



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



D2.9: Testbed requirements, developments and integrations – Update 1

Work package	WP 2
Task	Task 2.2
Due date	31/12/2019
Submission date	29/02/2020
Deliverable lead	Imec
Version	4.0
Authors	Brecht Vermeulen (imec), Wim Van der Meerssche (imec), Thijs Walcarius (imec), Albert (Yiu Quan) Su (SU)
Reviewers	Peter Van Daele (imec)

Abstract	This deliverable describes the specific requirements, developments and integrations that were done in the first 36 months to add new testbeds to the federation.
Keywords	Testbed integration, federation, testbed support

D2.9: Testbed requirements, developments and integrations – Update 1

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1	15/08/2018	TOC	Brecht Vermeulen (imec)
V2	5/03/2020	First complete version	Brecht Vermeulen (imec), Wim Van der Meerssche (imec), Thijs Walcarius (imec), Albert (Yiu Quan) Su (SU)

DISCLAIMER

The information, documentation and figures available in this deliverable are written by the **Federation for FIRE Plus (Fed4FIRE+)**; project's consortium under EC grant agreement **732638** and do not necessarily reflect the views of the European Commission.

The European Commission is not liable for any use that may be made of the information contained herein.

COPYRIGHT NOTICE

© 2017-2021 Fed4FIRE+ Consortium

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.



D2.9: Testbed requirements, developments and integrations – Update 1

EXECUTIVE SUMMARY

This deliverable describes the specific requirements, developments and integrations that were done in the first 36 months to add new testbeds to the federation. In total 28 testbeds were added to the federation, of 15 different types/implementations. Since D2.4, 7 testbeds have been added, 1 testbed was added as associated testbed and 1 testbed was removed (Fuseco does not allow public access anymore).

We have now added also two sections with feedback from the testbed providers on the experience and learnings they gather from open call experiments running on Fed4FIRE.

The federation template that we support together with Geni (Geni Control Framework, GCF) has been further extended, a.o. with general GDPR support. This results in a Docker AM which is fully functional for docker resources (including IPv6).

This deliverable is a full update of D2.4 (and thus includes all info of D2.4 as a complete reference), but the documentation for testbed developers itself is now kept online (as it is updated continuously).

TABLE OF CONTENTS

DISCLAIMER	2
COPYRIGHT NOTICE	2
ACKNOWLEDGMENT	2
1 TESTBED FEDERATION EFFORTS	7
1.1 GENERAL DEVELOPMENT FOR FEDERATING TESTBEDS	7
1.2 FEDERATED TESTBEDS	9
1.2.1 FUTEBOL Brazil/UFES	9
1.2.2 FUTEBOL Brazil/UFMG	9
1.2.3 Futebol Brazil/UFRGS.....	10
1.2.4 FUTEBOL VTT	10
1.2.5 OWL TUD	10
1.2.6 ARNO Sant'Anna Pisa Testbed.....	10
1.2.7 FIT R2Lab	10
1.2.8 ESAT MM Testbed.....	11
1.2.9 Imec City of Things Antwerp.....	11
1.2.10 Fuseco removed (new since D2.4).....	11
1.2.11 Imec GPULab (new since D2.4)	11
1.2.12 Inria Grid5000 (new since D2.4).....	12
1.2.13 Mandat International IoT Lab (new since D2.4)	14
1.2.14 Other testbeds.....	14
1.2.15 Associated testbeds	15
2 SUPPORT FOR TESTBEDS WANTING TO FEDERATE	16
2.1 GETTING STARTED	16
2.2 DOCUMENTATION	16
2.3 USEFUL LINKS	16
3 ADDED VALUE FOR FED4FIRE+ TESTBED PROVIDERS	17
3.1 TESTBED CITYLAB (IMEC)	17
3.2 TESTBED VIRTUAL WALL / W.ILAB-T / PORTABLE TESTBED / GPULAB / TENGU (IMEC)	17
3.3 TESTBED NITOS (CERTH)	17
3.4 TESTBED IOT LAB (MANDAT INTERNATIONAL)	18
3.5 TESTBED NETMODE (NTUA)	18
3.6 TESTBED I2CAT OFELIA (I2CAT)	18
3.7 TESTBED SMARTSANTANDER (UC)	18
3.8 TESTBED TRIANGLE (PERFORMLTE) (UMA)	18
3.9 TESTBED LOG-A-TEC (JSI)	19
3.10 TESTBED IRIS (TCD)	19
4 IMPROVEMENT ON FACILITIES / TOOLS IMPLEMENTED	20
4.1 TESTBED CITYLAB (IMEC)	20
4.2 TESTBED VIRTUAL WALL (IMEC)	20
4.2.1 EXPERIMENT: Stress-test Asvin.io (Asvin).....	20
4.2.2 EXPERIMENT: IIoT-REPLAN (Queens univ of Belfast).....	20
4.2.3 EXPERIMENT: ERASER (UPC).....	20
4.2.4 EXPERIMENT: PiAS (Televic)	21
4.2.5 EXPERIMENT: IntelligentNFVAutoscaler (Modio)	21
4.3 TESTBED W-ILAB.T (IMEC)	22
4.3.1 EXPERIMENT: MMT-IoT (Montimage).....	22
4.3.2 EXPERIMENT: Magic (Galgus).....	22

D2.9: Testbed requirements, developments and integrations – Update 1

4.3.3	EXPERIMENT: Simbed (Inesctec)	23
4.3.4	EXPERIMENT: Five (Feron).....	24
4.3.5	EXPERIMENT: SODA (Univ. of Montenegro).....	24
4.3.6	EXPERIMENT: Comfort-app (WINGS ICT Solutions).....	25
4.4	TESTBED TENGU (IMEC)	25
4.4.1	EXPERIMENT: Scaling NewSum (SciFY)	25
4.4.2	EXPERIMENT: DataTwin (Nissatech)	25
4.5	TESTBED NITOS (CERTH).....	25
4.5.1	EXPERIMENT: CLOUD-RAN BASED LTE-WIFI AGGREGATION (F4F-LWA)	26
4.5.2	EXPERIMENT: EXPERIMENTING IN FED4FIRE+ WITH VEHICLE COMMUNICATION SYSTEMS (FIVE)	26
4.5.3	EXPERIMENT: OFFLINE REAL-WORLD WIRELESS NETWORKING EXPERIMENTATION USING NS-3 (SIMBED)	26
4.5.4	EXPERIMENT: EXPERIMENTAL VALIDATION OF A QOE ANALYTICS FRAMEWORK FOR LTE AND WI-FI (FED4QOE)	27
4.5.5	EXPERIMENT: SEAMLESS URLLC NETWORK SLICE OF NB-IOT (SUNSET)....	27
4.5.6	EXPERIMENT: CDN EDGE-CLOUD COMPUTING FOR EFFICIENT CACHE AND RELIABLE STREAMING ACROSS AGGREGATED UNICAST-MULTICAST LINKS (CDN-X-ALL) 27	
4.6	TESTBED IOT LAB (MANDAT INTERNATIONAL)	28
4.7	TESTBED NETMODE (NTUA).....	28
4.8	TESTBED I2CAT OFELIA (I2CAT)	29
4.9	TESTBED SMARTSANTANDER (UC).....	29
4.10	TESTBED TRIANGLE (PERFORMLTE) (UMA).....	29
4.11	TESTBED LOG-A-TEC (JSI).....	29
4.12	TESTBED IRIS (TCD)	29
5	DOCKER-AM AS EXAMPLE AM.....	32
5.1	SUPPORTED AGGREGATE MANAGER FEATURES	32
5.2	HOW TO INSTALL THE AM ?	33
5.2.1	Dependencies	33
5.2.2	Download source code.....	33
5.2.3	Configure AM	33
5.2.4	Configure a DockerMaster.....	35
5.2.5	Generate certificate and key.....	35
5.3	STARTING THE AM.....	36
5.4	TRUST YOUR C-BAS INSTALLATION	36
5.5	CONFIGURING A REMOTE DOCKERMANAGER (OPTIONAL).....	37
5.5.1	Configure the remote.....	37
5.5.2	Configure the AM	38
5.6	HOW TO ADAPT THIS AM TO YOUR INFRASTRUCTURE ?	38
5.7	DEVELOPMENT NOTES	39
5.8	ADDITIONAL INFORMATIONS	39
5.9	TROUBLESHOOTING	39



LIST OF FIGURES

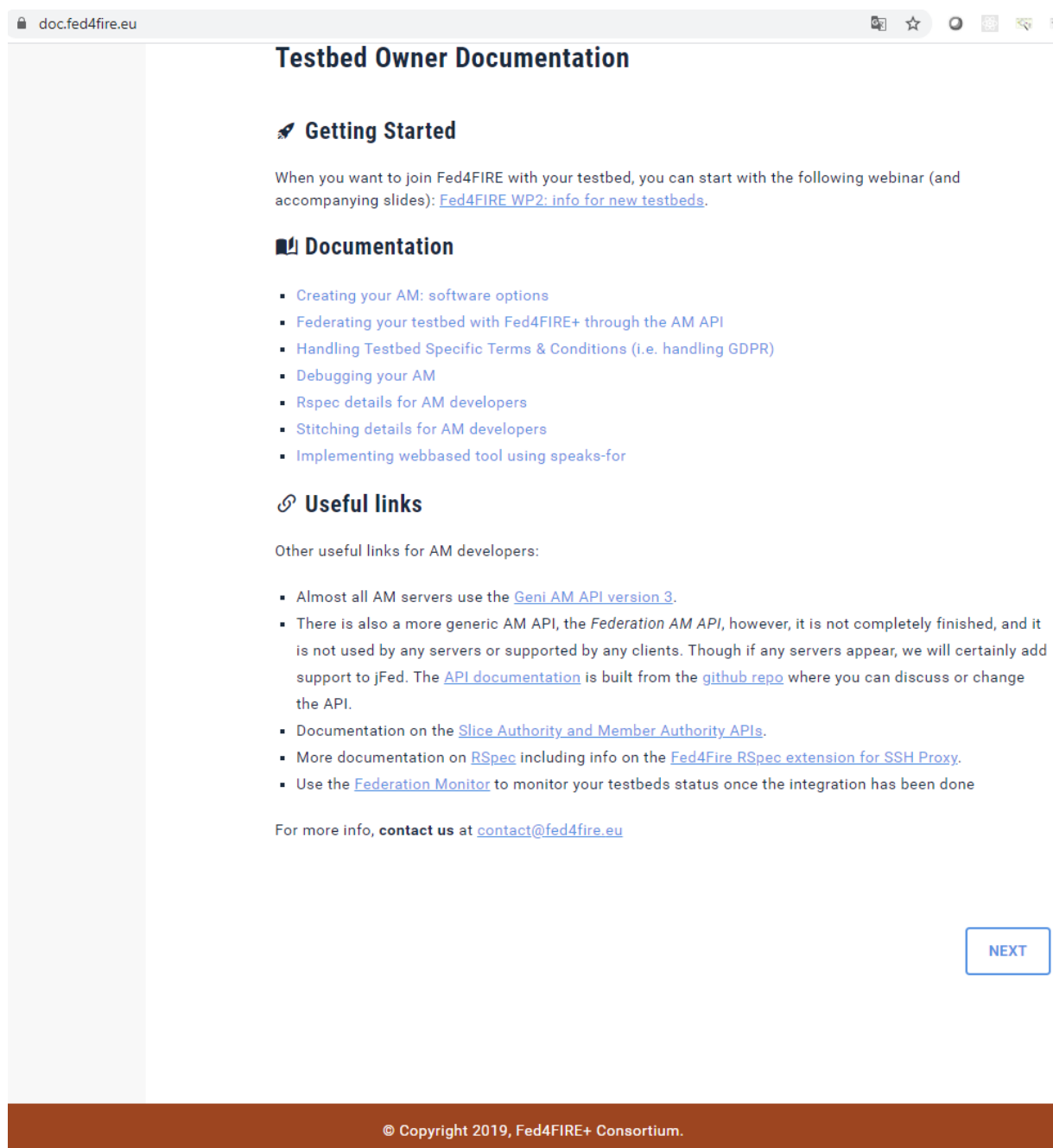
FIGURE 1: OVERVIEW OF FED4FIRE DOCUMENTATION FOR TESTBED DEVELOPERS	7
FIGURE 2: OVERVIEW OF JFED RESOURCE ICONS, INCLUDING NEW ICONS FOR RASPBERRY, 5G, IOT AND GTS (GEANT TESTBED AS A SERVICE)	8
FIGURE 3: THE NEW RASPBERRY PI AND IOT ICONS IN JFED.....	9
FIGURE 4: THE NEW ICONS IN JFED, INCLUDING THE 5G ICON	10
FIGURE 5: A LINK BETWEEN CITY OF THINGS NODES IN JFED.....	11
FIGURE 6: SINGLE-SIGN-ON FOR GPULAB THROUGH OAUTH	12
FIGURE 7: PART OF THE HARDWARE TYPES FOR GRID5000	13
FIGURE 8: CHOOSING A NODE FOR IOTLAB	14



1 TESTBED FEDERATION EFFORTS

1.1 GENERAL DEVELOPMENT FOR FEDERATING TESTBEDS

The documentation aimed at assisting testbed owners when federating their testbeds, was extended on many points, based on feedback and questions from the testbeds. All this documentation was bundled in the updated Fed4FIRE documentation website and can be found at: <https://doc.fed4fire.eu/#testbed-owner-documentation>.



The screenshot shows a web browser window with the URL `doc.fed4fire.eu`. The page title is "Testbed Owner Documentation". The main content is organized into sections:

- Getting Started**: A section with a rocket icon. It contains a paragraph: "When you want to join Fed4FIRE with your testbed, you can start with the following webinar (and accompanying slides): [Fed4FIRE WP2: info for new testbeds](#)."
- Documentation**: A section with a book icon. It contains a list of links:
 - Creating your AM: software options
 - Federating your testbed with Fed4FIRE+ through the AM API
 - Handling Testbed Specific Terms & Conditions (i.e. handling GDPR)
 - Debugging your AM
 - Rspec details for AM developers
 - Stitching details for AM developers
 - Implementing webbased tool using speaks-for
- Useful links**: A section with a link icon. It contains a paragraph: "Other useful links for AM developers:" followed by a list of links:
 - Almost all AM servers use the [Geni AM API version 3](#).
 - There is also a more generic AM API, the *Federation AM API*, however, it is not completely finished, and it is not used by any servers or supported by any clients. Though if any servers appear, we will certainly add support to jFed. The [API documentation](#) is built from the [github repo](#) where you can discuss or change the API.
 - Documentation on the [Slice Authority and Member Authority APIs](#).
 - More documentation on [RSpec](#) including info on the [Fed4Fire RSpec extension for SSH Proxy](#).
 - Use the [Federation Monitor](#) to monitor your testbeds status once the integration has been done

At the bottom of the page, there is a paragraph: "For more info, **contact us** at contact@fed4fire.eu".

A "NEXT" button is located in the bottom right corner of the page content area.

© Copyright 2019, Fed4FIRE+ Consortium.

Figure 1: Overview of Fed4FIRE documentation for testbed developers

D2.9: Testbed requirements, developments and integrations – Update 1

Inside the jFed experiment GUI tool, support for different link types was extended to manage the needs of the Futebol testbeds. jFed will now automatically select the right type of link, based on which nodes (of which testbeds) are connected with each other. This makes it much easier for users, and reduces the need to know all technical details about some testbed links.

Figure 2 gives an overview of the current support resource types (icons on the left), and examples of the new types in the canvas at the right.

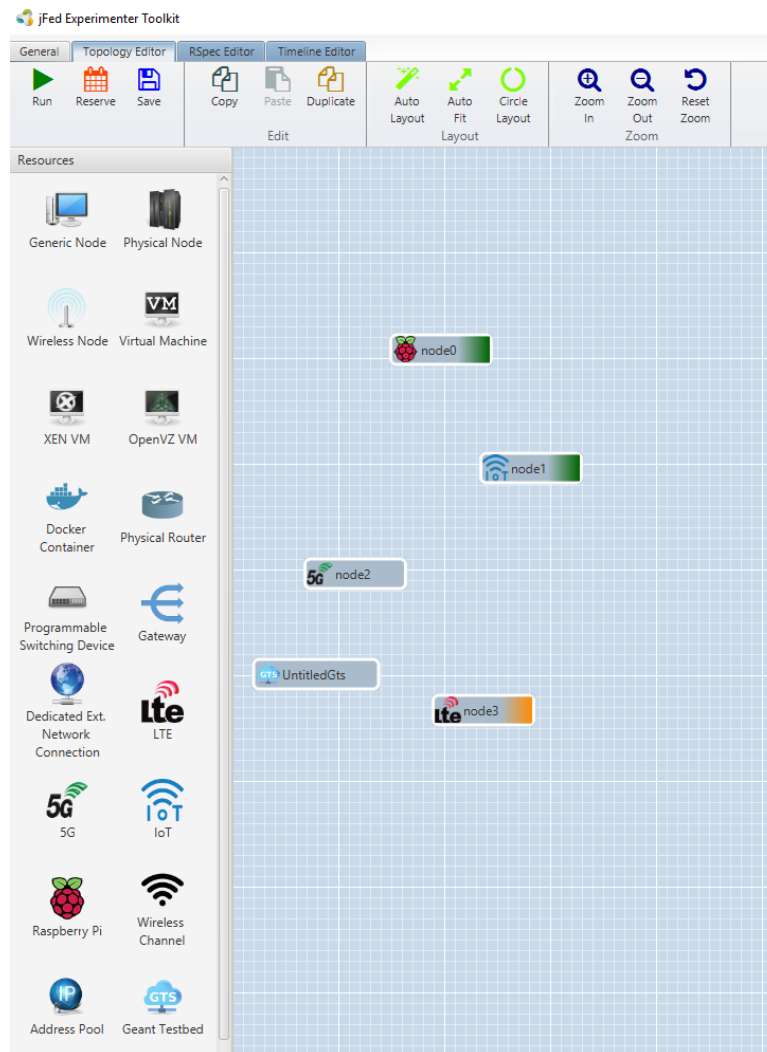


Figure 2: Overview of jFed resource icons, including new icons for Raspberry, 5G, IoT and GTS (Geant testbed as a service)

D2.9: Testbed requirements, developments and integrations – Update 1

1.2 FEDERATED TESTBEDS

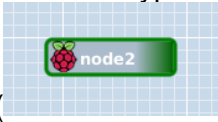
1.2.1 FUTEBOL Brazil/UFES

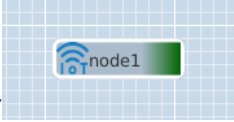
The FUTEBOL testbed of the Federal University of Espírito Santo was federated during April 2017 to Dec 2017. This testbed offers access to VMs. A lot of assistance was required, but no new jFed developments were needed.

1.2.2 FUTEBOL Brazil/UFMG

The FUTEBOL testbed of the Universidade Federal de Minas Gerais was federated begin 2017. This testbeds offers access to a wide range of experimental hardware: USRP (software defined radio) hardware, TelosB sensors, bare metal wifi nodes, and bare metal Raspberry Pi nodes.

jFed was extended with more complex support for hardware and sliver types. This allowed

adding more flexible “icons” in the jFed GUI. A Raspberry Pi () and an IoT

icon () were then added and linked to the appropriate configuration of the UFMG testbed.

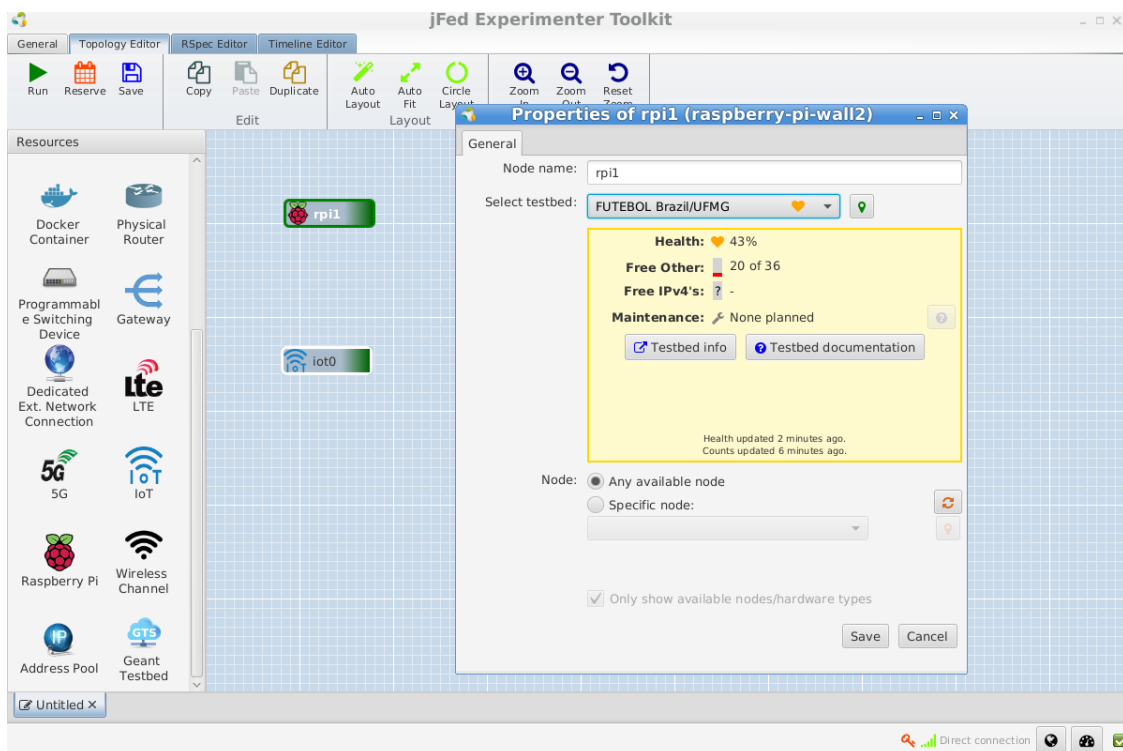


Figure 3: The new Raspberry Pi and IoT icons in jFed

D2.9: Testbed requirements, developments and integrations – Update 1

1.2.3 Futebol Brazil/UFRGS

The FUTEBOL testbed of the Federal University of Rio Grande do Sul was federated begin 2017. This testbed offers access to USRP (software defined radio) hardware, VMs, and bare metal Raspberry Pi nodes.

The federation work required for FUTEBOL UFMG was developed in parallel for this testbed, which had the same requirements. The new Raspberry Pi icon in jFed was linked to the appropriate configuration of the UFRGS testbed.

1.2.4 FUTEBOL VTT

The FUTEBOL VTT testbed of the Technical Research Centre of Finland was federated in April 2018. Little federation support was needed, and no new jFed development was required.

1.2.5 OWL TUD

The Online Wireless Lab (OWL) testbed of the Technische Universität Dresden was federated in March 2018. A lot of testbed side debugging was required.

The biggest issue turned out to be the very long image load time. jFed was modified to handle long load times in a more user friendly way.

A 5G icons was added to jFed, and the testbed was linked to this icon.



Figure 4: The new icons in jFed, including the 5G icon

1.2.6 ARNO Sant'Anna Pisa Testbed

The ARNO Testbed at Sant'Anna Pisa was federated between June and Aug 2017. The testbed is based on the docker AM, and it took some testbed side debugging to get it federated. The testbed offers access to LTE equipment. No jFed development was required for this tested.

1.2.7 FIT R2Lab

The R2Lab testbed of FIT was federated May 2018. This testbed allows access to wireless hardware. No new jFed development was needed, and the testbed was added under the 5G icon.

D2.9: Testbed requirements, developments and integrations – Update 1

1.2.8 ESAT MM Testbed

The ESAT Massive MIMO Testbed of KU Leuven was federated between October and December 2017.

We helped develop the modified GCF AM for this testbed, as some complex feature were needed. jFed was also extended to support some of these features, in particular, support for connecting to windows nodes using RDP was added, and support for working with gateway nodes added by the testbed (not requested by the user) was added. The testbed was added under the 5G icon in jFed.

1.2.9 Imec City of Things Antwerp

The City of Things testbed in Antwerp was federated. This testbed required little federation work. Because the testbed uses special links by default (gre links), jFed was extended to clearly differentiate between different link types in the GUI.

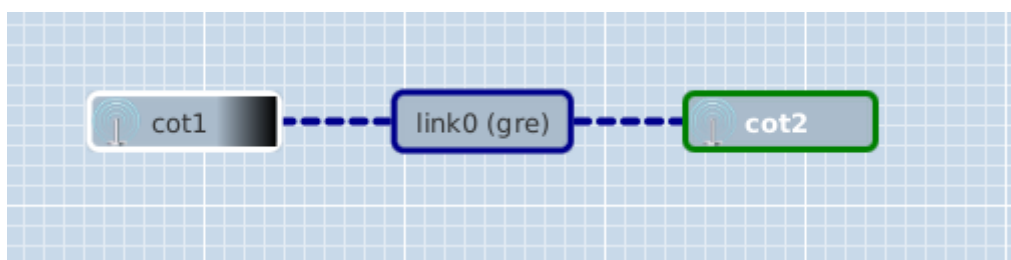


Figure 5: A link between City of Things nodes in jFed

1.2.10 Fuseco removed (new since D2.4)

The Fuseco testbed of Fraunhofer has been removed from the Fed4FIRE federation as Fraunhofer does not allow remote access anymore.

1.2.11 Imec GPU Lab (new since D2.4)

The imec GPU Lab testbed (<https://doc.ilabt.imec.be>) has been opened for public in 2019. GPUs were already available in Virtual Wall nodes (and accessible through jFed), but the problem was that all needed software (CUDA drivers and libraries, machine learning frameworks, etc) had to be installed by the users and this was a burden for the machine learning experts. GPU Lab answers those needs and can be accessed in three different ways:

- Interactively through a Jupyter notebook (<https://jupyterhub.ilabt.imec.be>)
- Website access to launch GPU Lab jobs (mostly used for newcomers) (<https://gpulab.ilabt.imec.be/>)
- Command Line launching of tools through the GPU Lab-cli tool (<https://doc.ilabt.imec.be/ilabt/gpulab/overview.html#command-line-interface>)

For the web-based access, the newly added OAuth functionality of the new user portal is used (see D3.4), so users can login web-based with a single sign on.

D2.9: Testbed requirements, developments and integrations – Update 1

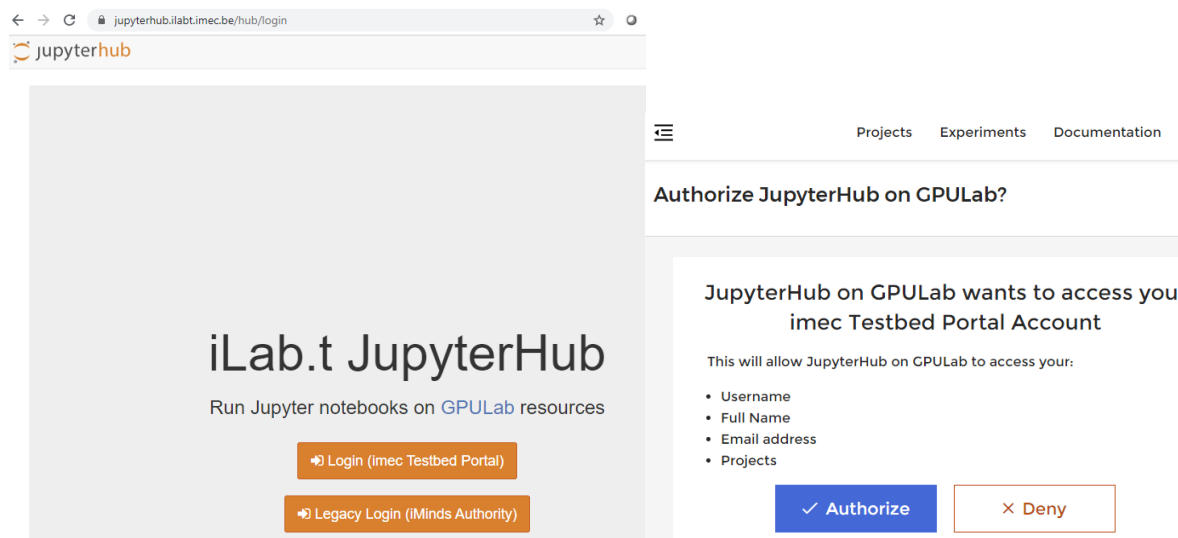


Figure 6: Single-sign-on for GPU Lab through OAuth

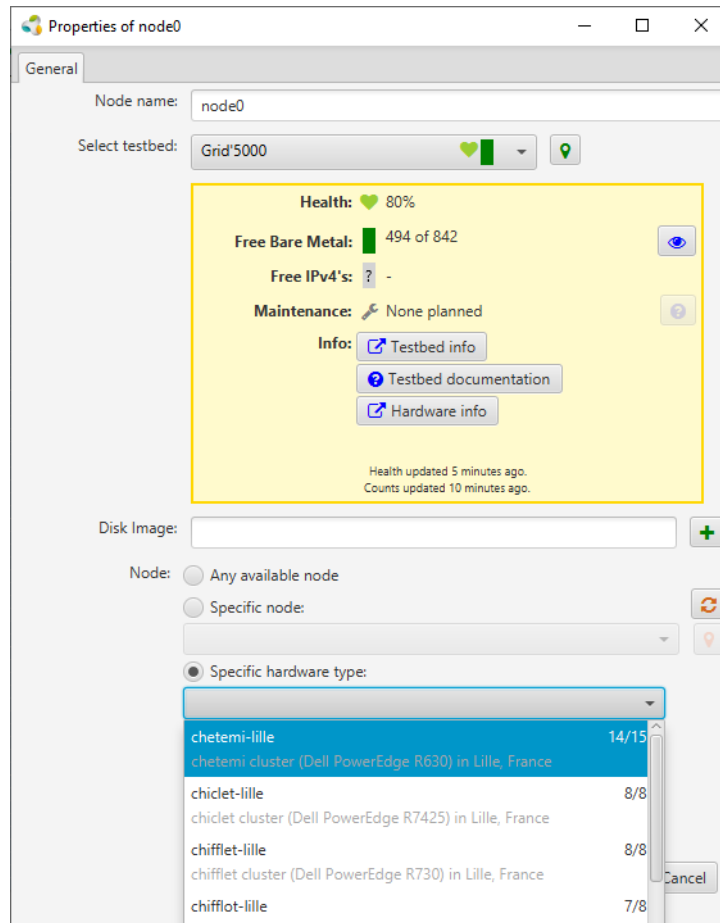
For the command-line, we use the same X.509 certificates (PEM file) as in jFed, see <https://doc.ilabt.imec.be/ilabt/gpulab/cli.html#prerequisites>. These are used to submit jobs and to login interactively with ssh.

1.2.12 Inria Grid5000 (new since D2.4)

Grid5000 has now also been federated (although not all native functionality is supported through the AM API). The following work has been done to support this:

- Support in mapping AM interface to grid'5000 testbed (by clearing up misconceptions and proposing ways to implement certain calls)
- Help in debugging some AM implementation issues
- jFed missing feature added: "testbed proxy" support for ESPEC
- Added development and production server of grid'5000 to jFed config
- Added jFed support for grid'5000 AM unique behaviour (= extra feature and workaround flags added and supported):
- No slice and ssh sharing is allowed by grid'5000: jFed now knows this, and takes it into account.
- grid'5000 AM only supports the user SSL SSH keys, not the full per slice flexibility normally provided by Provision call. jFed now knows this, and takes it into account.
- Multi site is reflected in the URNs: basic jFed support is implemented, more advanced support (= selecting site inside jFed GUI) is to be added.

D2.9: Testbed requirements, developments and integrations – Update 1



The screenshot shows the 'Properties of node0' window with the 'General' tab selected. The 'Node name' is 'node0' and 'Select testbed' is 'Grid'5000'. A yellow box highlights the health and resource information:

- Health: ♥ 80%
- Free Bare Metal: █ 494 of 842
- Free IPv4's: ? -
- Maintenance: 🔧 None planned

Below this box are buttons for 'Testbed info', 'Testbed documentation', and 'Hardware info'. A note at the bottom of the box states: 'Health updated 5 minutes ago. Counts updated 10 minutes ago.'

The 'Node' section has three radio buttons: 'Any available node', 'Specific node', and 'Specific hardware type'. The 'Specific hardware type' radio button is selected, and a dropdown menu is open showing a list of hardware types:

Hardware Type	Count
chetemi-lille	14/15
chetemi cluster (Dell PowerEdge R630) in Lille, France	
chiclet-lille	8/8
chiclet cluster (Dell PowerEdge R7425) in Lille, France	
chifflet-lille	8/8
chifflet cluster (Dell PowerEdge R730) in Lille, France	
chiffot-lille	7/8

A 'Cancel' button is visible at the bottom right of the dropdown menu.

Figure 7: Part of the hardware types for GRID5000

D2.9: Testbed requirements, developments and integrations – Update 1

1.2.13 Mandat International IoT Lab (new since D2.4)

The support for this AM was very straight forward and nothing had to be changed to jFed, apart from adding the testbed info and helping debugging.

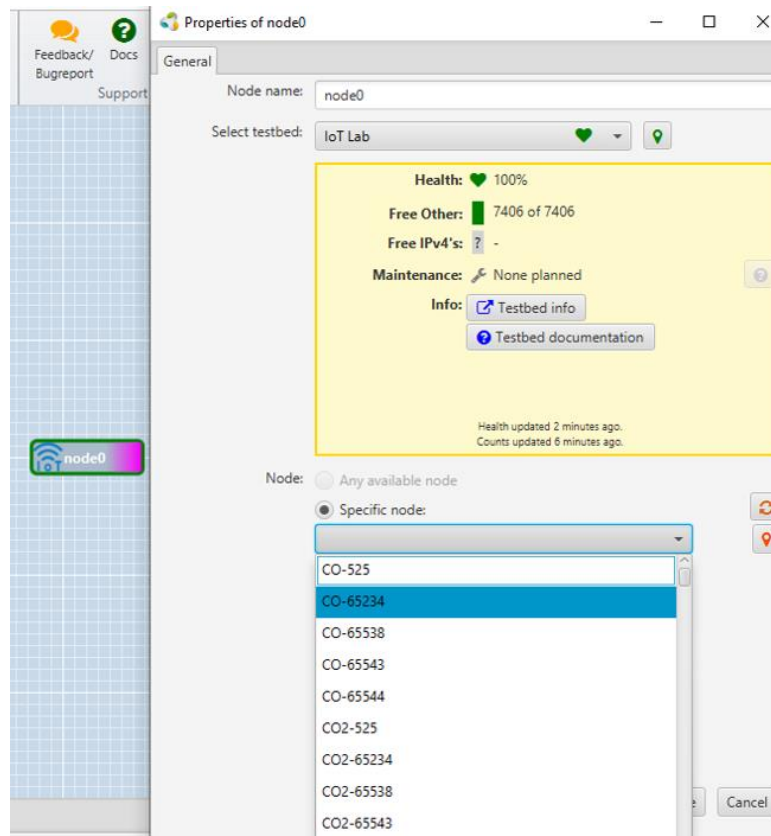


Figure 8: Choosing a node for IoT Lab

1.2.14 Other testbeds

Two development and test servers were added (for University Bristol and UFMG). These are not production testbeds that can be used by federation users.

A lot of new US ExoGeni and InstaGeni testbeds were federated. Adding these to the federation required almost no effort:

- ExoGENI CIENA HQ
- InstaGENI ODU
- InstaGENI Hawaii
- InstaGENI VT
- InstaGENI UVM

D2.9: Testbed requirements, developments and integrations – Update 1

- InstaGENI Louisiana
- InstaGENI UTDallas
- InstaGENI UCSD
- InstaGENI Utc
- InstaGENI University of Washington
- InstaGENI Colorado
- Exogeni UNF (new since D2.4)
- Exogeni LAT (new since D2.4)
- Instageni VCU (new since D2.4)
- Instageni ODU (new since D2.4)

1.2.15 Associated testbeds

One testbed has been added since D2.4 as associated testbed: 5G INNOVATION HUB NORTH run by Luleå University of Technology together with Ericsson, Telia and Tieto.

See <https://www.fed4fire.eu/associated-testbeds/>

The difference between advanced and light federation and associated testbeds is documented at <https://www.fed4fire.eu/add-your-facility/>.

- **Associated:** the testbed is listed on the website, with links to contact information and documentation. This option does not require technical integration.
- **Light:** access to the testbed's resources is realized by exposing a Web-based API. This option does not allow full control over the individual testbed resources, but ensures unified access to experimenters.
- **Advanced:** the testbed is fully integrated in the federation so that experimenters can interact with their experiment during all stages of the experiment's life cycle (resource selection, instantiation, control, monitoring, etc.). This option requires the implementation of the Federation AM API (Aggregate Manager Application Programming Interface) on top of your testbed.

Some testbeds might be included under advanced federation for usage of general resources, while specific features, such as proprietary services, are exposed through light federation.

2 SUPPORT FOR TESTBEDS WANTING TO FEDERATE

For testbeds wanting to federate we have the following things in place:

- Documentation how to start: <https://doc.fed4fire.eu/#testbed-owner-documentation>, see below
- Tools for testing the Aggregate Manager: jFed Probe and jFed Automatic tester (<https://doc.ilabt.imec.be/jfed-documentation-5.9/otherjfedtools.html>)
- An example/template Aggregate Manager with docker containers. This is used as well as front-end for an existing testbed. See <https://github.com/open-multinet/docker-am>
- Documentation on the AM API: <https://github.com/open-multinet/federation-am-api> and <https://fed4fire-testbeds.ilabt.iminds.be/asciidoc/federation-am-api.html>
- Documentation on the Slice Authority and Member Authority APIs: <https://github.com/open-multinet/CommonFederation-SA-MA-API> and <https://geni-nsf.github.io/CommonFederationAPI/CommonFederationAPIv2.html>
- Continuous monitoring and testing, including detailed information when clicking through: <https://fedmon.fed4fire.eu> (see also Deliverable 3.2 for more information on this).

2.1 GETTING STARTED

When you want to join Fed4FIRE with your testbed, you can start with the following webinar (and accompanying slides): [Fed4FIRE WP2: info for new testbeds](#).

2.2 DOCUMENTATION

- [Creating your AM: software options](#)
- [Federating your testbed with Fed4FIRE+ through the AM API](#)
- [Handling Testbed Specific Terms & Conditions \(i.e. handling GDPR\)](#)
- [Debugging your AM](#)
- [RSpec details for AM developers](#)
- [Stitching details for AM developers](#)
- [Implementing webbased tool using speaks-for](#)

2.3 USEFUL LINKS

Other useful links for AM developers:

- Almost all AM servers use the [Geni AM API version 3](#).
- There is also a more generic AM API, the *Federation AM API*, however, it is not completely finished, and it is not used by any servers or supported by any clients. Though if any servers appear, we will certainly add support to jFed. The [API documentation](#) is built from the [github repo](#) where you can discuss or change the API.
- Documentation on the [Slice Authority and Member Authority APIs](#).
- More documentation on [RSpec](#) including info on the [Fed4Fire RSpec extension for SSH Proxy](#).
- Use the [Federation Monitor](#) to monitor your testbeds status once the integration has been done

For more info, **contact us** at contact@fed4fire.eu

3 ADDED VALUE FOR FED4FIRE+ TESTBED PROVIDERS

For each of the testbeds, a short overview is presented below on what is perceived as added value by the testbed owner on the participation of the testbed in the Fed4FIRE+ program.

3.1 TESTBED CITYLAB (IMEC)

The most important feedback we got from Fed4FIRE+ can be summarized into three themes:

- Technologies: CityLab started by focusing solely on unlicensed technology in the city. While executing our experiment, we noticed the need for LTE and C-V2X. The step to adding these technologies was relatively simple (through a combination of an experimental license and a collaboration with the portable testbed for LTE, and opening up our smart highway testbed for C-V2X). This kind of feedback would have been very hard to get upfront from a general audience.
- Locations: experimenters gave us critical feedback on the number of gateways (good), the locations (denser needed) and the relative positioning (more line of sight needed). This kind of feedback was hard to assess upfront; we are very happy with this kind of external validation and feedback.
- Usage: our testbed is oriented very broadly to smart city applications. With the open calls, we were able to provide more specific use cases and share experimenter happiness (which was high). This became a strong instrument in attracting further (industrial) interest in our infrastructure.

3.2 TESTBED VIRTUAL WALL / W.ILAB-T / PORTABLE TESTBED / GPULAB / TENGU (IMEC)

Through the Fed4FIRE federation, we have been able to offer a much better toolset to the users (jFed, Federation monitoring, international testbed access). We have also a louder international voice as the federation of heterogeneous resources has a bigger international impact. Apart from that, we get a larger variety of experiments on our testbeds and as such need to extend the feature set and resources, from which our internal users can benefit then as well. E.g. the international connectivity (within Europe, but also to US, Brazil, Japan) would not have been established without Fed4FIRE. Finally, being part of a larger federation makes also that our internal users can use other testbeds, e.g. if we do not have the right resources, or if we do not have the resources available to them.

3.3 TESTBED NITOS (CERTH)

Through the participation in Fed4FIRE+, NITOS gained visibility and more experimenters through open access and open calls. It should be mentioned that CERTH supported and supports in overall 14 opencall experimenters. Being an active member of the federation it enables NITOS to follow the latest technology trends in testbeds and federation tools and services. Moreover, feedback received by the opencall experimenters allowed us to improve specific aspects of our testbed, which we had overlooked. These include, not updated documentation on recent changes of the testbed and its tools, misbehavior of certain resources and testbed's software tools bugs. Besides visibility and technical improvements, the

D2.9: Testbed requirements, developments and integrations – Update 1

participation in Fed4FIRE+ increased our expertise in testbeds allowing us to be part of 5GinFIRE as a new infrastructure, which supports 5G experimentation.

3.4 TESTBED IOT LAB (MANDAT INTERNATIONAL)

The participation of the IoT Lab testbed into the Fed4FIRE+ testbeds federation permits to the IoT Lab users, in particular the researchers, to access other kinds of resources that IoT Lab doesn't provide: virtual machines or robots for instance. So the complementary of IoT Lab and Fed4FIRE+ is obvious in our case: IoT Lab was designed as a testbed focused on crowd-sourcing and crowd-sensing researches while Fed4FIRE+ testbeds originally encompass more technical researches like the cloud computing or the network communication protocols. Allowing a larger panel of research domains will increase the number of users in Fed4FIRE+ and IoT Lab. The IoT Lab testbed is using the same technologies developed in the frame of Fed4FIRE+ and so, there is a strong collaboration and a lot of synergies between the engineers working in IoT Lab and Fed4FIRE+ testbeds.

3.5 TESTBED NETMODE (NTUA)

As Fed4FIRE+ testbed providers, we have already patroned two open call experiments. The feedback of these experiments was valuable to modify and extend our testbeds. Furthermore, being testbed provider of Fed4FIRE+ consortium improve the visibility of our testbed and allows us to meet researchers from both academia and industry and establish new collaborations.

3.6 TESTBED I2CAT OFELIA (I2CAT)

Our testbed benefits from being part of a catalogue of heterogeneous resources in that it reaches broader audience and therefore it has higher chances of attracting users to experiment in our infrastructure – whether testing alone or in conjunction with other testbeds.

3.7 TESTBED SMARTSANTANDER (UC)

Fed4FIRE+ increases the visibility of our testbed on domains which are not usually targeted by us. This contributes to a higher interest and usage of our platform, hence to a better sustainability of SmartSantander on the international research ecosystem.

SmartSantander is federated following the service oriented architecture approach (as opposed to the SFA/jFed one), so possible technical improvements on the federation tools during Fed4FIRE+ could enable new inter-domain experimentation which can be addressed by using a single testbed.

3.8 TESTBED TRIANGLE (PERFORMLTE) (UMA)

The participation in Fed4FIRE+ enables us to validate the new features offered by the testbed and identify usability enhancements in order to improve the testbed and guarantee its sustainability.

D2.9: Testbed requirements, developments and integrations – Update 1

3.9 TESTBED LOG-A-TEC (JSI)

As a testbed owner and through participation in Fed4FIRE+ we gain broader international visibility of the Log-a-Tec testbed as well as the increased worldwide visibility of our work. Thus, we managed to strengthen the cooperation with the SMEs and research organizations. In addition to the open call experiment and several OC experiments proposals, one of the outcomes is a bilateral project with the University of Banja Luka which includes also the software upgrades of the testbed.

3.10 TESTBED IRIS (TCD)

- The Fed4FIRE+ project has been instrumental in enabling CONNECT at Trinity College Dublin to participate in large international cooperative research projects and support direct collaborations with other research groups of high repute. For example, the infrastructure provided by the Iris testbed, supported by resources provided by the Fed4FIRE+ community, has acted as the foundation for many successful FP7 and H2020 projects that CONNECT at Trinity College Dublin has been involved. These include: WISHFUL, FUTEBOL, ORCA, 5GINFIRE, FORGE, eWINE, etc.
- Participating with highly reputable Universities and industry partners is increasing the impact and depth of our research activities and the quality of our students.
- The contacts and relationships developed supporting Fed4FIRE+ open call experimenters, attending FEC conferences, and interaction with Fed4FIRE+ consortia has encouraged and expedited new project collaborations and consortia with CONNECT at Trinity College Dublin for proposals across H2020, 5GPPP, and EU-Brazil initiatives. For example, CONNECT at Trinity College Dublin became involved in the 5GINFIRE project as a direct consequence of one of these relationships.
- The jFed toolset, testbed monitoring suite, and support provided by the IMEC team are fundamental to the experimenter and researcher success on our testbed. With these tools and continuous monitoring of the health of our testbed, novice and advanced experimenters are having a very good experience using our testbed facilities. The key is ease of use.
- Fed4FIRE+ funded open calls have acted as an excellent starting point support collaborations with Irish and EU SMEs including Tara Hill National Park Teoranta, ALLBESMART, BELETEL, srsLTE, etc.
- The continuously evolving Fed4FIRE+ toolset, the various demands and recommendations from open call experimenters and researchers are driving the evolution and upgrade of our testbed. We see these requirements as fundamental to support our future research and education activities at Trinity College Dublin. We expect these upgrades to the Iris testbed will form a solid foundation for future proposals past H2020.

4 IMPROVEMENT ON FACILITIES / TOOLS IMPLEMENTED

4.1 TESTBED CITYLAB (IMEC)

We have added the smart highway C-V2X testbed, which is closely related to the CityLab testbed, as part of our CityLab offer. This directly answers open questions regarding vehicular networks, and allows us to validate new infrastructure.

Moreover, we have requested an LTE testing license, to further enable LTE testing and collaborations with the portable testbed.

Finally, we have also added ath9k WiFi cards, to support low-level IEEE 802.11 experiments.

4.2 TESTBED VIRTUAL WALL (IMEC)

4.2.1 EXPERIMENT: Stress-test Asvin.io (Asvin)

"The Fed4Fire+ experiment helped us to validate our product for the market fit. Our expectations have been outperformed. Great experiment facilities and helpful insights. Thanks for the Support to all Members of Fed4Fire+."

- They used an automatic docker setup (we used internally already) and scaling where we helped with support.
- this will be added to the public documentation website

4.2.2 EXPERIMENT: IIoT-REPLAN (Queens univ of Belfast)

"We think that the following would be really nice to have and in order to enhance the overall Fed4FIRE+ experimentation experience: A more user-friendly interface for uploading/downloading files to/from the reserved nodes."

- IPv6 at client side will solve this as then you can use all kind of standard scp transfer tools. (as our nodes do not have a public IPv4 address by default).
- To help this (without IPv6), we have added a very basic upload and download possibility in jFed which this experiment used.
- In the meantime it is possible to upload and download files through a jupyter notebook web interface which is very user-friendly (<https://doc.ilabt.imec.be/ilabt/gpulab/storage.html#access-from-jupyterhub>)

4.2.3 EXPERIMENT: ERASER (UPC)

A functionality that could be great to add, if possible, is an experiment queue, allowing to schedule an experiment execution, once any (or several) of the current ones finish. In this way, if the duration of the experiment currently running would be known, the system could be able to give a prediction about the time when the scheduled experiment will start and finish.

Regarding JFed, it might be appropriate to include a node searching operation given the name of the physical machine (e.g., n061-02b.wall2.ilabt.iminds.be) where it is allocated at the experimentation facility. Otherwise, it can be difficult to find it in JFed among a large number of nodes. Moreover, it would be interesting to display information about the image that has been loaded in a node in the JFed GUI, once the scenario is setup and running (in a scenario where different images are installed at network nodes).

D2.9: Testbed requirements, developments and integrations – Update 1

- ➔ About queuing of the experiments: this is done for GPU Lab where the experiments are typically more job oriented. For the virtual wall, there is not really a demand for this by other experimenters because experimenters can typically run their experiment without any problem. This experiment used larger topologies and there we advise currently to contact us. In w-llab.t there is a reservation system, but this has also drawbacks (people are just not finished with their experiment and the slot is over, or they reserve too much time). For the Virtual Wall the current allocation system is the one comfortable for most (>95%) of the users.
- ➔ jFed find a node: this is available in the RSpec view and through the export as node login info (csv so easy searchable), see screenshots below (we have never had this question from the experimenters):

```
node_id, authentication_type, hostname, port, username, iface_id1, mac1, ip1_1, netmask1_1
W2a, ssh-keys, n101-05.wall2.ilabt.iminds.be, 22, bvermeul, W2a:if0,0cc47a7b28a5,192.168.0.1,255.255.255.0,
W2b, ssh-keys, n101-03.wall2.ilabt.iminds.be, 22, bvermeul, W2b:if0,0cc47a7b28f3,192.168.0.2,255.255.255.0,
W1a, ssh-keys, n071-20a.wall1.ilabt.iminds.be, 22, bvermeul, W1a:if0,003048cfb565,192.168.1.1,255.255.255.0,
W1b, ssh-keys, n071-19b.wall1.ilabt.iminds.be, 22, bvermeul, W1b:if0,003048f02585,192.168.1.2,255.255.255.0,
```

Node login information

Node name	Hostname	Port	Username	Login
W1a	n071-20a.wall1.ilabt.iminds.be	22	bvermeul	>_ Login
W1b	n071-19b.wall1.ilabt.iminds.be	22	bvermeul	>_ Login
W2a	n101-05.wall2.ilabt.iminds.be	22	bvermeul	>_ Login

- ➔ info about the image loaded on a node: this was indeed only possible through the manifest RSpec (XML). It's now implemented in jFed beta and will be there in the next release.

4.2.4 EXPERIMENT: PiAS (Televic)

"We were happy with the functionalities as they are"

4.2.5 EXPERIMENT: IntelligentNFVAutoscaler (Modio)

We are overall satisfied by the offering we have had since it helped us in achieving our initial goal and objectives. To that end, the most important help for us was the technical support we received from our Patron, which made possible to overcome the technical issues we have had, in order to be able to reach to a demonstrator of our prototype during the 4th Fed4FIRE+ conference.

D2.9: Testbed requirements, developments and integrations – Update 1

4.3 TESTBED W-ILAB.T (IMEC)

4.3.1 EXPERIMENT: MMT-IoT (Montimage)

"Starting a project with a single reference to the documentation implies a steep learning curve for most of the researchers. In this sense, a presentation of a "project kick start" session would improve this situation, and make all the researchers aware of all the features Fed4FIRE+ has to offer"

- ➔ basic tutorials at the start are useful and necessary -> These people did not attend a tutorial at the start because of the continuous SME call (which is thus not aligned with FECs).
- ➔ a recorded basic tutorial can help as well, so we will work on an update of this.

"More intensive support from the platform operators and a more clear and organized documentation. In this way, it would be possible to deploy and test more rapidly solutions and applications using the Fed4FIRE+ testbeds."

- ➔ we continuously try to update and reorganize the documentation as good as possible
- ➔ a tutorial at the start helps also to answer starting questions

"Valuable types of testbed infrastructures:

1. Support for deploying commercial solutions.
 2. Integration of third-party external IoT networks in the Fed4FIRE+ infrastructure.
 3. Improved interoperability and APIs for simultaneously accessing and automating the access to multiple platforms."
- ➔ The problem with commercial deployments is typically that they are not flexible enough for a testbed. We will try for 802.11ax to provide at least benchmarks of commercial equipment in the same room.

4.3.2 EXPERIMENT: Magic (Galgus)

"One KVM for each type of device (different devices may require different integration steps)."

- ➔ Currently we have KVM (keyboard, video and mouse switch) for the nodes of type Zotac and DSS (same as MOBILE) and the only type which is missing is the APU type. We are looking to add a serial console for this.

"A graphical monitoring tool to show the spectrum usage within the testbed in real time. This would be very useful to check the existence of interference, as well as to validate propagation models."

- ➔ It is possible to monitor the spectrum in 2 ways. A very basic one is using "iwlist scan" which gives you the used frequencies. A more complicated one is using a USRP with GNUradio. This latter needs more setup time from the experimenter indeed. There is also a specific device (imec SE) but it's a long time since someone used that one. We will update the documentation to reflect this.

D2.9: Testbed requirements, developments and integrations – Update 1

“The main bottleneck was the instability of mobile nodes. Nonetheless, we were able to minimize such bottleneck and finalize our experiments in time.”

- This is true, the mobile nodes need a lot of support from testbed admins. These are mechanical parts of the testbed which are less stable than computing nodes.

4.3.3 EXPERIMENT: Simbed (Inesctec)

It would be useful to have a new functionality allowing for radio spectrum reservation, as it happens for other resources such as nodes. Although difficult to enforce, this functionality would help experimenters announcing which channels are being used by them, so that the new experimenters can avoid that mutual interference. For example, NITOS has a spectrum analysis graph that allows checking which channels are currently being used, independently of the technology.

Another thing that can be added is the possibility of changing the current node image without needing to switch for another experiment slice. In the case of w-iLab.t this was not possible, however NITOS offered the possibility to change a node image through OMF.

- Spectrum reservation: as said, reserving spectrum is difficult as the experimenters have full control and root access on all nodes. There are two solutions for this: reserve all resources so you are sure no one else is using parts of the spectrum, or monitor the spectrum. See previous experiment. One cannot really trust a gentlemen’s agreement as also without intention (bug) something can go wrong. (on the other hand the lab is shared a lot of times for less critical experiments, where you monitor and use another channel.
- Changing the current node image: this is indeed something that is lacking in jFed/testbed API. It is natively possible on the testbed, but we need to look into adding this.

We would like to see some additional proactive support, especially in the beginning of the experiment. When the patrons are informed about the experiments that got approved for funding, they could organize a “bootstrap” skype meeting, giving a brief presentation on what they think would be the important tools/methodologies/known pitfalls/tutorials to follow in order to use their testbeds, having in mind the specific objectives of the experiment proposal.

- We try to scale this by giving tutorials, having good documentation, ... most of the users get away with that (especially the open access users).

D2.9: Testbed requirements, developments and integrations – Update 1

4.3.4 EXPERIMENT: Five (Feron)

In FERON we are mainly interested in radio access experimentation capabilities with both closed/commercial equipment (e.g. small cells, smartphones, dongles) and open prototype platforms (e.g. SDRs). This is undoubtedly a rapidly evolving area with new standards emerging for both traditional mobile services but lately also for vertical industries (for example automotive and IoT). In this respect, Fed4Fire+ support for these new communication technologies will be highly beneficial for early technology and related product testing. We report some possible additions in these directions:

- 4G-LTE networks and devices supporting the latest Releases and UE categories. These will enable the testing of new capabilities, such as Gigabit LTE, and new application areas, such Cellular IoT (LTE-M/NB-IoT)
- 5G-NR prototypes (when they become available)
- LoRa/LoRaWAN for IoT experimentation, including development platforms such as Sierra Wireless mangOH, u-blox C030, Pycom, etc.
- ITS-G5 development boards for IEEE 802.11p-based V2X experimentation
- New high-end SDR boards like Ettus USRP N310 for supporting higher bandwidth applications (e.g. NR)
- New small-form & low-cost SDR boards (e.g. LimeSDR/LimeSDR mini, bladeRF) for portable deployment and comparative performance evaluation with other solutions.

➔ Adding hardware is always challenging due to a cost involved and the fact that not all hardware is flexible enough to be put in a testbed and sometimes not useful enough to enough experimenters. That said, the w-iLab.t testbed has been extended with extra SDRs in the meantime, making the following SDRs available now: N210, B200/B210, X310, Zynq SDR. V2X experimentation should be available in Citylab/Smart highway testbed.

4.3.5 EXPERIMENT: SODA (Univ. of Montenegro)

“We would have liked to have a RESTful API for experiment provisioning that we could use for automation purposes. The provisioning process in our case took quite a long time, over 20 minutes, which was not the case on IoT-lab testbed. “

➔ RESTful API: We have a jFed command line version (<https://doc.ilabt.imec.be/jfed-documentation-5.9/otherjfedtools.html#experimenter-cli-2>) which is typically used to automate experiments (together with the ESpec). We have worked on a RESTful API for the F-interop EU project, but there was no real demand to use this API outside the F-interop project. So it has not been maintained anymore. Apart from this, it should be perfectly possible to keep the experiment running, but to automate the experiments with the IoT sensors with a RESTful API (or with the opentestbed API) depending on the needed experimentation. This will also solve the 20 minutes delay (which is because a lot of software needs to be installed as far as we understood. Default setup time is about 2-5 minutes.

D2.9: Testbed requirements, developments and integrations – Update 1

4.3.6 EXPERIMENT: Comfort-app (WINGS ICT Solutions)

We think that the following would be really nice to have and in order to enhance the overall Fed4FIRE+ experimentation experience:

- A more user-friendly API for uploading/downloading files to/from the reserved nodes.
- A jFed tool/extension to save the image of a node in order for the experimenters to be able to reproduce the experiment in the future. One case where this functionality would be extremely useful is for example when you “Reboot OS” and the installed packages of the nodes are uninstalled. Then you have to set the nodes up from zero.

- ➔ More user friendly API for uploading/downloading files: see above. When IPv6 is available at the client, standard scp tools can be used. If not, it is now possible to use web-based jupyter notebook to upload the data to the virtual wall and the copy it from there. It is not ideal. We constantly look how we can improve this. For some heavy users, we provide a VPN, which is not ideal either.

4.4 TESTBED TENGU (IMEC)

4.4.1 EXPERIMENT: Scaling NewSum (SciFY)

“The most difficult part during the experiments where to go through the hadoop and apache spark logs in order to perform debugging activities and troubleshoot errors. Installing commercial debugging tools could help optimize the development workflows and save time.”

- ➔ this is of course at the experiment/application level. Commercial tools have a cost of course. Free tools are available but may have a learning curve. It's a difficult one. We typically look at user demand (if 10% of the users ask for this, then it becomes interesting to add this kind of tool, or if one user deploys it and takes the time to document this, we can add it to the documentation)

4.4.2 EXPERIMENT: DataTwin (Nissatech)

“The existing offering is satisfactory. The only issue is that big data infrastructure should be updated regularly since new tools are appearing frequent”

- ➔ Because Tengu is more based on software stacks (than pure hardware infrastructure in other testbeds), it is more difficult to keep it up to speed with all latest stacks and new tools. This can not be easily solved (needs a lot of manpower). People can always install newest tools themselves or use virtual wall, but it might need some time from them.

4.5 TESTBED NITOS (CERTH)

NITOS was upgraded with the deployment of 4 more USRPS B210 and 2 USRPs N310 towards the support of 5G experimentation, which gains more and more popularity lately. NITOS updated the images offered to the experimenters with the latest version of OAI, that includes additions made by CERTH. Moreover, NITOS continuously updates the local OpenSourceMANO installation, which is an NFV-MANO compliant orchestrator and provides NFV capabilities for the testbed.

D2.9: Testbed requirements, developments and integrations – Update 1

Some improvements / changes were implemented in response to specific comments made by some experimenters:

4.5.1 EXPERIMENT: CLOUD-RAN BASED LTE-WIFI AGGREGATION (F4F-LWA)

Problems encountered / comments mentioned during the course of the experiment or listed in the report - Experimenter's comment:

"We were not able to do experiments only once, since there were no nodes with USRPs available (they were reserved from other experimenters). Other than that, the overall experience was very smooth."

Actions taken / Modifications implemented by the testbed

This experiment ran smooth, other than the problem with no resources during peak hours, there was no feedback regarding any major problem. Most of the problems and questions had been resolved through emails.

4.5.2 EXPERIMENT: EXPERIMENTING IN FED4FIRE+ WITH VEHICLE COMMUNICATION SYSTEMS (FIVE)

Problems encountered / comments mentioned during the course of the experiment or listed in the report

No negative comments in the report. They mention that documentation was sufficient for their experiment.

The experimenters had prior experience with NITOS testbed and they could start their experiment from day 1 of their project.

Actions taken / Modifications implemented by the testbed

More than 60 e-mails were exchanged and 3 telcos were organized during Stage 1 and Stage 2, mostly to discuss the deployment and adaptation of a NITOS ITS software stack implementation.

4.5.3 EXPERIMENT: OFFLINE REAL-WORLD WIRELESS NETWORKING EXPERIMENTATION USING NS-3 (SIMBED)

Problems encountered / comments mentioned during the course of the experiment or listed in the report- Experimenter's comments:

"For NITOS, we found important information missing from the website, such as the attenuators used and in which nodes they are installed."

"For NITOS it was necessary to use a nightly build to be able to properly access the testbed and make reservations."

Actions taken / Modifications implemented by the testbed

We have added the missing information in the documentation of NITOS (http://nitlab.inf.uth.gr/doc/wireless_example.html), so that new experimenters get informed about the attenuators currently attached to NITOS nodes.

IMEC has released a stable version of jFed, which fully supports NITOS testbed and provides the capability to reserve in advance resources.

D2.9: Testbed requirements, developments and integrations – Update 1

4.5.4 EXPERIMENT: EXPERIMENTAL VALIDATION OF A QOE ANALYTICS FRAMEWORK FOR LTE AND WI-FI (FED4QOE)

Problems encountered / comments mentioned during the course of the experiment or listed in the report - Experimenter's comments:

- “In general, yes, however we have felt some hardware limitations to increase the number of UE/cell load in the LTE NITOS testbed.”
- “In the NITOS testbed was not possible to test the impact of cell load on the performance metrics because the lack of UEs compliant with the UXPERT test requirements.”
- “The EPC needs to be updated to be compliant with more recent LTE modems.”

Actions taken / Modifications implemented by the testbed

- There are hardware limitations on the number of UEs that each femtocell supports (max 16/femtocell). This is defined by the hardware characteristics of the femtocells that we have adopted. The experimenters were suggested to move to experimental tools that may potentially provide more users/cell (e.g. OAI) but with less stability.
- We analyzed their experiment needs and suggested a workaround on this limitation by introducing more traffic flows per each UE. With this methodology, the measured indicators by the UXPERT tool would be the same as by having more UEs.
- The COTS NITOS EPC is supporting LTE Rel 10 (LTE Advanced) whereas the installed COTS femtocells are Rel.9 compatible. If more recent releases are needed, we provide all the necessary experimentation tools and infrastructure for running open source implementations of the LTE stack (e.g. OpenAirInterface/srsLTE) which support up to Rel. 14.

4.5.5 EXPERIMENT: SEAMLESS URLLC NETWORK SLICE OF NB-IOT (SUNSET)

Problems encountered / comments mentioned during the course of the experiment or listed in the report - Experimenter's comment:

“YES, we contacted some individuals from NITOS (CERTH) team to request assistance to provide missing components.”

Actions taken / Modifications implemented by the testbed

Experimenter visited us in order to deploy his own equipment and run his experiment in NITOS. However, at the airport the batteries used for their IoT devices were confiscated. We provided the experimenter with batteries that we use in our own IoT devices, allowing this way the experiment to continue. No other comments or problems mentioned in their feedback.

4.5.6 EXPERIMENT: CDN EDGE-CLOUD COMPUTING FOR EFFICIENT CACHE AND RELIABLE STREAMING ACROSS AGGREGATED UNICAST-MULTICAST LINKS (CDN-X-ALL)

Problems encountered / comments mentioned during the course of the experiment or listed in the report - Experimenter's comments:

- Yes, the available resources are useful and inline what our expectations. However, we would like to comment that there are aspects related with USRP testbeds that might be useful to have. For instance, having access to dedicated USRP resources with dedicated LTE band RF duplexers/amplifiers/filters would help to improve the quality of some of our experiments (and stability). This could be indeed an option when reserving USRP resources where an experiment could also ask for specific LTE bands with specific RF frontend configuration.

D2.9: Testbed requirements, developments and integrations – Update 1

- We decided to turn the number of connected UEs smaller to minimize the probability of having one of the UEs or elements from the eNodeB down suddenly. This aspect is quite related to our previous comment. We believe the reliability could be increased with dedicated LTE band RF isolation filters.
- The documentation is partially out of date (resources inventory and OMF syntax) and some atomic and updated examples should be published. It would be useful to include in the documentation the common/recurrent issues that may be found while experimenting and the instructions to solve them.
- Yes, the communication with the patron from NITOS testbed was continuous and fluent. We reported, connectivity issues, environment requirements, further documentation with examples, systems down and architecture requirements. We were quite satisfied NITOS patron provided support, fixing issues quickly and deploying new features to respond to our experiment demands very fast.
- Major aspects have been already addressed but we would like to emphasize that having access to dedicated USRP resources with dedicated LTE band RF duplexers/amplifiers/filters would help to improve the quality of the OAI SDR experience.

Actions taken / Modifications implemented by the testbed

- As the USRPs can be tuned in a wide range of frequencies (60Mhz - 6GHz), handling access for all the available spectrum will block almost all the other experiments. For this purpose, we provide spectrum monitoring tools for the frequencies in the testbed (through the NITOS scheduler) so as the experimenters tune their frequency usage. Specifically to the LTE experimentation, the frequency bands that can be used in the testbed are regulated by the wireless spectrum provider, so experimenters shall use only this spectrum.
- The experimenters run LTE in FDD mode, so installing such functionality would have made no difference at all. As they were running the LTE stack based on the OpenAirInterface implementation, we believe that their instability in the experiments were more related to using a "buggy" version of the software.
- The documentation has been updated accordingly.
- Continuous communication mainly through emails.
- Answered in comment 1 and 2.

4.6 TESTBED IOT LAB (MANDAT INTERNATIONAL)

During the period covered by this report, IoT Lab has technically joined the Fed4FIRE+ federation of testbeds. Thanks to the strong collaboration between the engineers of Mandat International and imec, the integration was mainly planned during the year 2019 taking into account all the technical constraints discovered in the initial phase of technical study done in 2018. Concretely, the IoT Lab integration was effective since the 5th February 2019: indeed, since this date, the status of the IoT Lab testbed is monitored online on Fedmon: <https://fedmon.fed4fire.eu/testbed/iotlab>. Afterwards, Mandat International has done some improvements and updates on the IoT Lab side to correct some bugs and ensure a better service availability at the long run.

4.7 TESTBED NETMODE (NTUA)

NETMODE testbed aims to shift from a wireless testbed to a modern testbed for Mobile Edge Computing experimentation. Towards this directions, an educational robot, i.e. Waveshare AlphaBot, and two Raspberry Pi 3 devices are added in the testbeds.

D2.9: Testbed requirements, developments and integrations – Update 1

4.8 TESTBED I2CAT OFELIA (I2CAT)

During this period, and specifically in the beginning of 2019, we implemented the logic to serve a privacy section in the OFELIA i2CAT Control Framework stack (accessed via the <https://f4f.lab.i2cat.net/privacy/> endpoint). This allows new users to review and accept (or not) the OFELIA i2CAT Terms & Conditions in a fully integrated form with jFed, or to alternatively point their browsers to read the conditions. Their Fed4FIRE certificate is used to extract information and save their preferences on the above stated.

4.9 TESTBED SMARTSANTANDER (UC)

No software upgrades have been carried out during this reporting period. However, new infrastructure based on LoRaWAN technology and new sensor devices have been deployed into the SmartSantander platform.

4.10 TESTBED TRIANGLE (PERFORMLTE) (UMA)

Implementation of new measurement points adapted to the experiment requirements. These measurement points helped to evaluate the main aspects target of analysis increasing also the accuracy of the results reported previously.

4.11 TESTBED LOG-A-TEC (JSI)

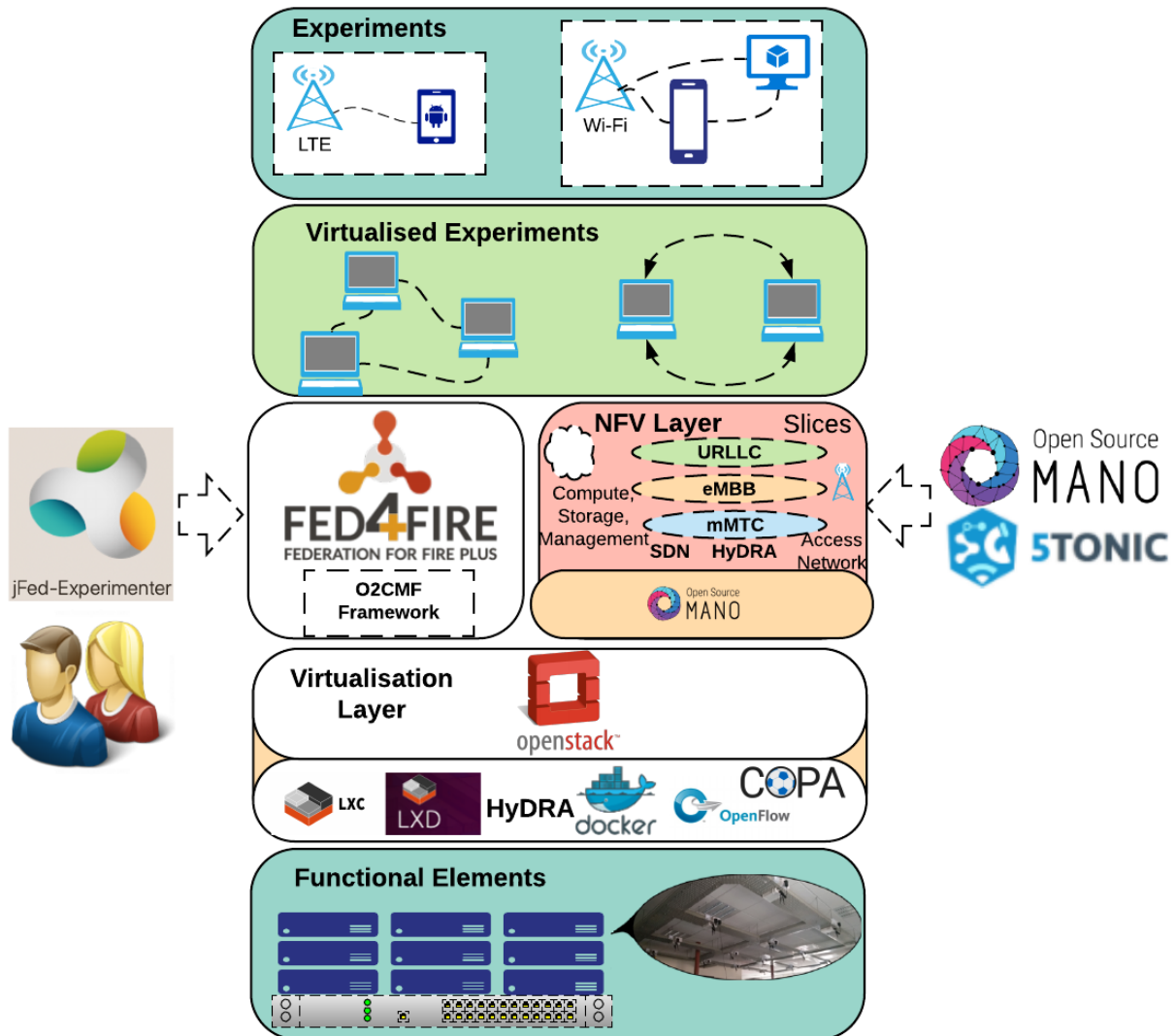
During the reporting period several new software solutions and updates were performed:

- Contiki-NG operating system was ported in order to support TSCH standard for IIoT applications
- Enabling Continuous Integration (CI) approach
- Implementation of the Zero-Touch configuration WiFi solution

4.12 TESTBED IRIS (TCD)

As discussed below in Section 1.3, the WINS_5G 5GINFIRE project objectives were aligned with our goals as infrastructure providers in Fed4FIRE+. Essentially, the WINS_5G 5GINFIRE project has acted as a foundation to the upgrade the Iris testbed infrastructure to support a dynamic 5G cloud environment. Towards integrating the Fed4FIRE+ and 5GINFIRE projects during the second Fed4FIRE+ reporting period (July 18' to Dec 19'), we extended the O2CMF aggregate manager framework, an open platform for control and management of experiments (O2CMF) developed during the FUTEBOL H2020 project, to support SDR devices in the 5GINFIRE OpenStack cloud environment.

D2.9: Testbed requirements, developments and integrations – Update 1



FIRE integration at WINS_5G

O2CMF is an SFA entity responsible for managing testbed resources and Fed4FIRE+ slivers. It was designed to support repeatable and reproducible experimentation of heterogeneous computing and networking resources by utilizing NFV principals. This is seen as an extension to current Fed4FIRE+ and GENI capabilities which did not offer adequate virtualisation mechanisms to support NFV functionality ¹. Due to this extension, both the 5GINFIRE project and Fed4FIRE+ projects share the same cloud-based infrastructure and USRP equipment among project experimenters. The OpenStack VIM central access point for O2CMF and 5TONIC OSM enables VM and USRP resources to be shared between both projects. Furthermore, there is no resource over provision or conflict between project experimenters as the OpenStack VIM allocates resources on a first come, first served (FCFS) basis. This has

¹ Ceravolo, Isabella de A., et al. "O2cmf: Experiment-as-a-service for agile fed4fire deployment of programmable nfv." *Optical Fiber Communication Conference*. Optical Society of America, 2018.

D2.9: Testbed requirements, developments and integrations – Update 1

the added benefit of making Fed4FIRE+ equipment available to the 5GINFIRE community and vice versa. In total, this represents 29 NI USRPs and 13 high powered Dell servers. We plan to make the Iris testbed O2CMF extension to support USRPs available as a branch on O2CMF GitLab ² to the 5GINFIRE and Fed4FIRE+ communities.

As a consequence of the 5GINFIRE cloud upgrade to use OpenStack, the Iris radio testbed now pairs underlying flexible radio and computations resources with various hypervisors in the form of software defined radio (SDR) frameworks, virtualized network functions (VNFs), and SDN network slicing capabilities to realize various research and experimentation configurations. We employ a Dell Networking S4048T-ON high performance SDN/OpenFlow 1.3 enabled switch connected to underlying radio devices. Radio resources include 24 NI USRP N210 ceiling mounted nodes equipped with SBX daughterboards supporting frequency ranges of 400 MHz-4400 MHz offering up to 20 MHz of bandwidth. This equipment supports experimentation with Wi-Fi, WiMAX, S-band transceivers and 2.4 GHz ISM band transceivers, and so forth. We also employ 5 NI USRP X310s supporting DC to 6 GHz frequencies and up to 40 MHz of baseband bandwidth. These platforms are connected to a private computational cloud based on the OpenStack suite of technologies, allowing us to deploy an array of dynamic virtualised computational environments. To expose the functionality of these platforms for applications, we employ a variety of radio hypervisors that freely enable prototyping of wireless systems, as exemplified by GNURadio, srsLTE 3GPP, and Open-Air Interface. These radio hypervisors combined with dynamic distributed network functions enable the realization of heterogeneous radio platforms that can support malleable and adaptable networks.

² O2CMF, <https://gitlab.com/futebol/O2CMF>

5 DOCKER-AM AS EXAMPLE AM

This section describes the features and install instructions for the Docker AM (supporting IPv6 resources as well !), that we advise as example/template of an Aggregate Manager. Code for this can be found at <https://github.com/open-multinet/docker-am> .

5.1 SUPPORTED AGGREGATE MANAGER FEATURES

- Every basic feature (Allocate, Provision, Delete, Status, ListResources, Describe, Renew)
- Some PerformOperationalAction call are supported
 - `geni_update_users` : Update SSH authorized keys or add a user
 - `geni_reload` : If you want to "reset" your container
 - Other options have no effect
- You can provide a sliver-type to get different kind of containers (for example limited memory or CPU container). Check the advertisement RSpec, and have a look at `gcf_to_docker.py` for details.
- Install a custom docker image by providing a name from a DockerHub or a URL to a Dockerfile or a ZipFile containing a Dockerfile and dependencies.
- Restart the AM without losing the state of existing slivers: Running docker containers will keep running when the AM stops, and can be controlled again when the AM restarts. (You can safely remove `am-state-v1.dat` to clear the state and thus force config reload. You will need to kill any running docker containers manually in that case.)
- Multiple physical host for Docker. That means you can increase the scalability easily by setting up a new "DockerMaster" on remote host. To scale the setup, integration with kubernetes is probably preferable.
- `install` and `execute` can be used to install a zipfile in a specific directory and execute commands automatically when the container is ready.
- IPv6 per container can be configured in addition to the IPv4 port forwarding of the host.
- The is demo code that can be used as a basis to customize the AM. Two features are demonstrated in this code: ** Supporting custom non-container external resources. (See `resourceexample.py`) ** Automatically adding a gateway proxy per slice. (See "proxy" in the configuration parsing)



D2.9: Testbed requirements, developments and integrations – Update 1

5.2 HOW TO INSTALL THE AM ?

5.2.1 Dependencies

```
apt-get install -y python2.7 python-xml git python-m2crypto python-dateutil python-openssl libxmlsec1 xmlsec1 libxmlsec1-openssl libxmlsec1-dev python-pip
```

Pyro4 is also required (for remote dockermasters):

```
pip install pyro4
```

5.2.2 Download source code

- First clone the git repository

```
git clone https://github.com/open-multinet/docker-am.git
```

- Initialize submodule (geni-tools source code)

```
cd docker-am
cd geni-tools
git submodule init
git submodule update
git checkout develop
```

5.2.3 Configure AM

Two files are used to configure the AM.

5.2.3.1 gcf_config

This file is for the generic GCF configuration. This contains the basic AM setup. The docker AM specific functionality is activated by setting `delegate` to `testbed.DockerAggregateManager`

Path of this file : `docker-am/gcf_docker_plugin/gcf_config`

- `base_name` : Typically the name of your machine. This is the name used in the URN (`urn:publicid:IDN+docker.example.com+authority+am`)
- `rootcadir` : A directory where your trusted root certificates are stored. These are the certificates of the MA/SA servers who's users the AM will trust. (`wall2.pem` for example)
- `host` : This should be the DNS name of the server. It is used for binding the server socket. `0.0.0.0` is often a good choice if the hostname is not correctly configured on the server.

D2.9: Testbed requirements, developments and integrations – Update 1

- port : You are free to choose a port. 443 is recommended (because it is infrequently blocked by client side firewalls).
- delegate : To activate the docker AM code, this must be `testbed.DockerAggregateManager`
- keyfile and certfile = Private key and certificate of the AM server. This is used for SSL authentication. See the section below.

5.2.3.2 docker_am_config

This is the configuration that is specific for the docker AM.

Path of this file : `docker-am/gcf_docker_plugin/docker_am_config`

The `[general]` section currently contains one parameter.

- `public_url`: the URL to the AM, as advertised in the `Getversion` reply. This URL must contain the FQDN of the host. A raw IP address is discouraged. The following values for the hostname are forbidden here: `0.0.0.0` `127.0.0.1` `localhost`

A `[proxy]` section is also allowed, but not mandatory (no automatic proxy is used if not specified). Check the example config for details.

Each other "section" in the config (a section start with `[name_of_the_section]`) in this file represents a DockerMaster (a dockermaster host one or more containers). If you want to configure multiple DockerMaster just duplicate the first section and change the name. Then, configure parameters :

- `max_containers` : The maximum number of container hosted by your DockerMaster
- `ipv6_prefix` : If you have an IPv6 address on your host, set the prefix in /64 or /80 (for example : `2607:f0d0:1002:51::`) and each container will be assigned an IPv6 in this range
- `dockermaster_pyro4_host`, `dockermaster_pyro4_password` and `dockermaster_pyro4_port` : Parameters to connect to the dockermanager using pyro4 RPC (only when using a remote dockermanager, to use a local docker service, skip these options)
- `node_ipv4_hostname` : The IPv4 of your DockerMaster host (will be used to expose an SSH port on the containers)
- `starting_ipv4_port` : This is the first range used by docker for port forwarding. For example if you set 12000, the first container should be reachable on port 12000, the second on port 12001, ... The AM uses the first port available from 12000 to `12000+max_containers`

D2.9: Testbed requirements, developments and integrations – Update 1

5.2.4 Configure a DockerMaster

You can run the docker service on either the same node as the AM, or use the remote DockerMaster feature (which uses pyro4 for RPC) to run the service on another node.

On the node where docker needs to run, do the following:

First, install the docker engine: <https://docs.docker.com/engine/installation/>

Then, make sure the daemon is running (with systemd) : `systemctl start docker.service`
If you want the docker daemon restart automatically after reboot : `systemctl enable docker.service`

See the section "Configuring a remote DockerManager (Optional)" for more details on a remote DockerManager

5.2.4.1 Configure IPv6 for Docker

If you want to use IPV6 on your container, you have to configure Docker bridge to use a specified prefix.

Create a new file (this path is available on Debian based distribution) `/etc/systemd/system/docker/service.d/docker.conf` :

```
[Service]
```

```
ExecStart=
```

```
ExecStart=/usr/bin/docker daemon --ipv6 --fixed-cidr-v6="2607:f0d0:1002:51::/64" -  
H fd://
```

By replacing the IPv6 example by your own with the proper prefix length (64 or 80)

Then restart your docker daemon : `systemctl restart docker.service`

5.2.5 Generate certificate and key

You need to get a server key and certificate for the AM. You can either get a real one (using your regular way to get SSL Certificate, or using "Let's Encrypt"), or you can create a self signed AM server certificate. In the later case, you will need to add the self signed certificate to the trust store of all clients (which is not a big deal, as you need to add other server info anyway).

It is advised not to use the `bootstrap-geni-am/geni-tools/src/gen-certs.py` script provided by gcf, it does not generate a good AM server certificate. A valid server certificate should have:

- A Subject Name containing a CN equal to the server hostname
- One or more Subject Alternative Names of type DNS matching the server hostname(s)

D2.9: Testbed requirements, developments and integrations – Update 1

- The server hostname mentioned in the 2 points above, should be a DNS name, never a raw IP address
- There is no real need for the certificate to contain a Subject Alternative Name of type URI that contains the URN of the AM. But it is off course no problem if it is included.

Also note that your AM server certificate has nothing to do with the root certificate of your clearinghouse (= MA/SA). The clearinghouse root certificate is used for trusting credentials and for SSL client authentication, it has nothing to do with SSL *server* authentication!

To generate a self signed server certificate, you can use the provided script:

```
cd generate-AM-server-cert/  
./generate-certs.sh
```

Make sure the location of the server key and certificate matches the keyfile and certfile options specified in bootstrap-geni-am/gcf_docker_plugin/gcf_config

You can find some more details on how a certificate can be generated at <https://stackoverflow.com/a/27931596/404495>

5.3 STARTING THE AM

```
sh docker-am/run_am.sh
```

```
Or with the systemd service : cp am_docker.service /etc/systemd/system/
```

Edit the WorkingDirectory according to your installation, reload the systemd daemon

```
: systemctl daemon-reload, then start the AM : systemctl start am_docker.service
```

```
Check the status : systemctl status am_docker.service
```

5.4 TRUST YOUR C-BAS INSTALLATION

If you use C-BAS as Member Authority (MA) and Slice Authority (SA), you have to trust credentials from this. To do, just copy certificates used by C-BAS in your "rootcadir" (configured in gcf_config), usually there stored C-BAS/deploY/trusted/certs.

Restart your AM, if you check the output of the server you should have this at the beginning :

```
INFO:cred-verifier:Adding trusted cert file sa-cert.pem
```

```
INFO:cred-verifier:Adding trusted cert file ma-cert.pem
```

```
INFO:cred-verifier:Adding trusted cert file ch-cert.pem
```

```
INFO:cred-verifier:Adding trusted cert file ca-cert.pem
```

D2.9: Testbed requirements, developments and integrations – Update 1

```
INFO:cred-verifier:Combined dir of 4 trusted certs /root/C-  
BAS/deploy/trusted/certs/ into file /root/C-  
BAS/deploy/trusted/certs/CATedCACerts.pem for Python SSL support
```

Of course this AM is not C-BAS dependent and you can trust the certificate of any MA/SA. For example you can trust the MA/SA from iMinds, so you will be able to create a slice on wall2 and use it on your AM.

5.5 CONFIGURING A REMOTE DOCKERMANAGER (OPTIONAL)

You can set up several DockerManager hosted on different physical machine in order to increase scalability (for example).

5.5.1 Configure the remote

First of all you need to install dependencies on the remote host :

```
apt-get install python2.7 python-pip git  
pip install pyro4
```

And install docker-engine : <https://docs.docker.com/engine/installation/>

Now download the source code repository :

```
git clone https://github.com/open-multinet/docker-am.git
```

```
And try : python2 docker-am/gcf_docker_plugin/daemon_dockermanager.py --host  
127.0.0.1
```

You should get a warning about not using any password and a URI the server listening on.

You can use it in this way but it's more convenient to configure a systemd service. To do this, just copy the service file :

```
cp bootstrap-geni-am/dockermanager.service.sample  
/etc/systemd/system/dockermanager.service
```

Then edit the WorkingDirectory and ExecStart line in this file to match to your configuration. The "--host" parameter should be an IP reachable from the AM, so a public IP or, if your AM is on the same network a private IP.

Finally, do `systemctl daemon-reload && systemctl start dockermanager.service` and check with `systemctl status dockermanager.service`

D2.9: Testbed requirements, developments and integrations – Update 1

5.5.2 Configure the AM

On the AM, edit `docker-am/gcf_docker_plugin/docker_am_config` and add or edit a section to match the three parameters (`dockermaster_pyro4_host`, `dockermaster_pyro4_password`, `dockermaster_pyro4_port`) with the parameters set on the remote

Then delete `am-state-v1.dat` (to force configuration reload) and restart your AM.

5.6 HOW TO ADAPT THIS AM TO YOUR INFRASTRUCTURE ?

If you want to test the AM with your hardware (not with Docker or in addition to Docker) you have to develop your own Python Class which manages your hardware.

You can follow the docker model as example. It is based on three classes : `DockerMaster` (`dockermaster.py`), `DockerContainer` (`dockercontainer.py`), and `DockerManager` (`gcf_to_docker.py`).

- `DockerMaster` is more or less just a pool of `DockerContainers`, because a `DockerMaster` should be a unique physical machine
- `DockerContainer` represents a single docker container, with some information like to ssh port, the IPv6, ...
- `DockerManager` is a generic class to manage docker from Python

So, if you want to represent a physical machine which can be reserved by a user the Python class should be a merge between `DockerMaster` and `DockerContainer`, you should inherit your class from `ExtendedResource`. This class is formed of all used methods by the AM, so you have to implement at least those methods (also have a look to `geni-tools/src/gcf/geni/am/resource.py`)

To kickstart coding this, the class "ResourceExample" is provided. It's dummy external resource manager, which can act as a starting template. You must write configuration processing code in `testbed.py` > `_init_` to enable the resource. The dummy resource does nothing, to get started, edit the file `resourceexample.py` and find lines with `ssh="exit 0;"` and follow the instruction on the lines above. Note that the kickstart code assumes that your AM has SSH access to the external resource.

Once your resources are ready, you have to init them in `testbed.py` in the `_init_` method by adding them to the aggregate configuration parsing. Be sure to delete `am-state-v1.dat` when testing, to force configuration reload.

Note : You should probably implement a generic wrapper for your infrastructure like `DockerManager`, it's easier to maintain, especially if you have different kinds of resources.



D2.9: Testbed requirements, developments and integrations – Update 1

5.7 DEVELOPMENT NOTES

If you want make some contribution to this software, here there is a quick explanation of each file :

- `testbed.py` : The aggregate manager main class, handle calls from the API
- `dockermaster.py` : Docker Master is a pool of docker container, it is called to get some `DockerContainer` instances
- `dockercontainer.py` : Represents a Container with methods to manage it
- `gcf_to_docker.py` : The `DockerManager` class, used as generic wrapper for Docker in Python, mostly used by `DockerContainer`
- `resourceexample.py` : A dummy resource to kickstart you to develop your own resource
- `extendedresource.py` : A generic resource class which adds some usefull methods to the base `Resource` class (which is in `resource.py`, in the `geni-tools` repo)
- `daemon_dockermanager.py` : The daemon used to create a remote `DockerMaster` using `Pyro4` framework.

5.8 ADDITIONAL INFORMATIONS

- Objects are serialized in `docker-am/am-state-v1.dat`, so you can restart the AM without consequence
- Slivers expiration is checked every 5 minutes, and on each API call
- Warning : If you restart the host, docker containers are lost, to keep consistent state delete `am-state-v1.dat` before restarting the AM.
 - It will mostly work without deleting the file but you could have some unexpected behaviors

5.9 TROUBLESHOOTING

- If you get the error "Objects specify multiple slices", you probably made a typo in `component_manager_id` (during `allocate` call)
- If your configuration is not taken in account, delete `docker-am/am-state-v1.dat` and remove all running containers `docker rm -f $(docker ps -a -q)`
- If you get an SSL error (like host not authenticated) check if you correctly add your AM/SA certs in trusted root

