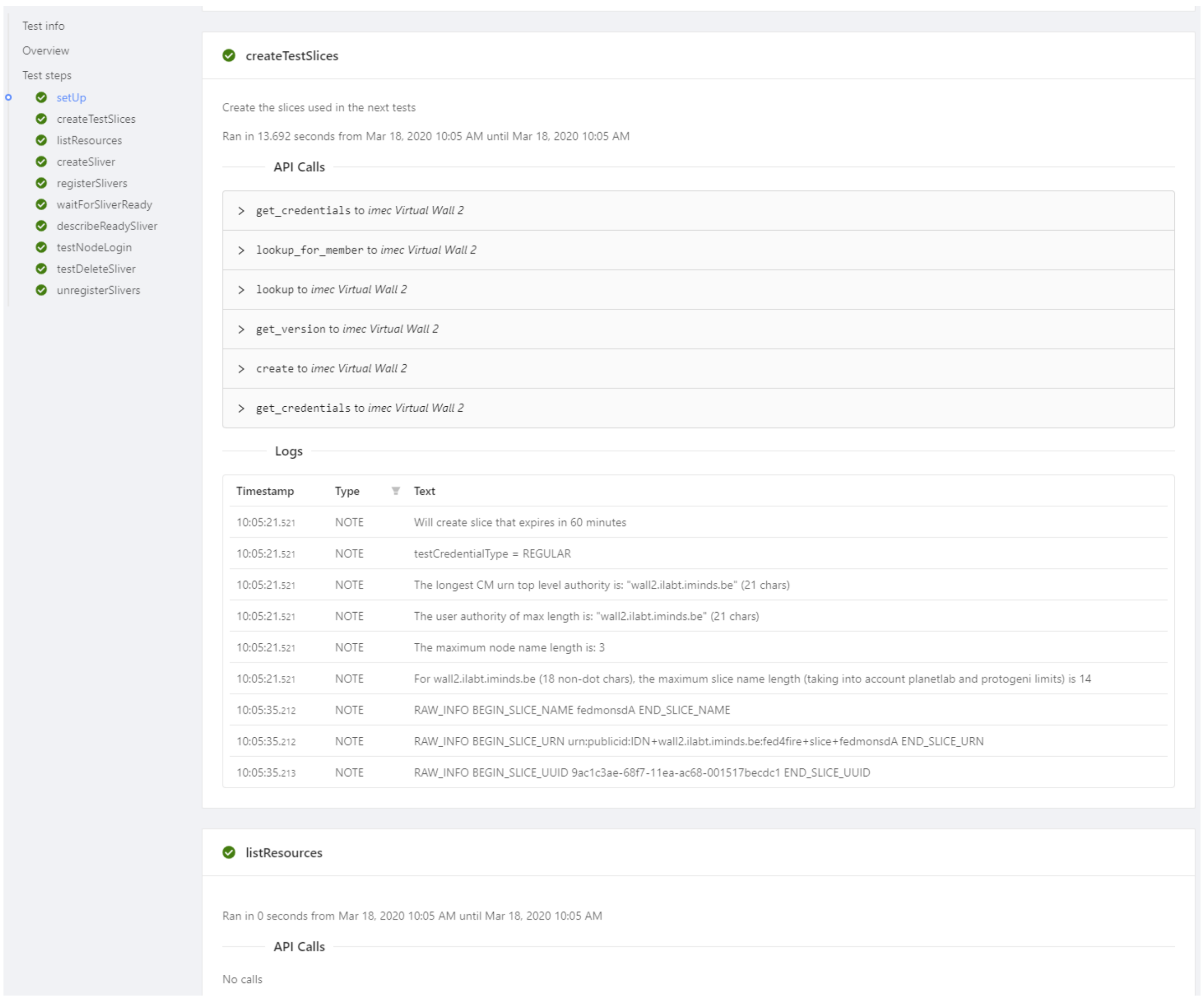
UUID: `80a27346-3451-11ea-ac68-001517becdc1`

Steps

- ✓ setUp
- ✓ createTestSlices
- ✓ listResources
- ✓ createSliver
- ✓ registerSlivers
- ✓ waitForSliverReady
- ✓ describeReadySliver
- ✓ testNodeLogin
- ✓ testDeleteSliver
- ✓ unregisterSlivers

Figure 20: Overview of Fedmon test report

Co-funded by the Horizon 2020
Framework Programme of the European Union


Test info
Overview
Test steps
- setUp
- createTestSlices
- listResources
- createSliver
- registerSlivers
- waitForSliverReady
- describeReadySliver
- testNodeLogin
- testDeleteSliver
- unregisterSlivers

✓ createTestSlices

Create the slices used in the next tests

Ran in 13.692 seconds from Mar 18, 2020 10:05 AM until Mar 18, 2020 10:05 AM

API Calls

> `get_credentials` to *imec Virtual Wall 2*
> `lookup_for_member` to *imec Virtual Wall 2*
> `lookup` to *imec Virtual Wall 2*
> `get_version` to *imec Virtual Wall 2*
> `create` to *imec Virtual Wall 2*
> `get_credentials` to *imec Virtual Wall 2*

Logs

Timestamp	Type	Text
10:05:21.521	NOTE	Will create slice that expires in 60 minutes
10:05:21.521	NOTE	testCredentialType = REGULAR
10:05:21.521	NOTE	The longest CM urn top level authority is: "wall2.ilabt.iminds.be" (21 chars)
10:05:21.521	NOTE	The user authority of max length is: "wall2.ilabt.iminds.be" (21 chars)
10:05:21.521	NOTE	The maximum node name length is: 3
10:05:21.521	NOTE	For wall2.ilabt.iminds.be (18 non-dot chars), the maximum slice name length (taking into account planetlab and protogeni limits) is 14
10:05:35.212	NOTE	RAW_INFO BEGIN_SLICE_NAME fedmonsdA END_SLICE_NAME
10:05:35.212	NOTE	RAW_INFO BEGIN_SLICE_URN urn:publicid:IDN+wall2.ilabt.iminds.be:fed4fire+slice+fedmonsdA END_SLICE_URN
10:05:35.213	NOTE	RAW_INFO BEGIN_SLICE_UUID 9ac1c3ae-68f7-11ea-ac68-001517becdc1 END_SLICE_UUID

✓ listResources

Ran in 0 seconds from Mar 18, 2020 10:05 AM until Mar 18, 2020 10:05 AM

API Calls

No calls

Figure 21: Detailed steps of a test report

Co-funded by the Horizon 2020
Framework Programme of the European Union

6.4 BUG FIXES

Apart from the above, a number of small improvements and fixes have been done as well for the Fedmon system.

- Bugfixes related to ESpec and to multi-slice tests
- Included ESpec logs to test output
- Improved testbed ping test robustness and feedback
- Reuse fedmon slice names to put less burden on the systems (especially the authority)
- Improved GPULab test
- Showing intermediate results of expired tests
- Fixed health calculation bug
- Increased the storage time of the fedmon tests, so we can have statistics over longer time
- Upgraded to java 11
- Fix for Fedmon false mails

Co-funded by the Horizon 2020
Framework Programme of the European Union

7 INTERCONNECTIVITY

Based on demand of experimenters we have extended the international connectivity in three ways.

7.1 INTERCONNECTION OF CITYLAB TESTBED WITH EXOGENI

For an experiment, there was need for a quick setup of a layer 2 connection between the Citylab testbed in Antwerp and exogeni in Amsterdam. We used the existing Software Defined eXchange (SDX) at imec Gent to make this connection as can be seen below. The GRE tunnel runs over the public research network, but both Citylab and the SDX in Gent are connected to Belnet (Belgian NREN) with 10Gb/s links, so this is not a bottleneck.

We verified also the usable bandwidth (Figure 23) and we see that the link between Amsterdam and Gent has a perfect throughput (930Mb/s without TCP/IP headers), while the link between Gent and Antwerp only delivers about 830Mb/s at its maximum (4 parallel TCP connections). It seems that this bottleneck is caused by the (light) hardware of the wireless nodes at Antwerp, but of course this is more than enough compared to the wireless bandwidth needed for the experiment.

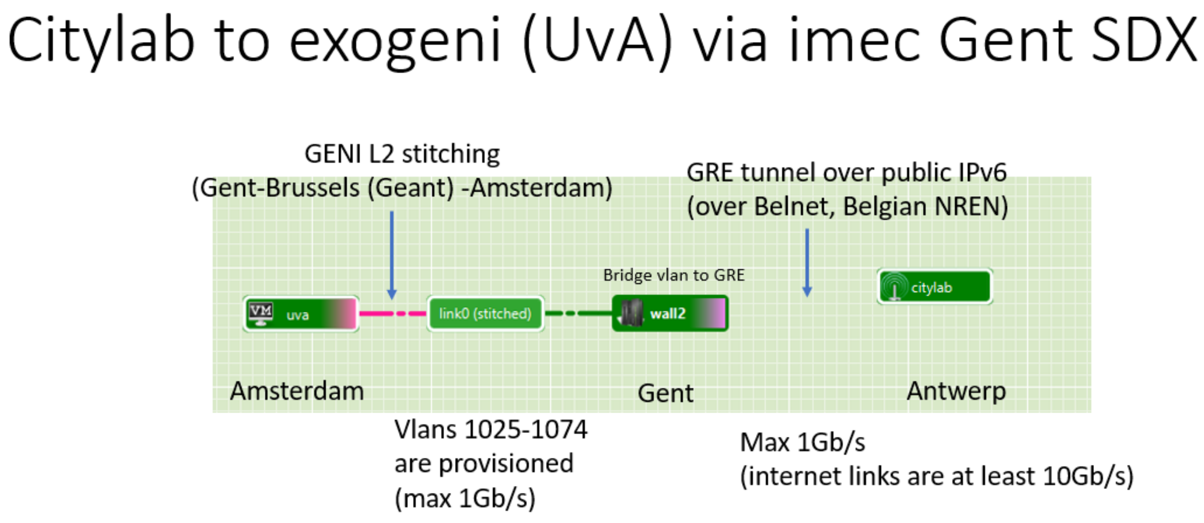

Citylab to exogeni (UvA) via imec Gent SDX
GENI L2 stitching
(Gent-Brussels (Geant) -Amsterdam)
GRE tunnel over public IPv6
(over Belnet, Belgian NREN)
Bridge vlan to GRE
VM
uva
link0 (stitched)
wall2
citylab
Amsterdam
Gent
Antwerp
Vlans 1025-1074
are provisioned
(max 1Gb/s)
Max 1Gb/s
(internet links are at least 10Gb/s)

Figure 22: Interconnection of Citylab with Exogeni via imec SDX

Co-funded by the Horizon 2020 Framework Programme of the European Union

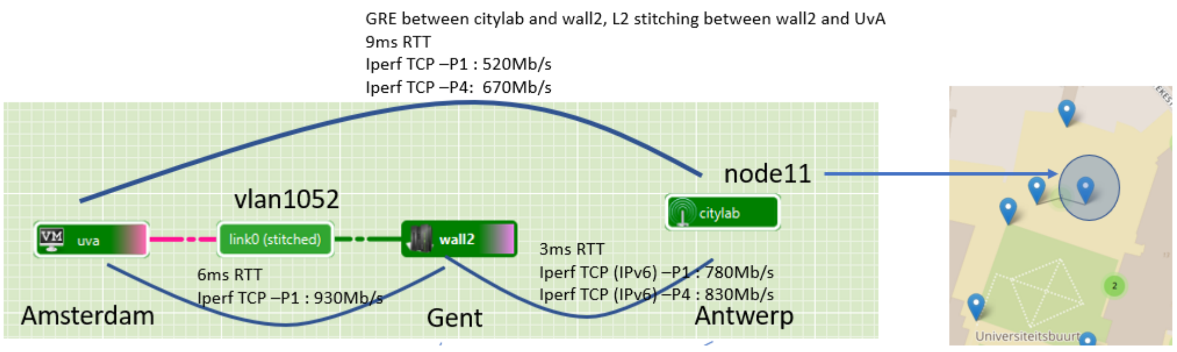
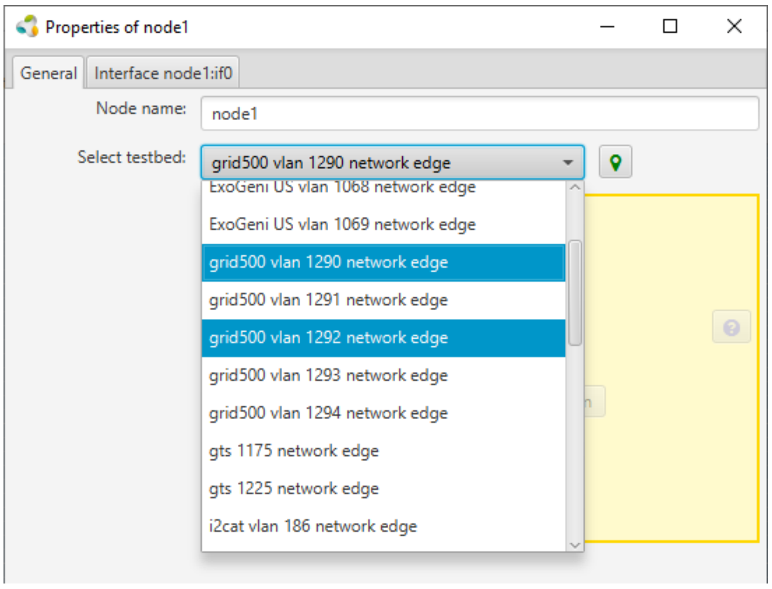


Figure 23: Verification of usable bandwidth between Citylab and exogeni

7.2 LAYER 2 CONNECTIONS BETWEEN VIRTUAL WALL AND GRID5000

During this 2nd cycle we have also enabled and debugged 5 vlans between Grid5000 and the Virtual wall testbed. These are available in jFed as external networks. The goal is to make these also available through the Grid5000 federated testbed.



Node name: node1

Select testbed: grid500 vlan 1290 network edge

ExoGeni US vlan 1068 network edge
ExoGeni US vlan 1069 network edge
grid500 vlan 1290 network edge
grid500 vlan 1291 network edge
grid500 vlan 1292 network edge
grid500 vlan 1293 network edge
grid500 vlan 1294 network edge
gts 1175 network edge
gts 1225 network edge
i2cat vlan 186 network edge

Figure 24: 5 new vlans between Grid5000 and Virtual Wall

Co-funded by the Horizon 2020
Framework Programme of the European Union

7.3 USING EXOGENI FLUKES WITH A FED4FIRE ACCOUNT FOR COMPLEX CONNECTIVITY TO EXOGENI

7.3.1 Overview

The Exogeni testbeds (see Figure 25) can be used in two different ways:

- Through the standard GENI AM API (the API that we also use in Fed4FIRE) and with standard GENI/Fed4FIRE stitching
- Through a proprietary API, via the Exogeni Flukes tool (http://www.exogeni.net/2015/09/exogeni-getting-started-tutorial/)

Most of the default functionality (for cloud experiments, etc) can be done through the GENI AM API (and thus be used via jFed).

However, very complex network connectivity between the Exogeni testbeds is only available through the proprietary API and the Exogeni Flukes tool. To give an example: automated stitching between the Virtual Wall and Exogeni Amsterdam can be done through GENI/Fed4FIRE stitching with jFed. But for an ongoing experiment we needed direct layer 2 connectivity from the imec SDX in Gent to the US. Vlans were setup by the exogeni people from the US to Amsterdam and with half-way stitching we can now set up the part in Europe with jFed, and the US part with the Flukes tool. But together with the Exogeni people we made it possible to use a Fed4FIRE account for this.

This means that for the complex connectivity you need two different tools, but only a single account.

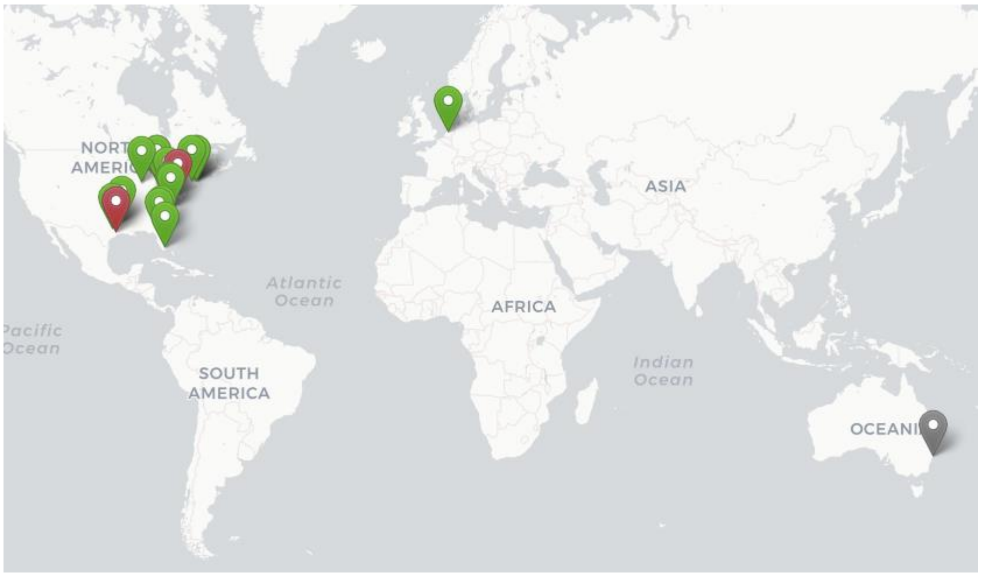
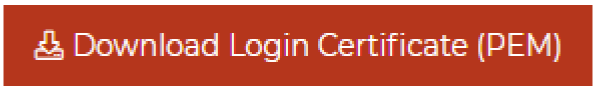


Figure 25: Map of Exogeni testbeds

Co-funded by the Horizon 2020 Framework Programme of the European Union

7.3.2 Download the PEM credential file from the portal

Download the PEM file from the Fed4FIRE portal (bottom of the dashboard after you login):



7.3.3 Flukes install and configuration

Follow http://www.exogeni.net/2015/09/exogeni-getting-started-tutorial/, from install/configure flukes.

It is possible you need to add an exception to the java security list, where you add http://geni-images.renci.org

.flukes.properties file (point certfile and certkeyfile to the downloaded pem file, ssh keys should also point to local files):

```
orca.xmlrpc.url=https://geni.renci.org:11443/orca/xmlrpc
user.certfile=k:/flukes/login_ilabt_imec_be_bvermeul@ugent.be.pem
user.certkeyfile=k:/flukes/login_ilabt_imec_be_bvermeul@ugent.be.pem
enable.modify=true
ssh.key=~/.ssh/id_dsa
# SSH Public key to install into VM instances
ssh.pubkey=~/.ssh/id_dsa.pub
# Secondary login (works with ssh.other.pubkey)
ssh.other.login=bvermeul
# Secondary public SSH keys
ssh.other.pubkey=~/.ssh/id_dsa.pub
# Should the secondary account have sudo privileges
ssh.other.sudo=yes
# Path to XTerm executable on your system
xterm.path=/opt/X11/bin/xterm
```

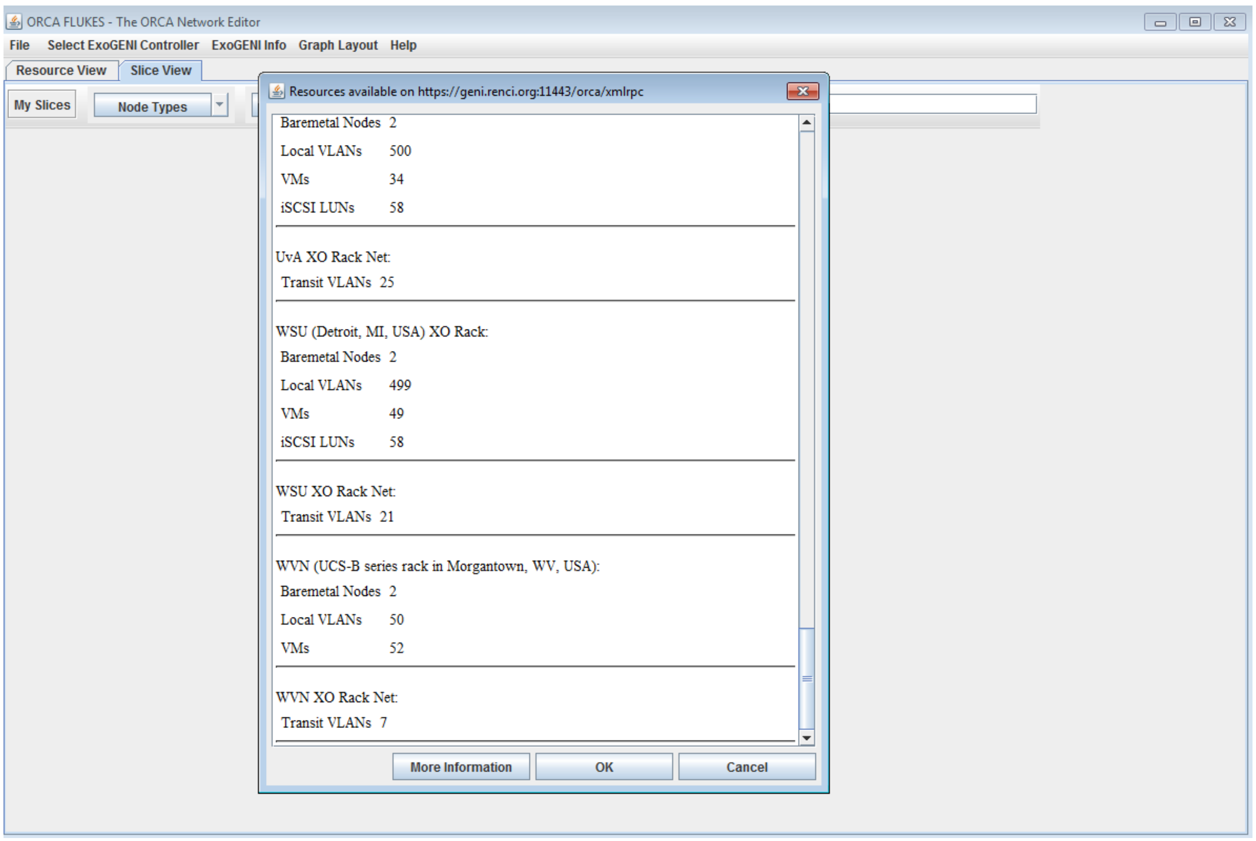
copy this .flukes.properties to your homedir (e.g. c:\users\administrator on windows)

Co-funded by the Horizon 2020 Framework Programme of the European Union

7.3.4 Run Flukes and test

```
Javaws flukes.jnlp
```

To test the account, click exogeni info, available resources. It will ask the password of the pem file, but it's just empty, click ok. It can take some time but should not give an error, but the below screen:


ORCA FLUKES - The ORCA Network Editor
File
Select ExoGENI Controller
ExoGENI Info
Graph Layout
Help
Resource View
Slice View
My Slices
Node Types
Resources available on https://geni.renci.org:11443/orca/xmlrpc
More Information
OK
Cancel

```
 Baremetal Nodes  2
 Local VLANs      500
 VMs              34
 iSCSI LUNs       58

UvA XO Rack Net:
 Transit VLANs  25

WSU (Detroit, MI, USA) XO Rack:
 Baremetal Nodes  2
 Local VLANs      499
 VMs              49
 iSCSI LUNs       58

WSU XO Rack Net:
 Transit VLANs  21

WVN (UCS-B series rack in Morgantown, WV, USA):
 Baremetal Nodes  2
 Local VLANs      50
 VMs              52

WVN XO Rack Net:
 Transit VLANs  7
```

Figure 26: Exogeni Flukes GUI with Fed4FIRE account

7.3.5 Using Flukes

See further at http://www.exogeni.net/2015/09/exogeni-getting-started-tutorial/

Co-funded by the Horizon 2020 Framework Programme of the European Union

7.3.6 Specific stitching to Fed4FIRE

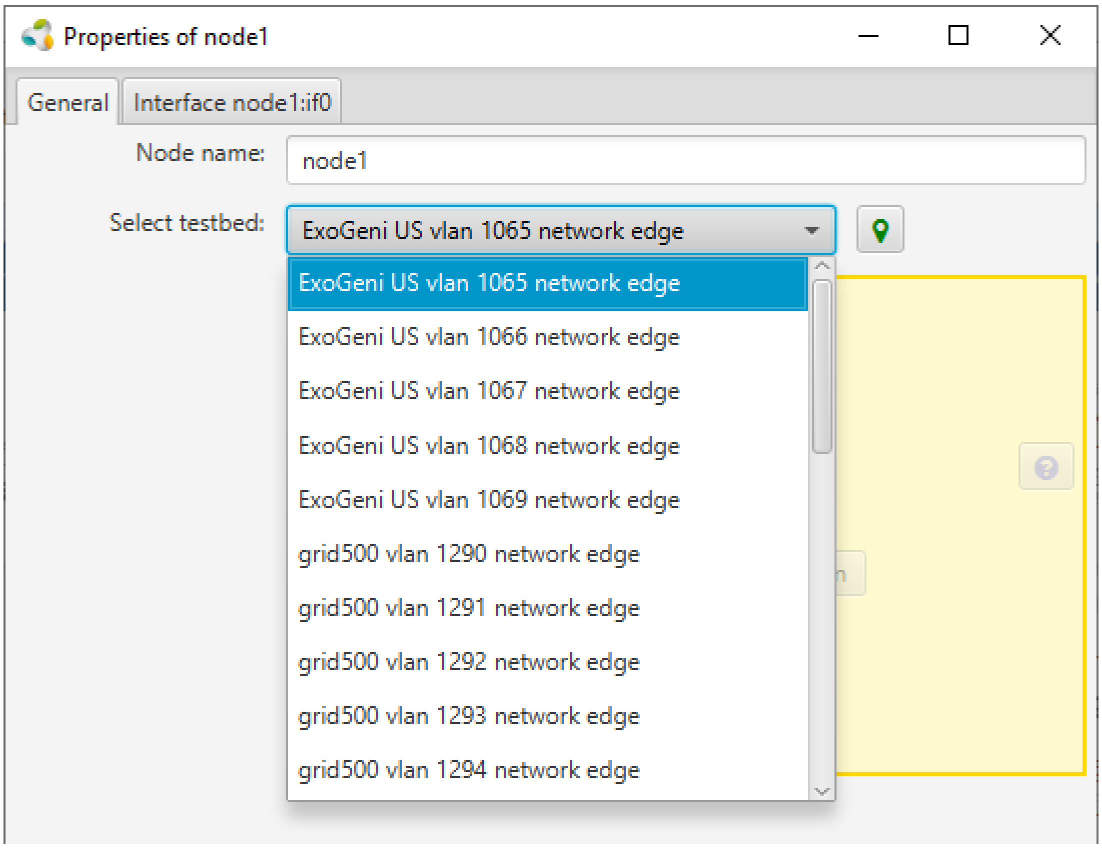
Put a stitchport in your layout, with the following info:

Port URL: http://geni-orca.renci.org/owl/uvanlNet.rdf#Uva-nlNet/Force10/S4810/TenGigabitEthernet/0/2/ethernet

Label/tag (vlan number): 1065 (valid 1065-1069)

7.3.7 Fed4FIRE side

Put a dedicated external network connection in your layout and chose US Exogeni 1065-1069.



Properties of node1

General | Interface node1:if0

Node name: node1

Select testbed: ExoGeni US vlan 1065 network edge

ExoGeni US vlan 1065 network edge
ExoGeni US vlan 1066 network edge
ExoGeni US vlan 1067 network edge
ExoGeni US vlan 1068 network edge
ExoGeni US vlan 1069 network edge
grid500 vlan 1290 network edge
grid500 vlan 1291 network edge
grid500 vlan 1292 network edge
grid500 vlan 1293 network edge
grid500 vlan 1294 network edge

Figure 27: Exogeni US vlans in jFed

Co-funded by the Horizon 2020 Framework Programme of the European Union

8 SERVICE ORCHESTRATION (YOUREPM)

The objective of the Fed4FIRE+ YourEPM is to provide a tool that allows experimenters to design and execute business processes modeled in BPMN, so experimenters can design cross-testbed processes or execute 3rd parties' services. For a complete description of the orchestration solution, refer to D3.1.

The requirements for the development of this second cycle were presented on deliverable D3.3 – Requirements and specifications for the second cycle, which are also summarized in *Requirements coverage* below.

The list of developments during the second cycle is:

- Integrated version with latest improvements, including enhanced multitenancy;
- Support for F4F OAuth2 authentication, instead of a certificate-based mechanism. Regular and admin users are supported. Additionally, we defined a super admin user, capable of reviewing all defined workflows in the platform.
- YourEPM has been installed in the federation.

8.1 REQUIREMENTS COVERAGE

The requirements defined for the component are presented in the table below, which provides an explanation on the maturity achieved during the second cycle and the next steps for the third cycle. The requirements highlighted in green can be considered complete.

ID	Title	Coverage in Second cycle
YourEPM-03	Assistant in the selection of the exposed services.	Work in progress
YourEPM-08	Integration with the authentication and authorization of Fed4FIRE+	YourEPM authorizes user by using OAuth2 service

Co-funded by the Horizon 2020 Framework Programme of the European Union

8.2 ARCHITECTURE

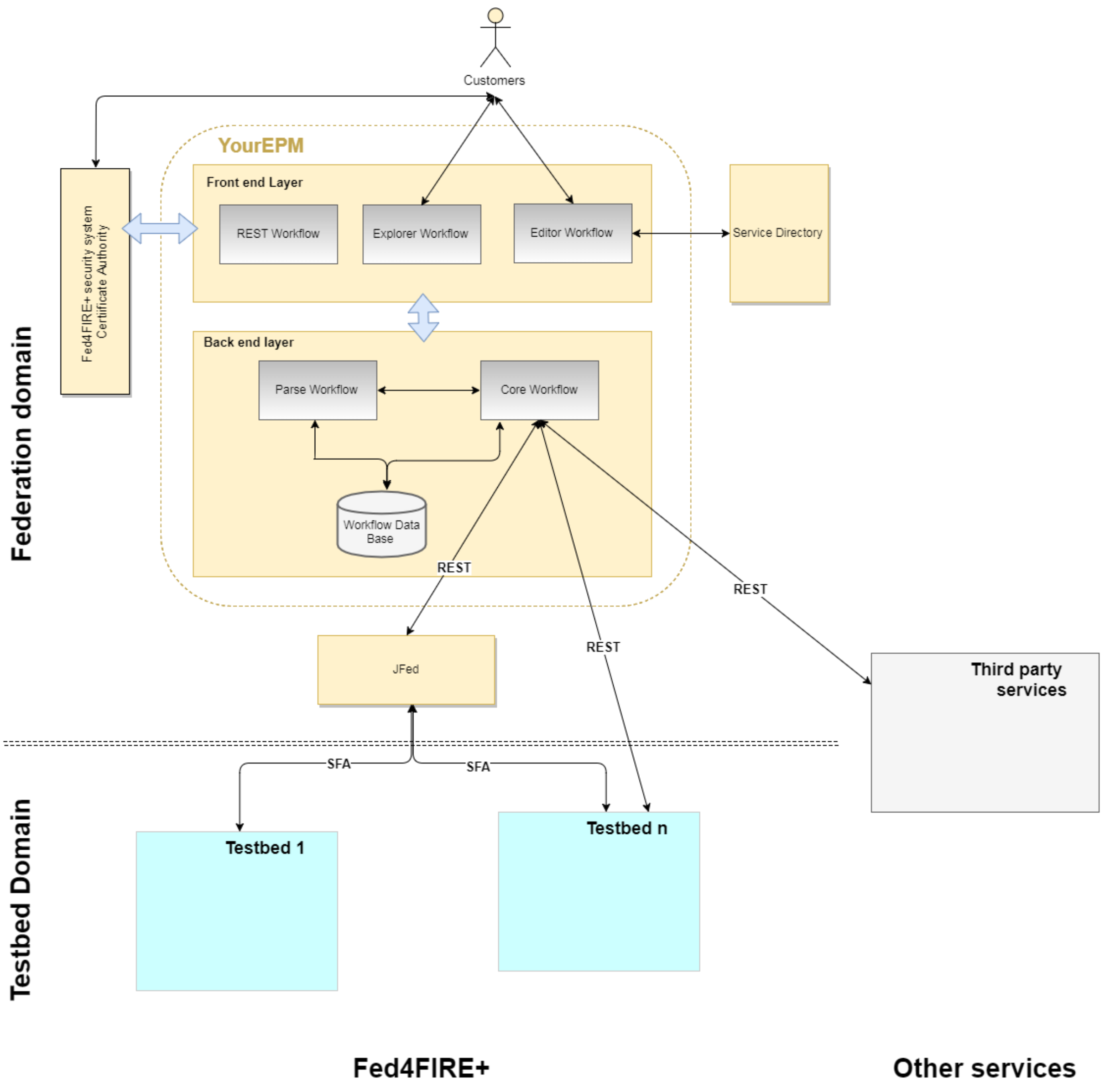


Figure 28: Architecture

The architecture of YourEPM, already presented in D3.1, is shown in Figure 28. It is composed of a Frontend layer (with a set of UIs and REST APIs) and a Backend layer, which contains business logic modules and the database.

Co-funded by the Horizon 2020 Framework Programme of the European Union

8.3 INSTALLATION MANUAL

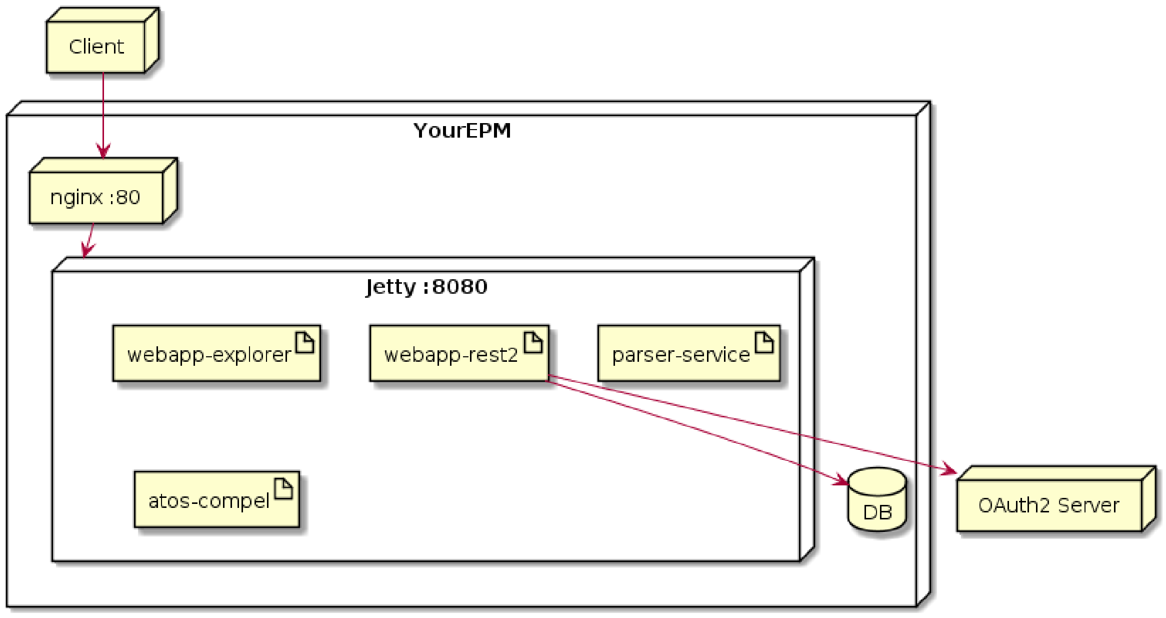


Figure 29: Deployment diagram

Figure 29 shows the deployment diagram of the YourEPM component. The component itself is composed of several WAR files served with a Jetty server. An nginx running in front of Jetty serves HTTP(S). YourEPM uses a MySQL or MariaDB Database server for storage. YourEPM delegates authentication and authorization to iMinds Authority server.

The SW prerequisites to install YourEPM are the following:

- Java 8+
- Mysql 5.5+ / MariaDB 10.3
- nginx

The following instructions have been tested on a Debian 10.1, but instructions on any distribution with systemd should be similar.

1. Build the distributable package (this can be done on development machine or on target server)

```
git clone https://gitlab.atosresearch.eu/ari/Fed4FIREPlus_YourEPM.git
cd workflow-engine
bin/bootstrap.sh
cd ../dist
bin/make-dist.sh
```

Co-funded by the Horizon 2020 Framework Programme of the European Union

The distributable package is now in dist/target. If built on development machine, zip that folder, upload to the server and unzip. From now on, DIST=<target of distributable package on server>

2. Configure $DIST/etc/env.sh
   The file has sensible defaults and you probably only need to set the OAuth password of the YourEPM application.

3. Configure database and users.

```
mysql -h localhost -u root -p < $DIST/share/database.sql
```

4. Create "yourepm" user to run the Workflow-Engine service.

```
adduser yourbpm
```

5. Configure the workflow-engine service in systemd. The service will run as the user just created.

```
ln -s $DIST/bin/run-workflow-engine /usr/local/bin/
ln -s $DIST/bin/stop-workflow-engine /usr/local/bin/
ln -s $DIST/etc/systemd/workflow-engine.service /etc/systemd/system
systemctl enable workflow-engine
systemctl start workflow-engine
```

6. Configure nginx

```
cd /etc/nginx/sites-available
ln -s $DIST/etc/nginx/workflow-engine .
cd /etc/nginx/sites-enabled
rm default
ln -s /etc/nginx/sites-available/workflow-engine .
```

7. Test installation at http://yourepm.fed4fire.eu/activiti-explorer.

8.3.1 Configuration file

The configuration file of YourEPM contains information about the ports where the application is exposed, the access to database and OAuth2 configuration. The default values shown below are suitable for production and development, but the passwords, which should be set to their actual values.

- START_PORT (default:8080). Port where Jetty listens.
- STOP_PORT (default:8081). Jetty listen on this port to initiate a shutdown when the STOP_KEY is received.
- STOP_KEY (default: sooooo).
- JDBC_URL (default: jdbc:mysql://localhost:3306/activiti_f4f). Database URL.
- JDBC_USERNAME (default: f4f). Username to access the database.
- JDBC_PASSWORD: Password to access the database.
- IDM_HOST (default: https://authority.ilabt.iminds.be). URL of OAuth2 server.

Co-funded by the Horizon 2020 Framework Programme of the European Union

- IDM_HOST_URL_LOGIN (default: /oauth_authorize.php). Path in OAuth2 server to redirect from YourEPM to initiate a login if user is not logged in.
- IDM_HOST_URL_TOKEN (default: /oauth_token.php). Path in OAuth2 server to get the OAuth2 token.
- IDM_APP_ID (default: atos_yourepm). ID of YourEPM in OAuth2 server.
- IDM_APP_PWD. Password of YourEPM in OAuth2 server.
- IDM_CALLBACK (default: http://yourepm.fed4fire.eu/activiti-explorer). Callback URL from OAuth2 server to YourEPM once user is authenticated.

8.4 FUTURE WORK

The following points have been considered for development during the third cycle:

- Make YourEPM accessible to users, so they can create workflows that interact with services exposed by the testbeds. These services must be added to the corresponding Service Directory.

Co-funded by the Horizon 2020
Framework Programme of the European Union

9 AUTHENTICATION PROXY SERVICE

The Fed4FIRE authentication proxy allows service providers, authenticated by Fed4FIRE+ credentials, to declare new HTTP or HTTPS endpoints (services or testbeds) to be made accessible to any authenticated Fed4FIRE+ user. It is a self-service service for testbed or service providers

The service or testbed provider must configure his endpoint so that the proxy can forward requests to it. This can be done by different means: by configuring his firewall at the endpoint or by uploading credentials to the proxy. In effect, this implies trusting the proxy to only transmit requests originating from authenticated Fed4FIRE+ users.

Indeed, Fed4FIRE+'s architecture is based on the SFA framework, where users identify themselves with certificates and authenticate themselves by using these certificates to secure an TLS link to service providers. While this architecture has a lot of good properties, it is not a good fit to consume services implemented using a REST or SOAP architecture, very common in today's Internet. Inria's work for this component has consisted in the implementation of a gateway, the Fed4FIRE authentication proxy, to bridge the gap between Fed4FIRE+ credentials and services.

It is available at https://f4fauthproxy.fed4fire.eu/, complete with user documentation.

9.1 REQUIREMENTS COVERAGE

The requirements defined for the component are presented in the table below, which provides an explanation on the maturity achieved during the second cycle and the next steps for the third cycle. The requirements highlighted in green can be considered complete.

ID	Title	Coverage in Second cycle
1	Expose HTTP servers (exposing APIs or websites) only to authenticated Fed4FIRE+ users through a self-service authentication proxy	implemented
2	Allow HTTP server owners to control who can access their server through the authentication proxy	implemented
3	Transmit information extracted from the user certificate (urn, authority urn and email) to HTTP server accessed through the authentication proxy through HTTP headers	implemented

Co-funded by the Horizon 2020 Framework Programme of the European Union

4	Allow HTTP server owner to share control of the configuration of their service as declared in the authentication proxy to members of a Fed4FIRE group	implemented
5	Allow HTTP server owners to provide a login URL to be used by the authentication proxy if a login process is required to access parts of their server	On-going
6	Allow HTTP server owners to secure the link between the authentication proxy and their server using a login and a password	implemented
7	Allow HTTP server owners to secure the link between the authentication proxy and their server using a TLS client certificate	implemented
8	Allow HTTP server owners to secure the link between the authentication proxy and their server by declaring the authority of their server's certificate	implemented
9	Allow declaration of a new HTTP server accessible through the authentication proxy through an API	implemented
10	Allow complete configuration of the proxy service to an HTTP server accessible through the authentication proxy through an API	implemented

9.2 ONLINE DOCUMENTATION

All the documentation included hereafter is available online at https://f4fauthproxy.fed4fire.eu/.

Co-funded by the Horizon 2020
Framework Programme of the European Union

9.3 INSTALLATION MANUAL

9.3.1 Overview

- You need to configure apt sources to be able to download and install the software
- You will need to setup a database to be used by the service
- You will need to install Fed4FIRE's trusted root authorities
- You will need to install a client credential for f4f-auth-proxy service so it can interact with other services in the federation
- You will need to configure a key so data at rest in the database is stored encrypted
- You will need to setup an apache frontend, complete with its TLS certificates
- You will need to start the service, and check it works

9.3.2 Debian specific instructions

9.3.2.1 Installation of the f4f-auth-proxy package

```
echo "deb http://packages.grid5000.fr/deb/f4f-auth-proxy/stretch /" | sudo
tee /etc/apt/sources.list.d/f4f-auth-proxy.list
sudo apt-get install gnupg dirmngr # probably already configured on your
machine
sudo apt-key adv --keyserver hkp://pool.sks-keyservers.net --recv-keys
B1F34F56797BF2D1
#sudo apt-key adv --keyserver hkp://134.93.178.170 --recv-keys
B1F34F56797BF2D1
sudo apt update
sudo apt install f4f-auth-proxy
```

The last command should have output a message telling you no database configured for the package was found.

9.3.2.2 Installation of a local database for f4f-auth-proxy

For the impatient, here are quick instructions to setup a local database on the machine hosting the service.

```
sudo apt-get install mariadb-server
/usr/bin/f4f-auth-proxy rake db:create # to print the expected database
parameters (you can also read/adapt /opt/f4f-auth-
proxy/config/database.yml)
sudo mysql -e "
CREATE USER auth_proxy IDENTIFIED BY '6bd17588552116e';
CREATE DATABASE f4f_auth_proxy_production CHARACTER SET utf8 COLLATE
utf8_unicode_ci ;
GRANT ALL privileges on f4f_auth_proxy_production.* to auth_proxy@'%'
identified by '6bd17588552116e' ;
GRANT ALL privileges on f4f_auth_proxy_production.* to auth_proxy@localhost
identified by '6bd17588552116e' ;"
sudo mysql_secure_installation
sudo /usr/bin/f4f-auth-proxy rake db:migrate
```

Co-funded by the Horizon 2020
Framework Programme of the European Union

Warning: There is no point in saving that database without saving the content of the encryption key, usually found in `/etc/f4f-auth-proxy/conf.d/value-encryption-key.sh` (see paragraph below).

9.3.3 Configuration of the service

9.3.3.1 Install Fed4FIRE's trusted roots

The configuration below expects all trusted roots to be concatenated in the `/etc/f4f-auth-proxy/certs/trusted_roots.pem` file. This file is expected by the `SSLCACertificateFile` for the apache frontend and by the `fed4fire_cacerts` configuration option of the file linked through `/etc/f4f-auth-proxy/fed4fire.yml`. The service makes no assumptions on how you keep that file up to date with the official list of trusted root for the federation.

9.3.3.2 Create a client account so the service can interact with other services

Admin and usage rights to proxies created using f4f-auth-proxy can be limited to members of the groups a user belongs to, as defined by the slice authority. For this functionality to work, f4f-auth-proxy needs access to a user certificate corresponding to a user with privileges.

For the iminds authority, you need to create a user in the `userinfo` group. This user will be used by `f4f-auth-proxy` when it interacts with SFA APIs as a client. It uses this capacity to query the list of groups a user belongs to, using the `lookup_for_member` call defined for Project Member Service Methods in the Federation Service APIs for slice authorities.

The client certificate will need stored at a location defined by the `service_cert` entry of the `/opt/f4f-auth-proxy/config/fed4fire.yml` configuration file (`/etc/f4f-auth-proxy/certs/sfa_client_cert.pem` by default). The password is expected to be found in the environment by `f4f-auth-proxy`, as set by the startup configuration script fragment in `/etc/f4f-auth-proxy/conf.d/f4f-auth-proxy-sfa-client-pass.sh`.

```
# check we have a proper client cert
cat /etc/f4f-auth-proxy/certs/sfa_client_cert.pem
# check we have saved the client cert's password
cat /etc/f4f-auth-proxy/conf.d/f4f-auth-proxy-sfa-client-pass.sh
# check permissions
ls -al /etc/f4f-auth-proxy/conf.d/f4f-auth-proxy-sfa-client-pass.sh
```

9.3.3.3 Setup encryption key

The Fed4FIRE authentication proxy service will very probably be storing some secrets to it can connect to the services it proxies. As these are precious, they are stored encrypted in the database. The encryption key used by rails is found by the application in the F4F_VALUE_ENCRYPTION_KEY environment variable, which can populated when the application is started with the `f4f-auth-proxy` using a configuration file in the `/etc/f4f-auth-proxy/conf.d` directory. The debian package generates a `value-encryption-key.sh` file for this purpose.

Co-funded by the Horizon 2020 Framework Programme of the European Union

The contents of that file are precious. All the tesbed specific data stored in the application's database is useless if the value is lost. Should you want to change the value generate at install time, we strongly recommend you save the old value and date of change so as to be able to migrate all values previously stored in the database.

9.3.3.4 Setup apache frontend for user authentication

The f4f-auth-proxy application expects to receive the certificate used to secure the TLS connection between the client and the service as an http header. Extracting that certificate from the network layer so it can used as a root of trust in the application layer is a task delegated to apache in the reference implementation.

Apache must be configured for TLS, with a valid certificate (in the following example managed by letsencrypt, but if you are just testing the software, sample certs are available at `/etc/f4f-auth-proxy/certs/`), and with the proper options to check the client cert if one is used. The Federation's trusted roots should be available as `/etc/f4f-auth-proxy/certs/trusted_roots.pem` in this example. Here is a sample `/etc/apache2/sites-available/f4f-auth-proxy.conf`.

```
<VirtualHost *:443>
    ServerName localhost

    DocumentRoot /opt/f4f-auth-proxy/public/
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
  LogLevel info ssl:info

  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
    SSLEngine on

    #   A self-signed (snakeoil) certificate can be created by installing
    #   the ssl-cert package. See
    #   /usr/share/doc/apache2/README.Debian.gz for more info.
    #   If both key and certificate are stored in the same file, only the
    #   SSLCertificateFile directive is needed.
    #SSLCertificateFile    "/etc/letsencrypt/live/proxy.fed4fire.eu/fullchain.pem"
    #SSLCertificateKeyFile "/etc/letsencrypt/live/proxy.fed4fire.eu/privkey.pem"
    SSLCertificateFile    "/etc/f4f-auth-proxy/certs/server.pem"
    SSLCertificateKeyFile "/etc/f4f-auth-proxy/certs/server.key"

    #   Server Certificate Chain:
    #   Point SSLCertificateChainFile at a file containing the
    #   concatenation of PEM encoded CA certificates which form the
    #   certificate chain for the server certificate. Alternatively
    #   the referenced file can be the same as SSLCertificateFile
    #   when the CA certificates are directly appended to the server
    #   certificate for convinience.
    #SSLCertificateChainFile "/etc/letsencrypt/live/proxy.fed4fire.eu/fullchain.pem"
    SSLCertificateChainFile "/etc/f4f-auth-proxy/certs/ca/org.valid.pem"

    #   Certificate Authority (CA):
    #   Set the CA certificate verification path where to find CA
    #   certificates for client authentication or alternatively one
    #   huge file containing all of them (file must be PEM encoded)
    #   Note: Inside SSLCACertificatePath you need hash symlinks
    #                to point to the certificate files. Use the provided
    #                Makefile to update the hash symlinks after changes.
    #SSLCACertificatePath /etc/ssl/certs/
```

Co-funded by the Horizon 2020 Framework Programme of the European Union

```
  SSLCACertificateFile "/etc/f4f-auth-proxy/certs/trusted_roots.pem"

  #   Client Authentication (Type):
  #   Client certificate verification type and depth.  Types are
  #   none, optional, require and optional_no_ca.  Depth is a
  #   number which specifies how deeply to verify the certificate
  #   issuer chain before deciding the certificate is not valid.
# SSLVerifyClient set Location by Location
  # SSLVerifyClient optional
  SSLVerifyDepth  10

  #   SSL Engine Options:
  #   Set various options for the SSL engine.
  #   o FakeBasicAuth:
  #        Translate the client X.509 into a Basic Authorisation.  This means that
  #        the standard Auth/DBMAuth methods can be used for access control.  The
  #        user name is the `one line' version of the client's X.509 certificate.
  #        Note that no password is obtained from the user. Every entry in the user
  #        file needs this password: `xxj31ZMTZzkVA'.
  #   o ExportCertData:
  #        This exports two additional environment variables: SSL_CLIENT_CERT and
  #        SSL_SERVER_CERT. These contain the PEM-encoded certificates of the
  #        server (always existing) and the client (only existing when client
  #        authentication is used). This can be used to import the certificates
  #        into CGI scripts.
  #   o StdEnvVars:
  #        This exports the standard SSL/TLS related `SSL_*' environment variables.
  #        Per default this exportation is switched off for performance reasons,
  #        because the extraction step is an expensive operation and is usually
  #        useless for serving static content. So one usually enables the
  #        exportation for CGI and SSI requests only.
  #   o OptRenegotiate:
  #        This enables optimized SSL connection renegotiation handling when SSL
  #        directives are used in per-directory context.
  #SSLOptions +FakeBasicAuth +ExportCertData +StrictRequire
  #<FilesMatch "\.(cgi|shtml|phtml|php)$">
  #   SSLOptions +StdEnvVars
  #</FilesMatch>
  SSLOptions +ExportCertData +StrictRequire

  #   SSL Protocol Adjustments:
  #   The safe and default but still SSL/TLS standard compliant shutdown
  #   approach is that mod_ssl sends the close notify alert but doesn't wait for
  #   the close notify alert from client. When you need a different shutdown
  #   approach you can use one of the following variables:
  #   o ssl-unclean-shutdown:
  #        This forces an unclean shutdown when the connection is closed, i.e. no
  #        SSL close notify alert is send or allowed to received.  This violates
  #        the SSL/TLS standard but is needed for some brain-dead browsers. Use
  #        this when you receive I/O errors because of the standard approach where
  #        mod_ssl sends the close notify alert.
  #   o ssl-accurate-shutdown:
  #        This forces an accurate shutdown when the connection is closed, i.e. a
  #        SSL close notify alert is send and mod_ssl waits for the close notify
  #        alert of the client. This is 100% SSL/TLS standard compliant, but in
  #        practice often causes hanging connections with brain-dead browsers. Use
  #        this only for browsers where you know that their SSL implementation
  #        works correctly.
  #   Notice: Most problems of broken clients are also related to the HTTP
  #   keep-alive facility, so you usually additionally want to disable
  #   keep-alive for those clients, too. Use variable "nokeepalive" for this.
  #   Similarly, one has to force some clients to use HTTP/1.0 to workaround
  #   their broken HTTP/1.1 implementation. Use variables "downgrade-1.0" and
  #   "force-response-1.0" for this.
  BrowserMatch "MSIE [2-6]" \
      nokeepalive ssl-unclean-shutdown \
      downgrade-1.0 force-response-1.0
  # MSIE 7 and newer should be able to use keepalive
  BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown
```

Co-funded by the Horizon 2020
Framework Programme of the European Union

```
  <Location "/">
    # Client verification optional so the app can display errors
    # messages with instructions
    SSLVerifyClient optional
      RequestHeader set X-Fed4fire-Client-Cert %{SSL_CLIENT_CERT}s
    ProxyPass "http://127.0.0.1:8000/"
    ProxyPassReverse "http://127.0.0.1:8000/"
    RequestHeader set X-Forwarded-Proto https
  </Location>
  # Allow users to browse doc and read the landing page
    # without being requested a client cert
  <Location ~ "/(install|documentation|user|$|favicon.ico$)">
    SSLVerifyClient none
  </Location>
  <Location "/assets/">
    Require all granted
    # Use of ETag is discouraged when Last-Modified is present
    Header unset ETag
    FileETag None
    # RFC says only cache for 1 year
    ExpiresActive On
    ExpiresDefault "access plus 1 year"
    ProxyPass !
  </Location>

</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

You then need to enable the required modules and then the site:

```
sudo a2enmod ssl headers proxy proxy_http expires
sudo a2ensite f4f-auth-proxy.conf
```

9.3.4 Starting and checking the service

```
sudo systemctl start apache2 f4f-auth-proxy
sudo systemctl status apache2 f4f-auth-proxy
```

You should now be able to access the service’s homepage at https://your.location/ If you don’t provide a client cert, you will only have access to the documentation (an in app version of the API documentation, and if you do, you should be able to create a proxy for an endpoint, and then use it.

9.3.5 Debugging tools

```
sudo systemctl status apache2 f4f-auth-proxy
sudo less -R /var/log/f4f-auth-proxy/production.log
sudo less -R /var/log/apache2/error.log
```

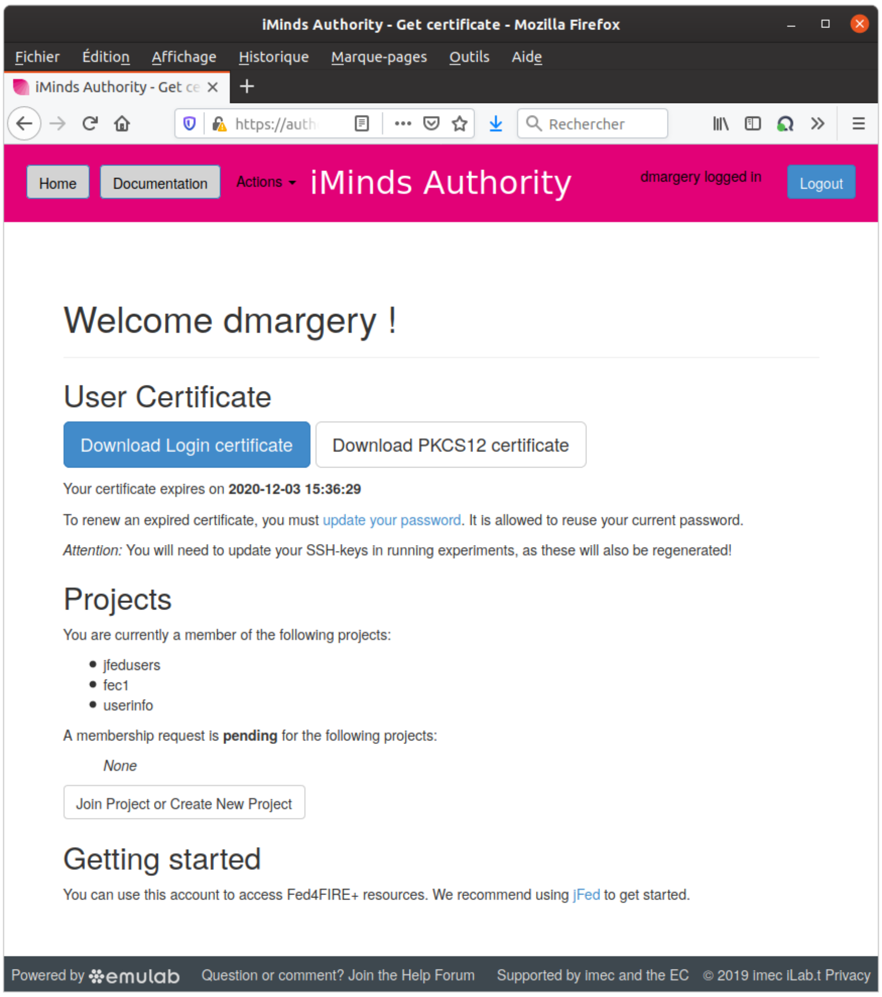
Co-funded by the Horizon 2020
Framework Programme of the European Union

9.3.6 End-user manual

9.3.6.1 Authentication for all users

All user authentication for f4f-auth-proxy is based on certificates at TLS level. You need to be able to prove to the service you are the legitimate owner of a certificate signed by a Fed4FIRE+ member authority.

To access the web-based interface, you will therefore be required to upload such a certificate to your browser. For user of the iMinds authority, you can do so by uploading the PKCS12 cert available from the https://authority.ilabt.iminds.be/getcert.php. Your browser should try to handle the downloaded certificate itself, and ask whether it should import it.


iMinds Authority - Get certificate - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
iMinds Authority - Get ce
https://auth
Rechercher
Home
Documentation
Actions
iMinds Authority
dmargery logged in
Logout
Download Login certificate
Download PKCS12 certificate
Join Project or Create New Project
Powered by emulab
Question or comment? Join the Help Forum
Supported by imec and the EC
© 2019 imec iLab.t Privacy

Welcome dmargery !

User Certificate

Your certificate expires on **2020-12-03 15:36:29**

To renew an expired certificate, you must update your password. It is allowed to reuse your current password.

Attention: You will need to update your SSH-keys in running experiments, as these will also be regenerated!

Projects

You are currently a member of the following projects:

- jfedusers
- fec1
- userinfo

A membership request is **pending** for the following projects:

None

Getting started

You can use this account to access Fed4FIRE+ resources. We recommend using jFed to get started.

Figure 30: screenshot of a page allowing Fed4FIRE users to get a client certificate

If you are accessing f4f-auth-proxy's API, or the API of a service or testbed through a proxy entry-point, you might need the certificate in a PEM format. This is the 'login certificate' also available from the same page for users registered with the iMinds authority.

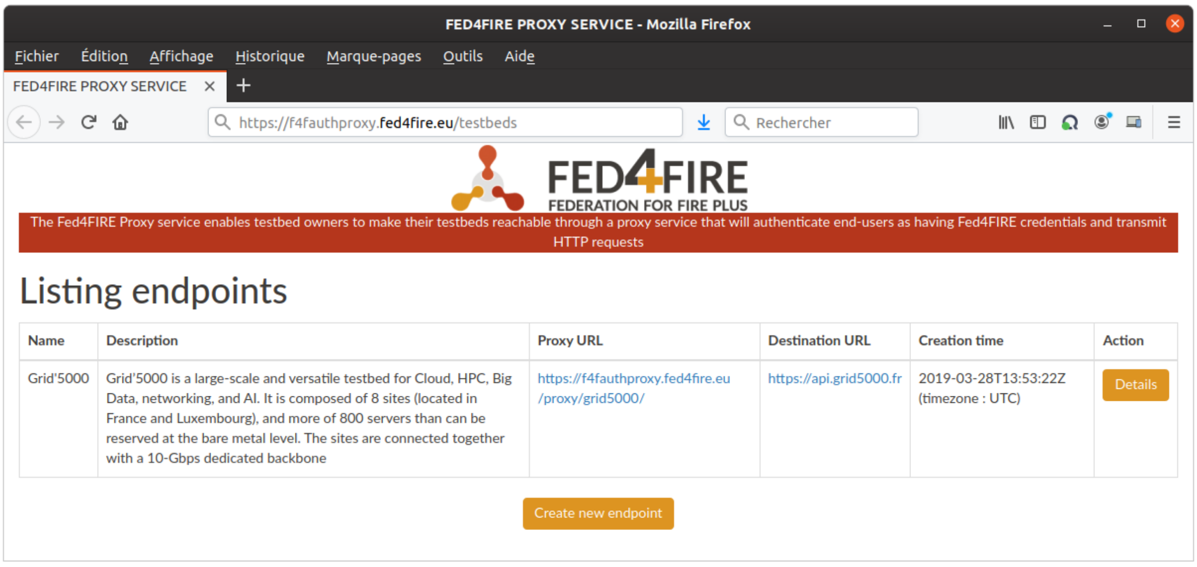
Co-funded by the Horizon 2020 Framework Programme of the European Union

9.4 AS A PROXY USER

For each registered testbed or endpoint, the f4f-auth-proxy acts as a proxy when queried using https://f4fauthproxy.fed4fire.eu/proxy/<testbed or end point name>. That url is derived from the short name registered by the testbed or endpoint owner, and described as the `proxy url` of each testbed on the page listing configured proxies.

When that url points to a service usable through the web browser, clicking on the link in the `proxy url` should bring up the testbed's web page. If it is an API, sending GET, POST, PUT, DELETE requests to the proxy url will forward these requests to the final endpoint.

For debugging purposes, the Via header is completed when answers come back through the proxy, allowing the user to differentiate errors at the proxy level and error at the endpoint.


FED4FIRE PROXY SERVICE - Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
FED4FIRE PROXY SERVICE
https://f4fauthproxy.fed4fire.eu/testbeds
Rechercher
FED4FIRE
FEDERATION FOR FIRE PLUS

The Fed4FIRE Proxy service enables testbed owners to make their testbeds reachable through a proxy service that will authenticate end-users as having Fed4FIRE credentials and transmit HTTP requests

Listing endpoints

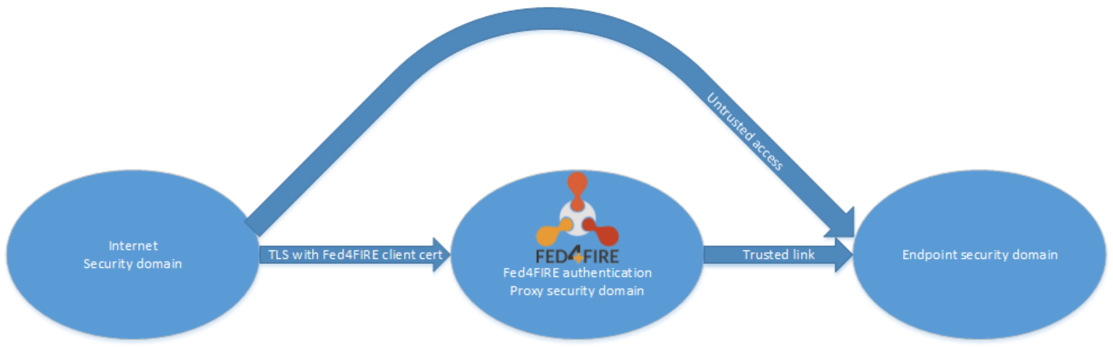
Name	Description	Proxy URL	Destination URL	Creation time	Action
Grid'5000	Grid'5000 is a large-scale and versatile testbed for Cloud, HPC, Big Data, networking, and AI. It is composed of 8 sites (located in France and Luxembourg), and more of 800 servers than can be reserved at the bare metal level. The sites are connected together with a 10-Gbps dedicated backbone	https://f4fauthproxy.fed4fire.eu/proxy/grid5000/	https://api.grid5000.fr	2019-03-28T13:53:22Z (timezone : UTC)	Details

Figure 31 : screenshot of the list of proxies, with the url to use each

9.4.1 Endpoint or testbed user manual

The core principle for an endpoint or testbed owner adding a proxy using f4f-auth-proxy's self-service portal is to secure the link between the service and his testbed or endpoint, as illustrated in Figure 45.

Co-funded by the Horizon 2020 Framework Programme of the European Union



To achieve this, the f4f-auth-proxy service allows owners of testbeds or endpoints to run a security setup wizard that presents them with a web interface enabling them to choose one or more of the following options:

- HTTP basic auth over https (username/password). Here, the HTTP server at the endpoint/testbed is configured to allow access for a given user identified by a password. This is only secure if the link to the endpoint/testbed is established using TLS. To enable this, you need
    - to configure the Fed4FIRE proxy with the username and password to use to access the endpoint. These will only be made visible to the people who can manage the endpoint/testbed in the application.

Figure 32:Securing the link between the service and an endpoint

    - Ensure the Fed4FIRE authentication proxy will trust the endpoint/testbed certificate presented to it while establishing the TLS link.
- TLS client auth. The TLS protocol allows a client to identify itself using a certificate while securing the TLS link. This requires the endpoint/testbed to be configured to check the client certificate. To enable this, you need to upload the certificate and private key to be used by the proxy.
- Firewall; It is also possible to restrict access to the endpoint/testbed by setting up a firewall or an access list at the endpoint security domain entry point. For this, you only need to know that the Fed4FIRE+ authentication proxy connects from the IP addresses shown by the interface and to setup your firewall or access list accordingly.
- Setup a connection proxy. This should be possible using the connections options that can be configured for each testbed, but is not supported by the security wizard. Please use the general configuration options of each testbed to configure them.

The general configuration page for each endpoint or testbed declared also gives control to owners to

- Specify users that are allowed to use the endpoint through the proxy (a specific user, all user in the same group as declared in a specific Slice Authority the service has access to, all Fed4FIRE+ users).
- Specify users that are allowed to change the configuration of the proxy service to the testbed or endpoint.

Co-funded by the Horizon 2020
Framework Programme of the European Union

- Specify headers that are forwarded to the endpoint. These headers come in 2 flavours : generic http headers and headers whose value is extracted from the client’s certificate (urn, email, authority having signed the cert, and the complete certificate)

10 CENTRAL BROKER

The Central Broker acts as an overarching service that can be utilized by the experimenters to discover resources that span the federation and fulfil their experimentation requirements. More specifically, the Central Broker receives as input general specifications from the users related to date and time they want to run their experiment, technologies involved like Wired, Wireless, 5G, IoT, OpenFlow, Big Data, hardware specifications of the nodes like number of CPUs, amount of RAM and storage and finally number of nodes that denote the scale of the experiment willing to conduct in Fed4FIRE. Upon receiving these requests, the Central Broker filters the available resources of the federation and maps the experimenters' requests to the actual testbed resources, providing as output a proposed set of resources to the experimenter, who in his turn decides whether to request/reserve these resources for his experiment.

To this end, the Central Broker needs to keep an up-to-date inventory of the available resources of all the testbeds that are part of the federation. The updated inventory allows the service to apply its mapping algorithm that take into consideration the user's requirements along with the characteristics of the available resources and provide the results without any significant delays.

10.1 REQUIREMENTS COVERAGE

The requirements defined for the Central Broker are presented in the table below, which provides an explanation on the maturity achieved during the second cycle and the next steps for the third cycle.

ID	Title	Coverage in Second cycle
1	Retrieve resource availability of all the testbeds of the federation.	It was fully covered by the usage of FedMon service which monitors all the testbeds and stores information regarding their available resources.
2	Keep an up-to-date inventory of the available resources of all the testbeds that are part of the federation.	It was fully covered by the extension of the Central Broker database schema and information model.
3	Provide a method that enables experimenters to express their resource requirements in a future proof extensible way.	It was fully covered by the utilization of the REST API and the JSON format that can be extended on will for future extensions.
4	Support all the different testbed technologies like Wireless, Wired, 5G, Cloud, IoT, Big Data and OpenFlow in the resource mapping algorithm.	It was fully covered by the refactoring of mapping algorithm and the provision of an updated API. For future updates on testbed resources

Co-funded by the Horizon 2020 Framework Programme of the European Union

		(upgrades/new additions) the same process will be followed.
5	Respond with a set of available resources that meet the experimenter's requirements.	It was partially covered by the continuous resource availability update every 20 minutes. The accuracy of Central Broker will be monitored and corrective actions will be made in the Third cycle if necessary.

10.2 ARCHITECTURE

In the following diagram the overall architecture can be seen. From the user's perspective the workflow includes a query to fed4fire.eu portal regarding date and time they want to run their experiment, technologies involved and number of nodes. These requirements are mapped by the Central Broker to a recommended set of resources that are presented to the experimenter. In the next step, the experimenter will be able to launch jFed through fed4fire.eu portal and request the recommended resources before starting the execution of his experiment.

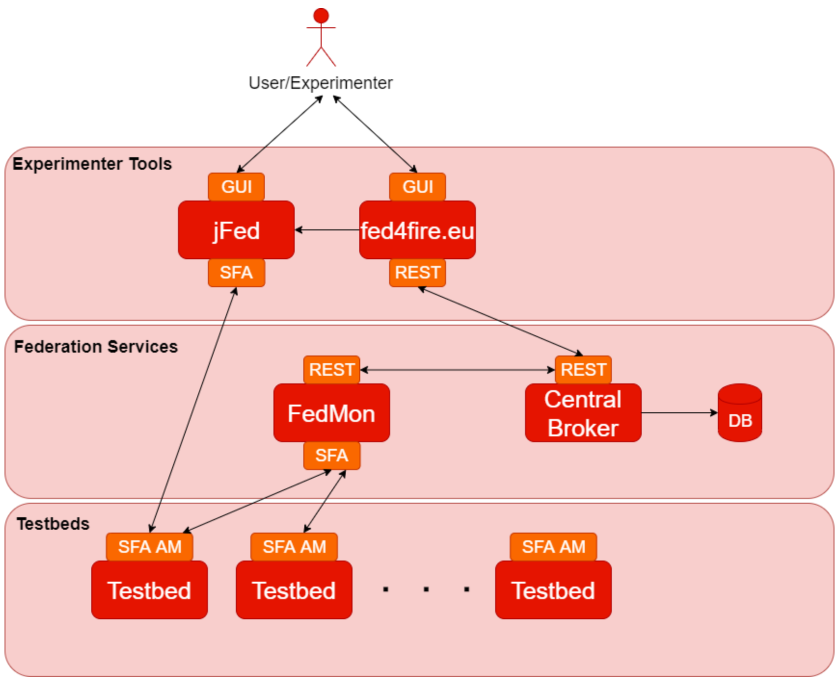


Figure 33: Architecture

Co-funded by the Horizon 2020 Framework Programme of the European Union

Behind the scenes, the following interactions take place. The federation monitoring service FedMon queries the Federation testbeds every 20 minutes, collecting information including available resources. The Central Broker queries the REST API of FedMon service in order to retrieve that information and keep an updated inventory in its internal database. Every time a user is providing its requirements through fed4fire.eu a REST call is made towards the REST interface of the Central Broker, which after mapping the requirements of the experimenter it responds with a set of available resources that match his requirements.

10.3 SEQUENCE DIAGRAMS

A more detailed representation of the interactions between the several architectural components can be seen in the sequence diagram below. There is a block for every actor which is involved in the process of discovering the required resources, requesting and executing the experiment. More specifically, the sequence diagram includes the following actors: i)Experimenter, ii)fed4fire.eu portal, iii)jFed iv)Central Broker, v)FedMon and vi)Testbeds. Every 20 minutes there is an action from FedMon which monitors the federation testbeds by executing several test routines, including the resource availability of each testbed. The Central Broker keeps its inventory updated by querying the FedMon service through its REST API and collecting the information regarding resource availability per testbed.

Every time an experimenter is using the fed4fire.eu portal to discover resources that match his requirements, a REST call is made to the Central Broker which includes the experimenter's resource requirements like date, time, volume of resources, type of technology and hardware capabilities. The Central Broker runs a resource mapping algorithm, which provides as a response a set of resources that match the experimenter's requirements.

Co-funded by the Horizon 2020
Framework Programme of the European Union

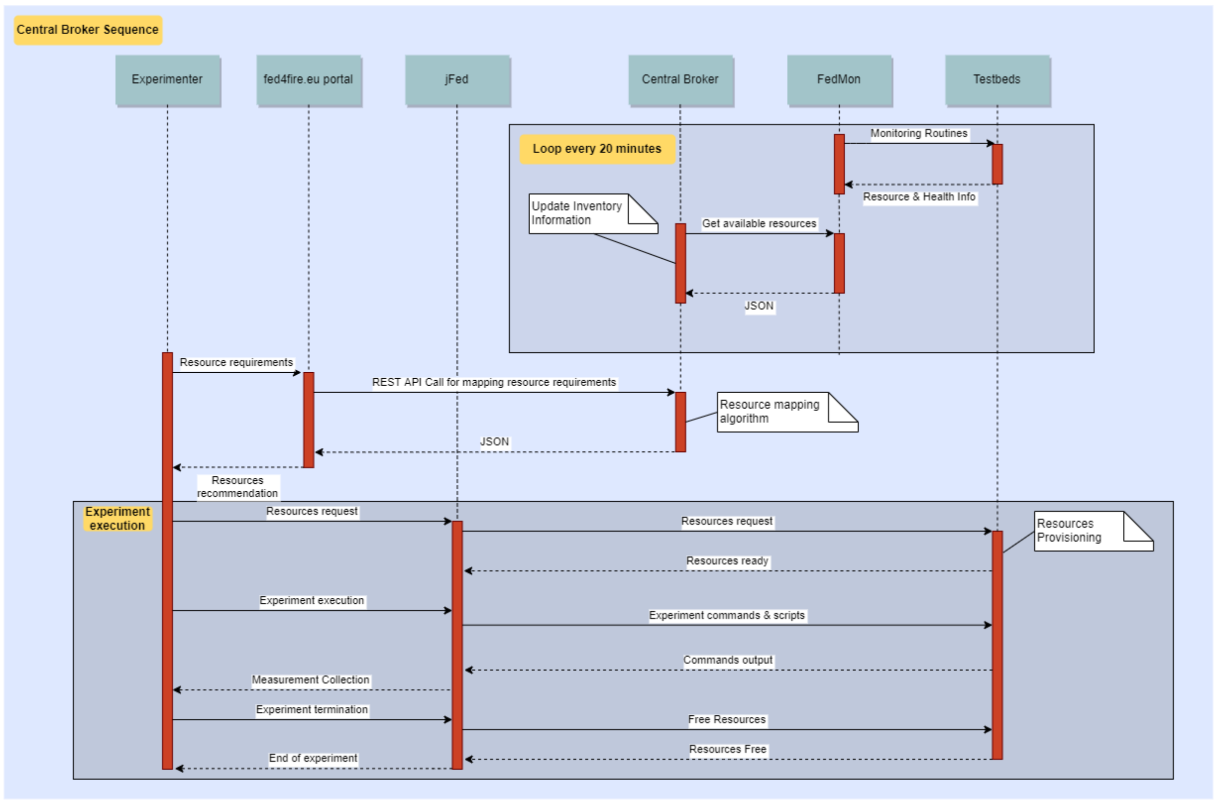


Figure 34: Sequence Diagram

Finally, the experimenter can launch jFed through fed4fire.eu portal in order to request the recommended resources by the testbeds and proceed to the experiment execution. During the execution, the resources will be provisioned by the testbeds, the experimenters will execute their scripts on the resources and the experiment measurements will be collected before the resources are released.

10.4 USER MANUAL

The REST endpoint of the Central Broker offers a single operation which is about mapping experimenters' resource requirements to a specific set of resources. The REST API call conveys the experimenters' requirements in JSON format and responds with a JSON that describes the recommended resources.

POST /API/mapper

This operation is used to perform a mapping task at the Central Broker.

Description:

Co-funded by the Horizon 2020 Framework Programme of the European Union

- It receives a JSON describing resource requirements like date, time, number and type of resources and hardware capabilities. It responds with a JSON listing all the valid resources that match the initial requirements.

REQUEST

```
POST API/mapper
Content-type: application/json

{
  JSON Resource Requirements
}
```

RESPONSE

```
201
Content-Type: application/json

{
  JSON Resource Representation
}
```

The Central Broker accepts the following JSON fields that can describe user's requirements regarding resources. In the following example an experimenter requests two resources with WiFi interfaces during Monday 2nd of December 09:00 until 17:00. There are also some minimum requirements regarding number of CPUs (2), RAM (4 GiB) and storage (60 GiB).

```
{
    "resources":  [
        {
            "exclusive": true,
            "hardware_type": “wifi”,
            "valid_from": “Mon Dec 2 09:00:00 EET 2019”,
            "valid_until": “Mon Dec 2 17:00:00 EET 2019”,
            "cpu": 2,
            "ram": 4,
            "storage": 60
        },
        {
            "exclusive": true,
            "hardware_type": “wifi”,
            "valid_from": “Mon Dec 2 09:00:00 EET 2019”,
            "valid_until": “Mon Dec 2 17:00:00 EET 2019”,
            "cpu": 2,
            "ram": 4,
            "storage": 60
        }
    ]
}
```

In the following table there is a complete list of the fields together with their description than can be present in the JSON body of a resource mapping request.

Co-funded by the Horizon 2020
Framework Programme of the European Union

Field	Description
exclusive	Valid values can be “true” or “false”, which denotes whether the experimenter has exclusive bare metal access to the resource, or he is sharing the resource (Virtual Machine).
hardware_type	Can be used in order to require a specific technology like WiFi, 5G, Cloud, Big Data, IoT, OpenFlow or Wired. Values of this field continuously change as testbeds add new resources and technologies. Current valid values include the following keywords: “wifi”, “5g”, “iot”, “bigdata”, “openflow” and “wired”.
valid_from	The experimenter can define a date and time that he wants his resources to be available.
valid_until	Accordingly, the experimenter can define the date and time that he is willing to end his experiment and free the requested resources.
cpu	Number of minimum CPUs required. Valid values: [2-16]
ram	Number of minimum RAM (GiB) required. Valid values: [2-48]
storage	Number of minimum storage (GiB) required. Valid values: [4-4000]

Based on the JSON request given as an example above, the following JSON shows a valid response from the Central Broker that meets the requirements set by the experimenter. It lists two resources with WiFi capabilities from the wireless testbed w-ilab.t that are available for reservation during the specified time window.

```
{
   "resources_response": {
      “resources”: [
         {
            "exclusive": true,
            "name": “nuc9-29”,
            "valid_from": “Mon Dec 2 09:00:00 EET 2019”,
            "valid_until": “Mon Dec 2 17:00:00 EET 2019”,
            "domain": “wilab1.ilabt.iminds.be”,
         },
         {
            "exclusive": true,
            "name": “nuc9-30”,
            "valid_from": “Mon Dec 2 09:00:00 EET 2019”,
            "valid_until": “Mon Dec 2 17:00:00 EET 2019”,
            "domain": “wilab1.ilabt.iminds.be”,
         }
      ]
   }
}
```

The following table describes the fields that can be found in a JSON response from the Central Broker.

Co-funded by the Horizon 2020
Framework Programme of the European Union

Field	Description
exclusive	Valid values can be “true” or “false”, which denotes whether the experimenter has exclusive bare metal access to the resource, or he is sharing the resource (Virtual Machine).
name	The name of the resource that meets the requirements.
valid_from	Date and time that the resource is available.
valid_until	Date and time that the resource will be released.
domain	Unique identifier of the testbed, which the resource belongs to.

10.5 FUTURE WORK

The federation service of Central Broker will need continuous maintenance and updates in order to keep following every resource change that happens in the testbeds. While testbeds add new resources or upgrade existing ones, the inventory of Central Broker must be extended in order to accommodate these changes. Moreover, the resource mapping algorithm needs to be refactored in order to reflect the new hardware capabilities available by the newly added resources. The REST API of the Central Broker will have to be updated as well in order to be able to express the new capabilities provided.

Apart from continuous updates and maintenance, the Central Broker will be assessed in order to verify whether the 20 minutes interval window is sufficient or not to provide accurate resource information to the experimenters. There will be two approaches for determining the successful and failed resource requests that were recommended by the Central Broker. First a feedback button will be created in fed4fire.eu portal when users submit their resource preferences and receive the corresponding recommendations by the Central Broker. The experimenter will be able to give immediate feedback on whether the recommendations were available or not and their sufficiency with regards to the provided requirements. A second approach will be made by checking the availability of resources every time a request is made to the Central Broker. This will be made possible by querying the corresponding testbeds regarding the availability of resources at the time a resource recommendation is generated by the mapping algorithm of the Central Broker.

The results of the Central Broker assessment on valid resource recommendations will allow for any necessary corrective actions. If the percentage of unavailable resources is high for every resource recommendation, a possible solution will be to offer alternative recommendations to pick from. Moreover, if there is a tendency on invalid availability of resources on specific testbeds due to high number of experimenters’ requests for resources, solutions like decreasing the polling interval window per testbed will be examined.

Co-funded by the Horizon 2020
Framework Programme of the European Union

11 ONTOLOGIES

11.1 SEMANTIC BASED RESOURCE DESCRIPTION

In a federated environment, such as the Fed4FIRE+ project, an in-depth description of the testbed facilities and their resources would support users in all phases of an experiment, like resource discovery, reservation and construction of an execution scenario. Given the heterogeneity of the various nodes employed in the case of Fed4FIRE+, one particular issue that emerges is the description of these offerings. SFA, the de facto standard API for testbed federation, uses XML-based Resource Specifications (RSpecs) with arbitrary extensions to describe, discover, provision and release resources. However, such tree-based data models, lack consistency, standardized vocabularies as well as semantic meanings, therefore impede interoperability within a federation [WiPa15] [MoWi16]. In the context of the WP3 Task 3.5, re-usage and extension of already well-defined standard semantic models are adopted for representing and linking Fed4FIRE+ federated resources. Additionally, the usage of a semantic registry repository for testbeds and resources will enable experimenters to find and book resources more easily. For this purpose, the OMN ontology suite [WiPa15], [MoWi16] is adopted and extended; the OMN ontology is being developed within the W3C Federated Infrastructures Community Group 4. The use of OMN leads to:

- Introduction of the necessary extensions and adoption of existing ontologies relative to the Fed4FIRE+ experimentation environment (Wireless Nodes, Interfaces, etc.),
- Maintenance of compatibility and interoperability with existing SFA-enabled infrastructures, by using the information model and the corresponding data models with the Aggregate Manager.

The OMN Wireless ontology, mainly used in the semantic description of the testbeds' resources, follows the common practice, in semantic modelling, of using already well-defined ontologies; it is generic, and it aims to describe any wireless entity utilized in a Fed4FIRE+ experiment.

11.1.1 Deployment of the OMN Wireless Ontology

An OMN Wireless Ontology demo is available to interact at http://147.102.13.123:7200/sparql ("NETMODE-testing" repository) via the GraphDB workbench. Furthermore, some instances of the NETMODE testbed's wireless nodes are created; The figure below depicts a complicated diagram showing the top relationships, where each of them is a bundle of links between the individual instances of two classes. Each link is an RDF statement where the subject is an instance of one class, the object is an instance of another class, and the link is the predicate. Depending on the number of links between the instances of two classes, the bundle can be thicker or thinner and gets the colour of the class with more incoming links.

Co-funded by the Horizon 2020 Framework Programme of the European Union

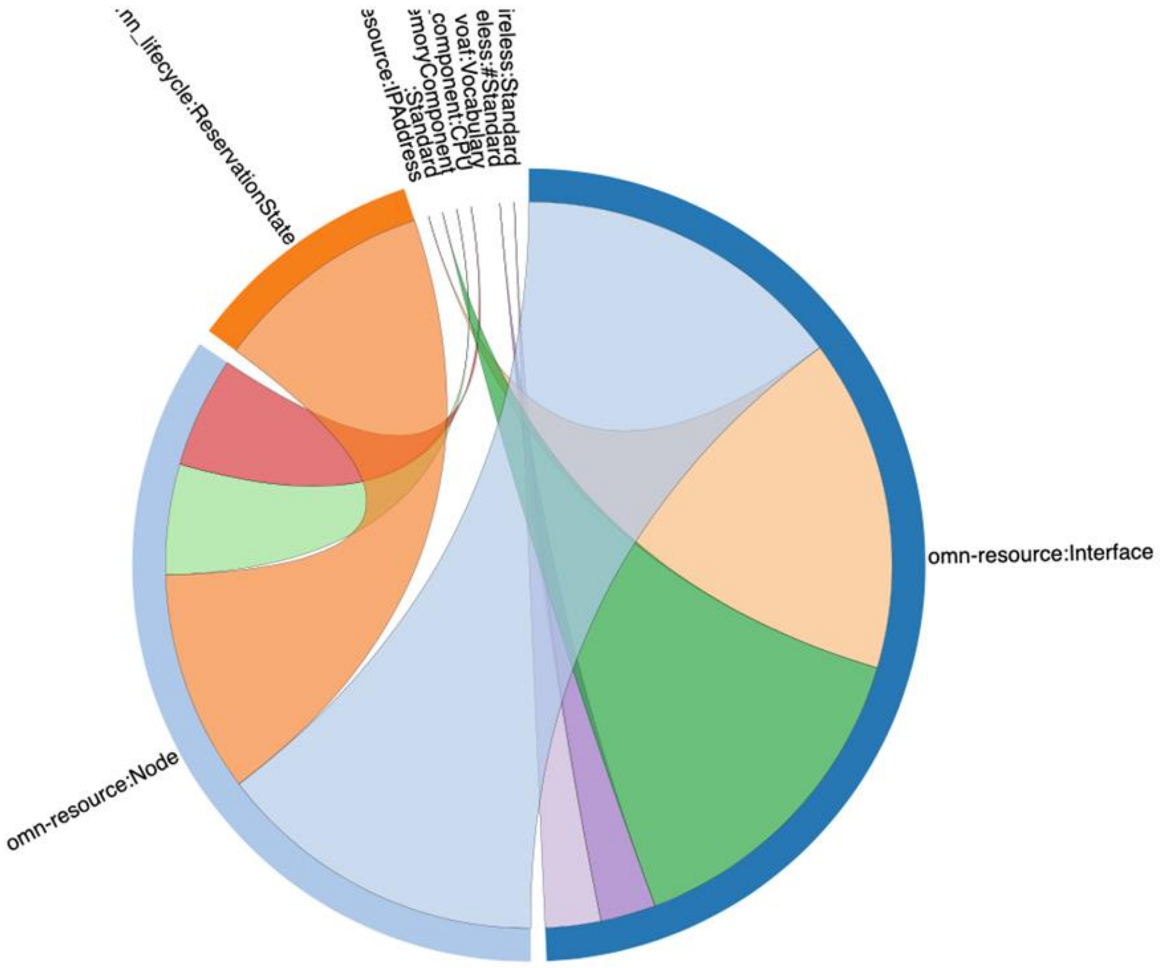


Figure 35: Class Relationships Stats

The figure depicts the graph of one Wireless Node instance, named "Alix01" and its relationships. This Wireless Node is based on an alix3d2 board and it is resource of the NETMODE testbed. Its reservation state is Unallocated. It is equipped with a main RJ45 interface, namely eth0 which supports the 802.3u standard and 2 wireless interfaces, wlan0 and wlan1, which support the 802.11a/b/g standards. These interfaces are instances of the omn-resource:Interface class. Furthermore, the Node is equipped with processor, memory and storage components, as shown; these components are instances of the omn-component:CPU, omn-component:MemoryComponent and omn-component:StorageComponent classes respectively. Data properties are not present in this visualization.

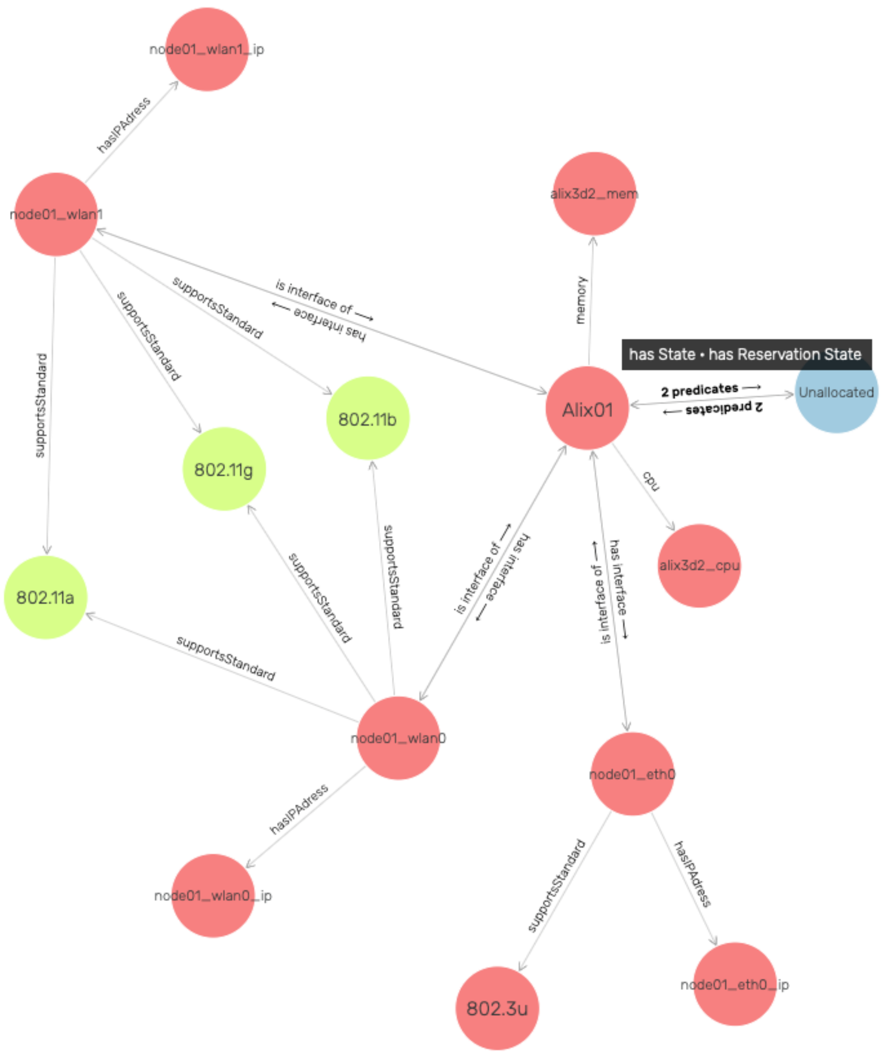


Figure 36: "Alix01" Wireless Node

11.1.2 Reasoning & Knowledge Inference on the OMN Ontology

The Semantic Web is considered to be a layered structure; at its core the Resource Description Framework (RDF) represents triples (subject-predicate-object) in a graph-based data model, independent of a specific serialization (single fact description). In order to describe simple ontologies, however, important vocabularies missing in the RDF specification have been defined with Resource Description Framework Schema (RDFS). By introducing classes, ranges and domains, along with subclassing properties and the accompanying transitive entailment rules, basic ontologies can be defined. In pursuance of constructing larger sets of more complex ontologies that require specifications like equivalency, symmetry or inverse

Co-funded by the Horizon 2020 Framework Programme of the European Union

relations, further sets of semantic annotations and rules have been introduced; Web Ontology Language (OWL) in its two variants, OWL Full & OWL DL.

In general, reasoning over the abovementioned semantic models allows the evaluation of axioms to automatically extend the knowledge within the stored RDF graph. Inside the RDF graph, data is modelled as a set of (named) relationships between resources. Inferencing can be identified as the set of procedures which can generate new relationships based on the existing stored data, along with some additional information in the form of a vocabulary, i.e. a set of rules. Certain software engines exist that apply those logical rules and deduce implicit knowledge. The main reasoning tasks performed by such inference engines are; (i) discovering inferences based on asserted information; (ii) computing all the subclass relationships among the classes, i.e. if concept A subsumes concept B (subsumption); (iii) checking whether the assertions in the Knowledge Base satisfy the given model (consistency); (iv) computing the instance class memberships, i.e. the set of instances belonging to a certain concept.

The Semantic Expressivity implies the Reasoning Capabilities in every Semantic Web's layer. The RDF layer consists of straightforward knowledge representations only, so no reasoning or inference capability is supported. The RDFS, with its more expressive vocabulary offers some limited knowledge extension capabilities through entailment rules. The highest level of expressivity is present at the OWL layer, which adds semantic richness and ontological capability to the schema thus making complete reasoning feasible. In particular, OWL Full provides the highest possible expressivity by allowing unrestricted use of capabilities, meaning that there exists no guarantee that all resulting statements are valid, due to undecidability issues. On the other hand, OWL DL, a syntactically restricted version of OWL Full, offers several production quality reasoners under the Direct Semantics, providing a sound and complete reasoning paradigm. The OMN ontology, is a validated OWL 2 DL Knowledge Base; examples of inferred relationships can be observed in the above figure, in some two-way relationships: isInterfaceOf is the counterpart relationship if hasInterface.

More specifically, in this context, the GraphDB reasoning engine[2] has been integrated to support the aforementioned functionality. GraphDB performs reasoning based on forward-chaining of entailment rules defined using RDF triple patterns with variables. The inference rules are applied repeatedly to the asserted statements until no further inferred statements are produced ("total materialization"). The included repository uses configured rulesets - sets of axiomatic triples, consistency checks and entailment rules, which determine the applied semantics - to compute all inferred statements at load time. By doing so, it has the desirable advantage that subsequent queries evaluation can proceed with no delay.

GraphDB offers predefined semantics by way of standard ruleset files (e.g. rdfs, rfs plus, owl-horst, owl-max etc.), but can also be configured to use custom rulesets with semantics better tuned to any particular domain. The syntax of a rule definition is as follows:

[2]http://ontotext.com/products/graphdb/

Co-funded by the Horizon 2020
Framework Programme of the European Union

```
Id: <rule_name>

    <premises> <optional_constraints>

    ------------------------------

    <consequences> <optional_constraints>
```

11.2 SEMANTIC AGGREGATE MANAGER ARCHITECTURE (SAM)

11.2.1 ARCHITECTURE description

The Semantic Aggregate Manager (SAM) architecture follows design principles which were formed by taking into consideration all the desired features that a manager framework should provide, based on the requirements that were delineated in Section 3. Initially based on the NITOS Aggregate Manager Architecture [StDa15], the framework was modified and expanded to accommodate the support of semantic aware resource descriptions in parallel with SFA-based legacy resource descriptions.

As depicted in Figure 3, the resulted framework is divided in several fundamental architectural components, each of which possesses significant role in a specific problem area of the facility management; (i) the communication interfaces which facilitate the communication with external actors, followed by the (ii) authentication/authorization context which acts as a security intermediary between the internal system and the outside world; (iii) the management layer where most of the framework's intelligence is accumulated, including the orchestration of the supported functionality to fulfil a requested action and the required database transactions.

11.2.2 Functional components

11.2.2.1 Communication Layer

One essential characteristic of the presented management framework is its versatility in terms of communication interfaces or else APIs. In our implementation, two major communication interfaces are utilized. Initially the custom REST API is developed to facilitate support of the experiment's lifecycle while leveraging semantically enriched resource descriptions. Next to that, an XML-RPC API has also been implemented supporting SFA, one of the widely used protocols in the field of testbeds, thus enabling interoperability with existing testbed management platforms.

(Semantic aware) REST API: The REST API is tailored to support the discovery, reservation, provision and release functionality of the networking resources. It leverages the OMN-based [WiPa15] resource descriptions stored in the local Semantic Graph Database, to provide the users/experimenters with semantically enriched information regarding the resources managed by the respective testbed. Thus, the users are able to allocate and provision resources that correspond to their experiments' specifications, as well as release these resources when no longer used. Complementary to this functionality, this API exposes the essential administrative management methods; namely, resource description retrieval, creation, update and deletion are supported.

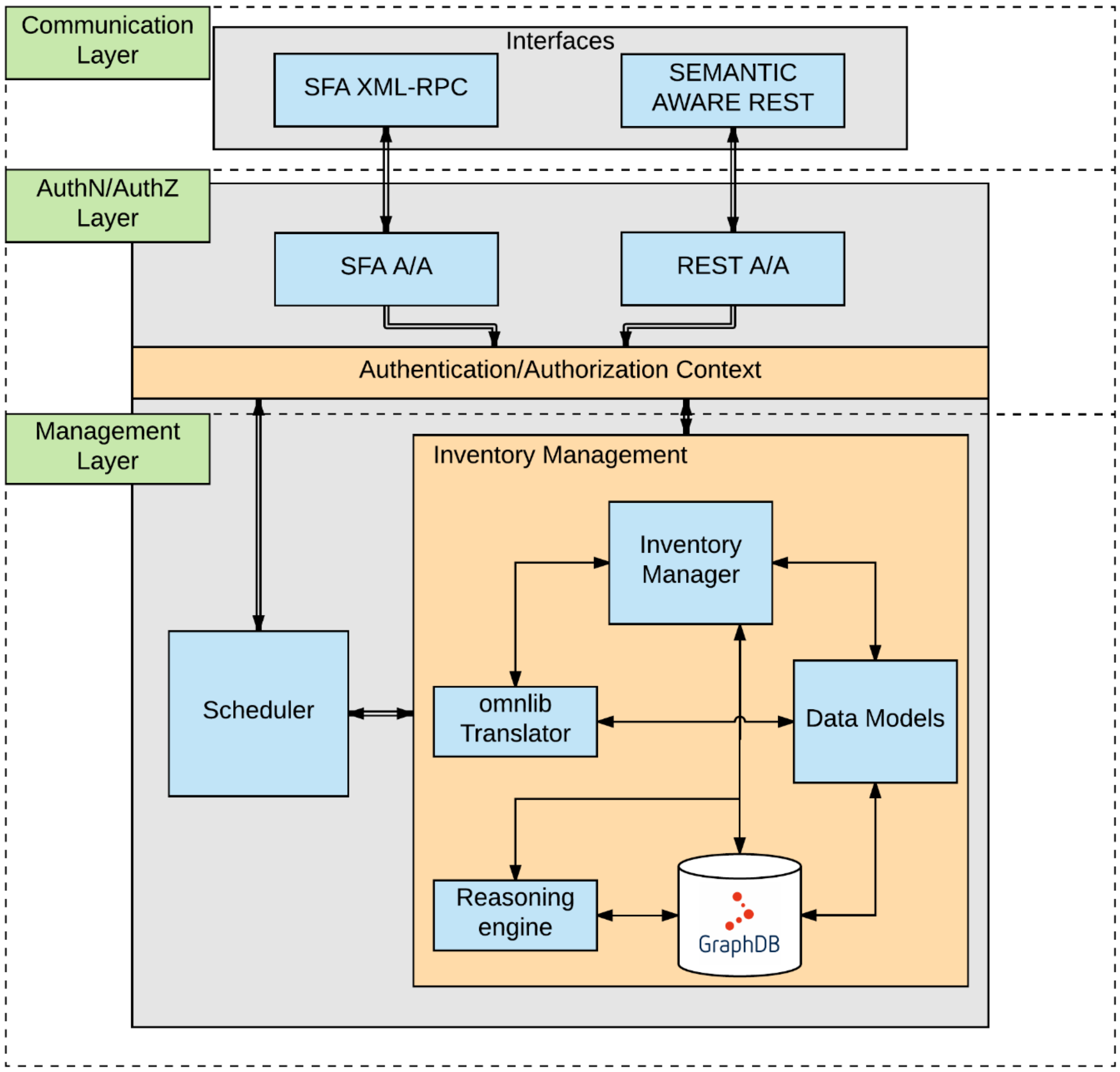


Figure 37: Semantic Aggregate Manager Architecture

(SFA enabled) XML-RPC API: This API is exposed by the AM in order for the SAM platform to be interoperable with existing SFA [PeSe09] enabled provisioning tools (e.g. jFED, omni) and to allow federation with existing testbed management platforms that confront with the SFA and the GENI API v3 [BeCh14] specification. It is practically an implementation of the GENI AM API v3 which is used both in GENI and FIRE [GaKa07] testbeds as a way of federation. Backwards compatibility with GENI AM API v2 is also present, allowing our framework to be

Co-funded by the Horizon 2020 Framework Programme of the European Union

reachable not only by tools that are compatible with the latest API, but also by tools that are compatible with v2 AMs. Throughout the XML-RPC API, multiple arguments and returns are labelled as an RSpec[3]. This resource specification is the primary data structure used within the API and follows a specific set of schemas.

11.2.2.2 Authentication/Authorization Layer

The authorization/authentication system is of paramount importance to any testbed management framework and our platform is no exception. The implemented Authentication/Authorization (A/A) module is where requests invoked in the communication layer are granted approval (or denied respectively), subject to the credentials submitted by the experimenter. Each of the APIs exposed in the communication layer, utilizes its own dedicated mechanism of handling credentials, which can be configured to authenticate the experimenters, thus determining their privileges (authorization).

The aforementioned experimenter privileges are derived through client-side X.509 certificates, in all the interfaces, and are assigned to members of testbeds federated with the given testbed ("certificate authorities"). However, these certificates' structure alternates whether they are targeted at the REST or the SFA communication interface, in a way that they contain distinct attributes meant to be handled by interface-specific methods. To illustrate this specificity, the SFA X.509 certificates use an extension in order to provide a uniform resource name (URN), which enables the testbed Aggregate Manager to link an associated SFA request to a specific experiment. Following the privileges extraction, it is then designated whether the user is permitted to (i) Retrieve, (ii) Create, (iii) Modify or (iv) Release a (a) Resource, (b) Account or (c) Reservation. More specifically, alongside each request comes a signed XML file containing the user's privileges, guiding the A/A module to map them with the abovementioned permitted actions.

The goal of the A/A module is to facilitate the modification and description of fine-grained policies defined by testbed administrators. As mentioned, in our platform this is accomplished thanks to the discrimination of policies which are divided based on their protocol (SFA, REST) and then mapped to a certain set of rules and assertions.

11.2.2.3 2Management Layer

omnlib Translator: One of the core innovations of this work package is the semantic description of experimental resources (OMN ontology) and the development of the respective management mechanisms. In order to support federation with SFA testbeds, which use legacy data formats, it is necessary the API method calls and the respective semantic descriptions to be translated into the respective SFA data models/formats. This component was initially implemented to help developers work with Open-Multinet related ontologies and was included

[3]http://groups.geni.net/geni/wiki/GENIExperimenter/RSpecs

Co-funded by the Horizon 2020
Framework Programme of the European Union

in the OMN suite. In particular, this integration of the omnlib Translator with our platform provides support for translating locally used GENI RSpecs (structured data models) into RDF-based graphs and back. The main advantage of this approach is the automation and speed up of conversion of data the is not using RDF, while ensuring that the quality of the generated RDF data corresponds to its counterpart data in the original system.

Scheduler: The Scheduler component is the provider of the main functionality of the system, since it is the part where decisions regarding resource reservation take place, based on their availability. More specifically, when a request for booking a resource is received in the Communication Layer and forwarded to the Scheduler component via the Inventory Manager, firstly the Scheduler compares the requested booking's start and expiration time with those of the existing, active reservations. Afterwards, it decides based on possible timeslot conflicts and while taking into account the authorization context, whether to fulfil the request or reject it. In our platform, a simple First-Come-First-Served (FCFS) policy is applied to the requests for resource reservation. However, with minor modifications in the Scheduler component, testbeds administrators are able to define their own resource allocation policies (e.g. implementing a user role/status policy).

Inventory Manager: The manager is where all the related policies and resource management is concentrated. Requests regarding resource discovery, booking and reservation, resource provisioning and release (i.e. user tasks), as well as resource description retrieval, creation, update and deletion (i.e. administrative tasks) are forwarded from the Communication Layer to the Inventory Manager. This component facilitates the orchestration and coordination of the actions required to fulfil the aforementioned requests. These actions include, but are not limited to, forwarding the received GENI RSpec to the Translator and receiving the respective Semantic description (and vice-versa), consulting the Scheduler about the feasibility of booking the requested resources, manipulating proper objects which achieve compliance with the established Data Models and storing/retrieving them to/from the GraphDB triplestore, formulating responses and directing them back to the Communication Layer. Throughout the whole process, the manager constantly addresses the A/A Layer in order to access policy-sensitive content and perform policy-sensitive tasks.

Data Models & RDF Triplestore: Taking into consideration the fact that a way to store and access the RDF data model described in section 2.1, an agile, software – integrated way is necessary, the Spira Ruby framework[4] has been utilized. Spira is a framework for using the information in RDF repositories as model objects. It provides the ability to work in a resource-oriented way without losing access to statement-oriented nature of linked data, if preferred. It can be used either to access existing RDF data in a resource-oriented way, or to create a new store of RDF data based on simple defaults. Every ontology class described has been modelled as a Spira Class and each ontology property as a Spira Class property, thus enabling

[4] https://github.com/ruby-rdf/spira

Co-funded by the Horizon 2020
Framework Programme of the European Union

the instantiation and manipulation of RDF Triples as Ruby Objects, leveraging the ORM technique[5].

In our case, an RDF persistent triplestore has been implemented leveraging a GraphDB graph database, as mentioned before. GraphDB is an enterprise ready Semantic Graph Database, compliant with W3C Standards. Semantic graph databases provide the core infrastructure for solutions where modelling agility, data integration, relationship exploration and cross-enterprise data publishing and consumption are important. In addition, inside GraphDB, data are modelled in a way that allows interlinking and querying entities, profiling the relationships between them. Last but not least, it can perform semantic inferencing at scale, allowing users to create new semantic facts from existing facts. This can be achieved through the designation of rules that represent domain specific logic as an explicit part of the data model.

Reasoning Engine: The functionality of this component relies on the specifications delineated in the previous section. Being an intermediate layer between the Inventory Manager and the Triplestore, its main function lies in the deduction of implicit knowledge based on explicit statements stored inside the Triplestore. This extended knowledge may become permanently stored or leveraged to assert miscellaneous requests received in the communication layer.

11.3 SEMANTIC AWARE MANAGEMENT REQUIREMENTS

In Deliverable 3.3, the functional and non-functional requirements were presented. These requirements cover the basic aspects of the semantic aware management of the F4F+ wireless resources. The following tables shows the status of the recorded requirements.

11.3.1 Functional Requirements

Semantic Based Resource Descriptions Requirements

ID	SRD_01
Title	**Semantic Modelling of Experiment Management Concepts**
Short description	All concepts participating in the experiment lifecycle should be modelled and linked using the appropriate semantics. These concepts include F4F+ wireless testbeds and their respective resources utilizing the already developed ontology of OMN suite. The aforementioned heterogeneous concepts should be described in a formalized manner to build a basis for their management based on their semantics, i.e. their underlying meaning and relations.
Relevant Use Case(s)	Discover available resources leveraging semantic descriptions; Discover available resources based on criteria; Discover available resources based on inferred knowledge; Retrieve information about some testbed resources

[5] https://en.wikipedia.org/wiki/Object-relational_mapping

Co-funded by the Horizon 2020 Framework Programme of the European Union

	& reservations; Create semantic descriptions representing testbed's resources; Validate experiment scenarios based on semantic resource descriptions
Type	"Data"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	SRD_02
Title	**Semantic Aware Support of Experiment Lifecycle**
Short description	The system should be able to support the experiment's lifecycle while leveraging semantically enriched resource descriptions. In this context, users should be able to discover, reserve, provision and release the testbed's semantically modelled resources. These are basic functionalities upon which more intelligence can be built. This requirement is of significant importance as it will facilitate the federation-wide interoperability.
Relevant Use Case(s)	Discover available resources leveraging semantic descriptions; Discover available resources based on criteria; Discover available resources based on inferred knowledge; Book resources suitable for the experiment's requirements; Book resources in advance; Check the status a pending/ongoing reservation; Renew an ongoing reservation; Cancel an ongoing reservation and release the respective resources; Access historical data of past reservations
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Ongoing
ID	SRD_03
Title	Semantic Aware Support of Administrative Tasks
Short description	The system should be able to support the administrative tasks while leveraging semantically enriched resource descriptions. In this context, administrators should be able to retrieve, create, update and delete the testbed's semantically modelled resources.
Relevant Use Case(s)	Retrieve information about some testbed resources & reservations; Retrieve information based on criteria; Create semantic descriptions representing testbed's resources; Modify descriptions and ongoing reservations; Delete descriptions; Access historical data of past reservations
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

Co-funded by the Horizon 2020
Framework Programme of the European Union

ID	SRD_04
Title	**Reasoning & Knowledge Inference**
Short description	The system should be able to apply logical rules to the stored knowledge base and deduce new (inferred) knowledge. The integration of a reasoning engine, capable of the aforementioned behaviour is essential for the F4F+ project to exploit the benefits of semantically modelled data.
Relevant Use Case(s)	Discover available resources based on inferred knowledge; Validate experiment scenarios based on semantic resource descriptions
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

SFA Based Testbed Management & Federation Requirements

ID	TMF_01
Title	**SFA Based User Authentication & Authorization**
Short description	Access to resources within a testbed should not be provided to unauthenticated users. Based on the trusted identity of a user and other attributes that will be provided, specific authorization policies should take effect. Adopting SFA, unified authentication and authorization should be based on X509 certificates. The federation Registry (and corresponding APIs) shall be able to authenticate users and provide them with the necessary credentials to book resources.
Relevant Use Case(s)	Authenticate using x509 certificates
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	TMF_02
Title	**SFA Based Support of Experiment Lifecycle**
Short description	The semantic aggregate manager should expose the appropriate API, providing methods that facilitate resource discovery, booking and reservation, resource provisioning and release, thus maintaining compatibility and enabling interoperability with existing SFA enabled infrastructures.
Relevant Use Case(s)	Discover available resources using SFA enabled tools; Book resources suitable for the experiment's requirements; Book resources in advance; Access historical data of past reservations; Check the status a pending/ongoing reservation; Renew an ongoing reservation; Cancel an ongoing reservation & release the respective resources
Type	"Functional"

Co-funded by the Horizon 2020 Framework Programme of the European Union

Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	TMF_03
Title	**Support of XML based RSpec documents**
Short description	In order to successfully integrate SFA, the Resource Specification (RSpec) XML documents should be adopted, alongside semantics, as a common language for describing resources, resource requests and reservations.
Relevant Use Case(s)	Discover available resources using SFA enabled tools; Book resources suitable for the experiment's requirements; Check the status a pending/ongoing reservation; Renew an ongoing reservation; Cancel an ongoing reservation and release the respective resources
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

Interoperability Requirements

ID	INT_01
Title	**Interoperability with SFA enabled Provisioning Tools**
Short description	The semantic aggregate manager should expose the appropriate version of the GENI API in order to facilitate interoperability with the well-known SFA enabled provisioning tools (e.g. MySlice, jFED, omni).
Relevant Use Case(s)	Authenticate using x509 certificates; Discover available resources using SFA enabled tools; Book resources suitable for the experiment's requirements; Check the status a pending/ongoing reservation; Renew an ongoing reservation; Cancel an ongoing reservation and release the respective resources
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	INT_02
Title	**Parallel Use of Semantic & SFA Based Resource Descriptions**

Co-funded by the Horizon 2020 Framework Programme of the European Union

Short description	The F4F+ testbeds should be able to become federated with existing testbeds that support SFA and use XML based RSpecs, while at the same time, the federation should exploit the benefits of a semantic web approach, (i.e data semantics). Translation mechanisms should be adopted in order to facilitate alternation between those data formats.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

General Requirements

ID	GEN_01
Title	**Scheduling of Bookings**
Short description	The system should enable users to book resources in advance. Each available F4F+ wireless resource should be able to become reserved for a period of time defined in a lease tag during the request.
Relevant Use Case(s)	Book resources in advance
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	GEN_02
Title	**Storage of Historical Data**
Short description	Users should be able to access historical data of their entire past executed actions and experiments.
Relevant Use Case(s)	Access historical data of past reservations
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	GEN_03
Title	**Asynchronous Request Processing**

Short description	The system should support asynchronous request processing for the faster completion of time-consuming tasks and the concurrent serving of multiple users.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

ID	GEN_04
Title	**Data Consistency**
Short description	Presented data to the users must be valid and correspond to the current status of their resources and experiments.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Functional"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Implemented

11.3.2 NON-Functional Requirements

Usability

ID	US_01
Title	**Consistent User Experience**
Short description	The semantic aggregate manager should provide consistent user experience.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Usability"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

ID	US_02
Title	**User-friendly Installation**

Co-funded by the Horizon 2020
Framework Programme of the European Union

Short description	Minimal and easy installation of the semantic aggregate manager on top of the testbed's existing infrastructure.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Usability"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

ID	US_03
Title	**User-friendly Configuration and Usage**
Short description	The configuration process and use of the semantic aggregate manager should be easy, intuitive and reliable.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Usability"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

ID	US_04
Title	**Comprehensive Platform and API Documentation**
Short description	The semantic aggregate manager and API documentation should be available, comprehensive and consistent with current functionality.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	"Usability"
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

Reliability

ID	REL_01
Title	**Reliable Service Infrastructure**
Short description	Ensure that the semantic aggregate manager and the relevant services are available at all times.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.

Co-funded by the Horizon 2020 Framework Programme of the European Union

Type	“Reliability”
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

ID	REL_02
Title	**Reliable data, context and content**
Short description	Ensure that context information is as accurate as possible.
Relevant Use Case(s)	This requirement is implied in most use cases where user actors are involved.
Type	“Reliability”
Priority Level	Mandatory
Identified by Partner(s)	NTUA
Status	Accepted

11.4 REFERENCES

[BeCh14] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar. “GENI: A federated testbed for innovative network experiments”. In: Computer Networks 61.3 (Mar. 2014), pp. 5–23.

[GaKa07] A. Gavras, A. Karila, S. Fdida, M. May, and M. Potts. “Future internet research and experimentation”. In: ACM SIGCOMM Computer Communication Review 37.3 (July 2007), p. 89.

[MoWi16] Morsey, M., Willner, A., Loughnane, R., Giatili, M., Papagianni, C., Baldin, I., Grosso, P. & Al-Hazmi, Y. (2016, April). DBcloud: Semantic Dataset for the cloud. In Computer Communications Workshops (INFOCOM WKSHPS), 2016 IEEE Conference on (pp. 207-212). IEEE.

[PeSe09] L. Peterson, S. Sevinc, J. Lepreau, and R. Ricci. Slice-based Federation architecture. Draft version. GENI, 2009. U R L : http://groups.geni.net/ geni/wiki/SliceFedArch (cit. on p. 3).

[StDa15] Stavropoulos, D., Dadoukis, A., Rakotoarivelo, T., Ott, M., Korakis, T., & Tassiulas, L. (2015, May). Design, architecture and implementation of a resource discovery, reservation and provisioning framework for testbeds. In *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)* (pp. 48-53). IEEE.

[WiPa15] Willner, A., Papagianni, C. A., Giatili, M., Grosso, P., Morsey, M., Al-Hazmi, Y., & Baldin, I. (2015). The Open-Multinet Upper Ontology Towards the Semantic-based Management of Federated Infrastructures. EAI Endorsed Trans. Scalable Information Systems, 2(7), e2.

•

Co-funded by the Horizon 2020 Framework Programme of the European Union

12 AUTOMATED OPENSTACK DEPLOYMENT

Because of a high user demand for an easy to set up Openstack environment, we made this available through the ESpec (Experiment Specification). For more details, see https://doc.ilabt.imec.be/ilabt/virtualwall/tutorials/openstack.html .

We use EnOS (https://enos.readthedocs.io/en/stable/) to setup OpenStack. EnOS targets reproducible experiments which allows easy deployment, customization and benchmarking of an OpenStack instance. We have integrated EnOS with the ESpec, so it can be launched easily from jFed.

For the deployment of OpenStack, EnOS uses Kolla-Ansible (https://docs.openstack.org/kolla-ansible/latest/user/quickstart.html) , which deploys the various OpenStack-components in Docker-containers.

EnOS differentiates between 3 types of host machines:

- **control**: runs the *OpenStack Dashboard (Horizon)* and other administrative components like APIs and databases.
- **compute**: hosts the VM's that are deployed on the OpenStack instance
- **network**: hosts the *Neutron*-network agents along with *haproxy/keepalived*.

We provide an ESpec to setup OpenStack via EnOS, which is available on https://gitlab.ilabt.imec.be/ilabt/enos-espec/ . You can instantiate an instance of this ESpec via jFed.

You can run this ESpec in jFed v6 or higher.

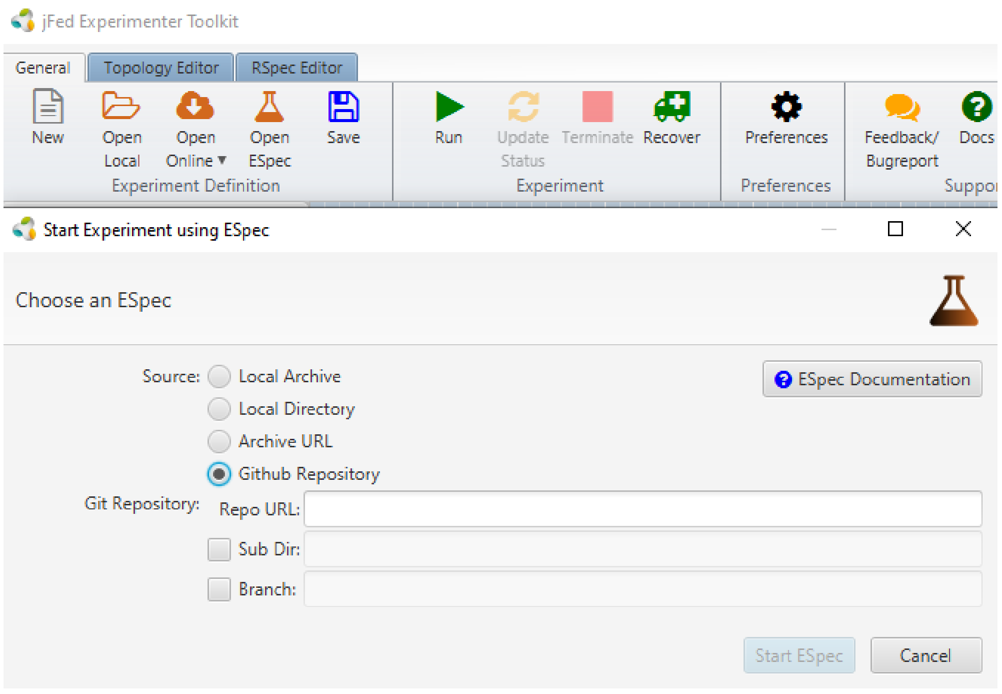
You can edit the number of nodes that will be swapped in by editing the number of nodes in the rspec.

To add a node:

- copy & paste an <node>-element
- change the client_id attribute of the <node>-element and it's child <interface>-elements
- change the ip_addresses in the <ip>-elements of the interfaces
- add a reference to the correct interface in both <link>-elements by adding a <interface_ref>-element

Open and run this ESpec in jFed with the 'Open ESpec'-button. You can

Co-funded by the Horizon 2020 Framework Programme of the European Union



Start Experiment using ESpec

Choose an ESpec

Source: ( ) Local Archive ( ) Local Directory ( ) Archive URL [✓ Github Repository]

Git Repository: Repo URL:

☐ Sub Dir:

☐ Branch:

Figure 38: jFed open ESpec button and direct git access

Deployment typically takes around 1 hour, and is performed from the folder /opt/enos on node0. Interesting log-files can also be found in your home-directory.

While the experiment is swapping in, let's have a look to the architecture:

Co-funded by the Horizon 2020 Framework Programme of the European Union

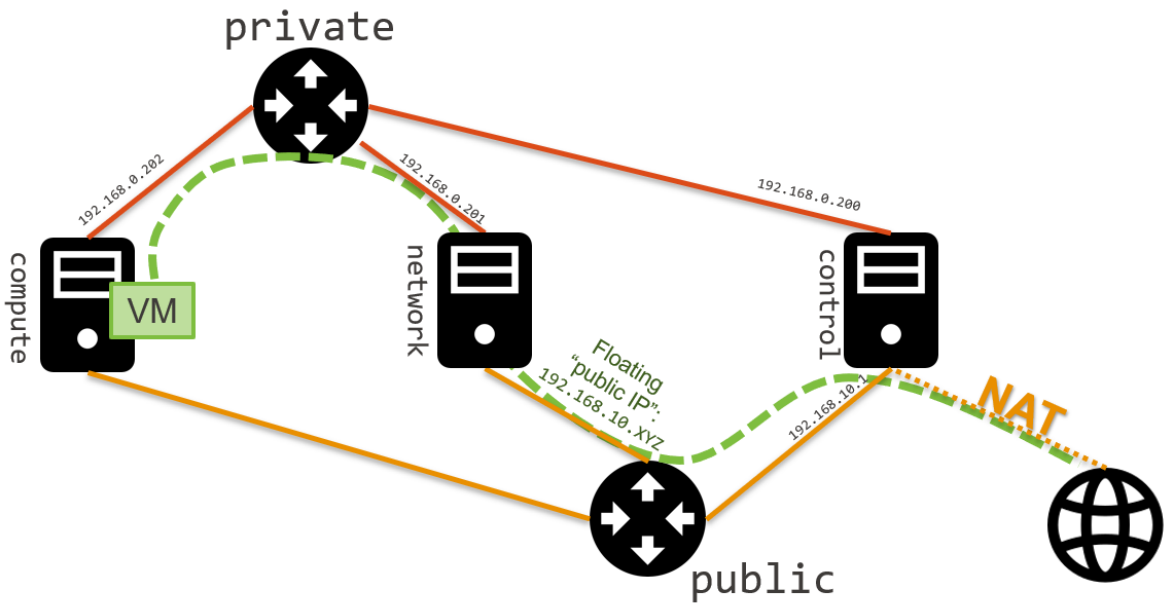


Figure 39: OpenStack setup

The testbed-link with the 192.168.0.0 subnet will be used as the private network for the OpenStack components.

The testbed-link with the 192.168.10.0 subnet will be used as the *public* network for the OpenStack VM's. When assigning a floating IP from the public-subnet to a VM, all traffic from that VM to the public internet will be pushed to that testbed-link.

The *public-subnet* of the *public* network in OpenStack is configured to hand out addresses between *192.168.10.2* and *192.168.10.199*. The gateway for this network is *192.168.10.1* which is assigned to the network interface of *node0* which is connected to this testbed-link. *node0* is configured by *deployment/setup-os-nat.py* to forward all traffic from this interface to the interface of the control network (which has access to the public internet). This also allows you to directly *ping* to OpenStack Server-instances with a "floating ip" attached to them on that node.

Co-funded by the Horizon 2020
Framework Programme of the European Union