Wireless Software and Hardware platforms for Flexible and Unified radio and network controL

Project Deliverable D5.2

Initial implementation of experimentation tools

Contractual date of delivery:	31-12-2015
Actual date of delivery:	23-12-2015
Beneficiaries:	IMINDS, TCD, TUB, UFRJ, RUTGERS, NCENTRIC
Lead beneficiary:	IMINDS
Authors:	Pieter Becue (IMINDS), Vincent Sercu (IMINDS), Ingrid Moerman (IMINDS), Nicholas Kaminski (TCD), Luiz DaSilva, Ivan Seskar (RUTGERS), Mikolaj Chwalisz (TUB), Kai Geißdörfer (TUB), Jose de Rezende (UFRJ)
Reviewers:	Nicholas Kaminski (TCD)
Work package:	WP5 – Fed4FIRE compliance
Estimated person months:	12
Nature:	R
Dissemination level:	PU
Version:	3.4

Abstract:

This deliverable reports on the activities in the first year of the WiSHFUL project. The first activity was to bring all WiSHFUL testbeds up to Fed4FIRE federation standards. It is shown that w-iLab.t, ORBIT, FIBRE Island @ UFRJ, TWIST and the Portable Testbed are fully Fed4FIRE compliant, while the IRIS testbed is expected to finalize their federation efforts early 2016. This enables the use of unified experimentation tools across all testbeds. For resource provisioning, the jFed tool was chosen, OMF6 enables automated experiment control and OML provides a uniform measurement tool. Access to the testbeds is provided through Fed4FIRE certificates. Installation of the above tools is made easy through the use of the IT automation tool Ansible. This deliverable is concluded by describing the integration of the WiSHFUL UPIs in testbeds. The proposed approach will result in a loose coupling between testbed tools and WiSHFUL UPIs, while maintaining a clean separation between experiment control and WiSHFUL control.

Keywords:

Fed4FIRE compliance, Experimentation tools, jFed, OMF6, OML, Ansible, UPI testbed integration, WTA GitHub, MySlice

Executive Summary

This Year 1 deliverable reports on the status of Fed4FIRE compliance of all WiSHFUL infrastructures. Most of the testbeds have now reached full Fed4FIRE compliance, namely w-iLab.t, ORBIT, FIBRE Island @ UFRJ, TWIST and the Portable Testbed. The IRIS testbed is expected to converge to Fed4FIRE compliance early 2016. All WiSHFUL testbeds aim to support the use of jFed for provisioning of the resources. This will allow the experimenter to use the same tool and certificate to access all testbeds in WiSHFUL. The OMF6 framework allows experimenters to do uniform experiment control on WiSHFUL infrastructure, while the OML instrumentation tool offers a generic software framework for the collection of measurements. Install scripts for the tools mentioned above are provided in the form of Ansible playbooks and are available in the ExperimentationTools repository in the WirelessTestbedsAcademy GitHub account. Sample configuration files for jFed, OMF and OML can be found in the same repository. The approach for the integration of WiSHFUL UPIs in testbeds is proposed and will be used as a guideline for the implementation during Year 2 of the project.

List of Acronyms and Abbreviations

AM	Aggregate Manager
AMQP	Advanced Message Queuing Protocol
AP	(Wireless) Access Point
CH	Clearinghouse
EC	Experiment Controller
FRCP	Federated Resource Control Protocol
F4F	Fed4FIRE
GCF	GENI Control Framework
GENI	Global Environment for Network Innovations
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
MCE	Monitoring and Configuration Engine
OEDL	OMF Experiment Description language
OMF	cOntrol and Management Framework
OML	(Orbit) Measurement Library
OMSP	OML Measurement Stream Protocol
PoE	Power over Ethernet
RC	Resource Controller
RAM	Random Access Memory
Rspec	Resource Specification
SFA	Slice-based Federation Architecture
SSH	Secure SHell
TCP	Transmission Control Protocol
UPI	Unified Programming Interface
VPN	Virtual Private Network
XML	Extensible Mark-up Language
XMPP	Extensible Messaging and Presence Protocol

Table of contents

1 Introduction

This deliverable reports on the work done in WP5 “Fed4FIRE compliance” and gives an overview of the tools that are offered as part of the first experimentation toolset.

The document first gives an overview of all WiSHFUL testbeds and their status of Fed4FIRE compliance. As shown in the overview table in chapter 2, most of the testbeds have now reached full Fed4FIRE compliance, namely w-iLab.t, ORBIT, FIBRE Island @ UFRJ, TWIST and the Portable Testbed. TCD is completing the final phases of testing for SFA support on the IRIS testbed and is expected to achieve Fed4FIRE compliance by March 2016.

Chapter 3 can be used as a guideline for experimenters. It contains information about all the necessary steps to run an experiment on WiSHFUL infrastructure, including the use of the Fed4FIRE tools. For every testbed, it is described which tools they support for the following stages in an experiment: resource provisioning, experiment control and experiment measurements. All WiSHFUL testbeds aim to support the use of jFed or MySlice for provisioning of the resources. The OMF6 framework will be supported on most of the testbed and allows experimenters to do uniform experiment control on WiSHFUL infrastructure. The OML instrumentation tool offers a generic software framework for the collection of measurements and will also be supported on most of the WiSHFUL testbeds. Install scripts for the tools mentioned above are provided in the form of Ansible playbooks and are available in the ExperimentationTools repository in the WirelessTestbedsAcademy GitHub account [1].

The final chapter in this deliverable (chapter 4) is dedicated to the integration of WiSHFUL UPI’s in testbeds (see section 4.1) and a tool for the automation of software deployment (see section 4.2). The integration of WiSHFUL UPI’s in testbeds is split into separate entities in the experimentation process: Node Discovery (see section 4.1.1), Experiment Setup (see section 4.1.2), Experiment Control (see section 4.1.3), WiSHFUL control (see section 4.1.4) and Monitoring & Measurements (see section 4.1.5). By using the proposed loose integration, a clean separation is made between experiment control (offered by testbed tools) and WiSHFUL control. To the experimenter, the WiSHFUL control software can be considered as extra functionality that can be included in his experiment to allow for more advanced control strategies.

2 Fed4FIRE compliance of WiSHFUL testbeds

This chapter gives an overview of all WiSHFUL testbeds and their status of Fed4FIRE compliance. Tasks that are not yet finished are shown in orange. These tasks are planned to be completed during Y2 of the project. The following sections give more detailed info per testbed.

	w-iLab.t (IMINDS)	**ORBIT (Rutgers)**	**FIBRE Island @UFRJ**	**IRIS (TCD)**	**TWIST (TUB)**	**Portable testbed**
Discovery	SFA (GENI AMv3)	XML SFA Planned (GENI AMv3)	SFA (GENI AMv2)	SFA Planned	RESTful / SFA (GENI AMv3)	SFA (GENI AMv3)
Requirements	SFA (GENI AMv3)	XML SFA Planned (GENI AMv3)	SFA (GENI AMv2)	SFA Planned	RESTful / SFA (GENI AMv3)	SFA (GENI AMv3)
Instant Reservation	SFA (GENI AMv3)	Web based (XML)	SFA (GENI AMv2)	SFA Planned	Web based / SFA (GENI AMv3)	SFA (GENI AMv3)
Provisioning	SFA (GENI AMv3)	custom (reservation based exclusive access)	SFA (GENI AMv2)	SFA Planned	custom (reservation based exclusive access) / SFA (GENI AMv3)	SFA (GENI AMv3)
Experiment Control	SSH or FRCP (OMF6)	SSH or FRCP (OMF5.5 + OMF6.0)	SSH or OMF 5.4	SSH (FRCP Planned)	SSH/Custom	SSH FRCP (OMF6)
Facility Monitoring	Zabbix +OML	OpenNMS + OML	ZenOSS	OML Planned	cacti + collected	None
Connectivity	Public IPv4 for AM Public IPv6 for resources	Public IPv4 + Firewalled IPv4/IPv6	Public IPv4 for AM Tunnel to private IPv4 for resources	Public IPv4 for AM Tunnel to private IPv4 for resources	Public IPv4/ VPN	Depending on location of installation
Documentation	OK	OK	OK	OK	OK	Partial
Policies	OK	ORBIT+GENI	OK	To be updated	OK	To be updated
Federation Level	Fully F4F compliant	GENI	Fully F4F compliant	Planned Full F4F compliance	Fully F4F compliant	Planned Full F4F compliance

2.1 w-iLab.t (IMINDS)

The w-iLab.t testbed is fully F4F compliant. Detailed information can be found in D5.1 section 2.6.

2.2 ORBIT (RUTGERS)

The ORBIT testbed is fully compliant with GENI. Detailed information can be found in D5.1 section 2.5.

2.3 FIBRE Island @ UFRJ

The FIBRE Island @ UFRJ is fully F4F compliant by the use of SFA AMv2. Detailed information can be found in D5.1 section 2.4.

2.4 IRIS (TCD)

TCD is currently deploying F4F support within its facility. At the time of writing, TCD is completing the final phases of testing for SFA support based on GENI control and federation software. In parallel to these efforts, TCD is developing resource descriptions in the form of RSpecs and mechanisms for automatic connectivity provisioning. Full support for SFA functionality is expected to be completed by March 2016.

Following the completion of SFA support, TCD will integrate OML support into the testbed monitoring. This functionality is expected within 2016.

2.5 TWIST (TUB)

a. SFA

TWIST Aggregate Manager supports SFA 2.0 by implementing the GENI Aggregate Manager API v3. The implementation is based on the reference gcf Aggregate Manager provided with geni-tools. The south side interface uses Ansible (IT automation tool), which allows to dynamically alter the available resources and perform reservation process independently to the AM API. Currently there is support for a number of TPLink routers. Support for Intel NUCs and the sensor network is work in progress. The legacy reservation system for sensor network is, and for the time being will be, available for users used to it. The general structure of TWIST AM is presented in Figure 1. It also shows its interactions with rest of the system.

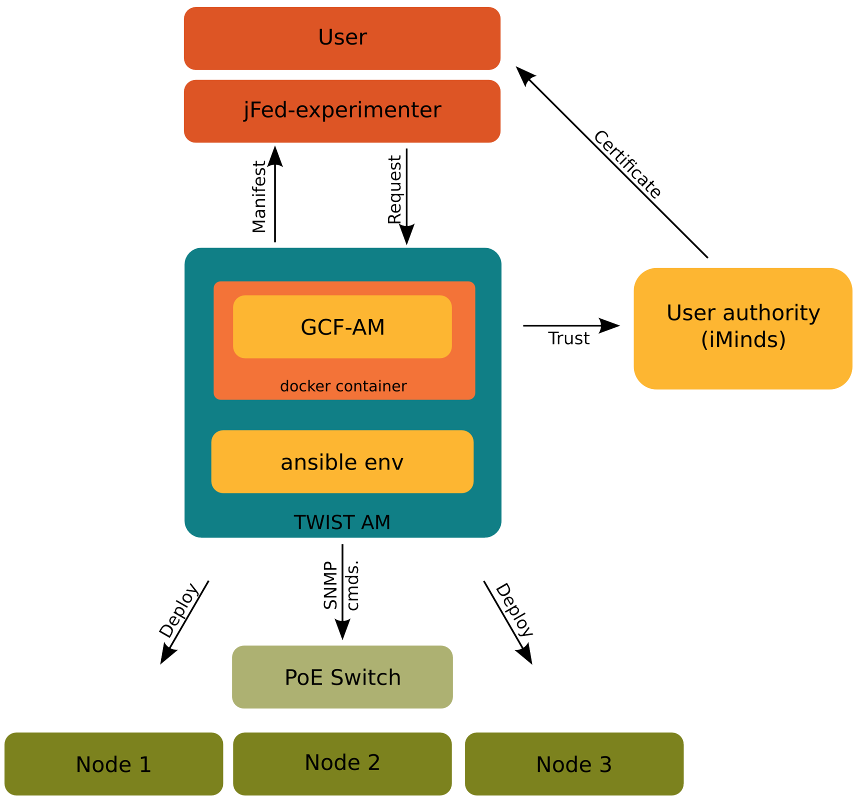


Figure 1 TWIST-AM structure

b. Base installation of the nodes

All TPLink nodes are installed with a basic OpenWRT image, that is not modifiable and only the AM has access to it. Dropbear and Python are installed providing SSH access and Ansible support. The USB Sticks, attached to every node, are used as mass storage during the provisioning process. All the nodes are connected to PoE switches, which can also be controlled using Ansible. This setup allows for programmatic rebooting regardless of the state of the running OS. After reboot the nodes will boot into the basic OpenWRT image.

c. Provisioning process

The provisioning process provides the experimenter with a fresh Linux installation (OpenWRT) with root access via SSH.

The provisioning workflow works as follows:

- SSH to protected testbed base system from twist-AM
- Download image from http server
- Extract image. Adjust settings and deploy experimenters public key
- Kexec new kernel specifying new rootfs
- After slice expiration: Cut power and reboot to protected system

This process is executed through Ansible providing workflow support, basic error resolution and parallel execution of tasks.

d. *OpenWRT images*

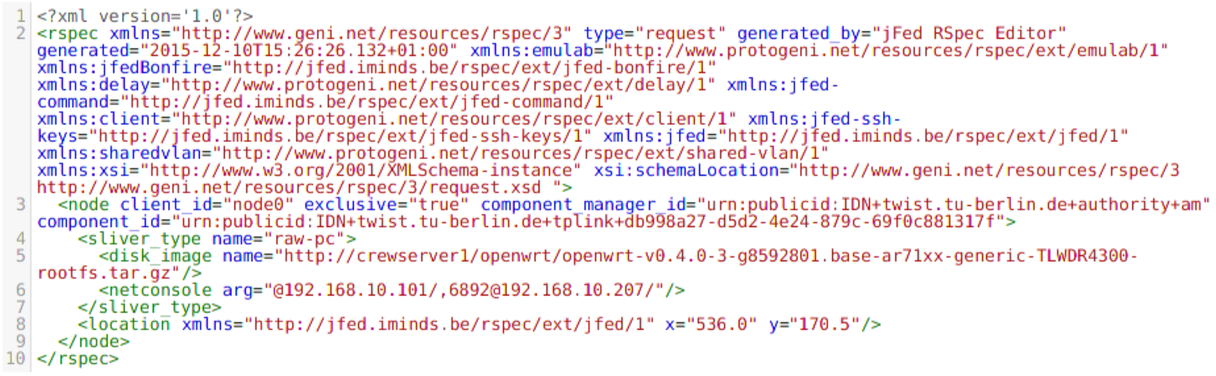
TUB has prepared a basic tool chain, that can be used to prepare own OpenWRT images that are compatible with the testbed. The changes to the stock OpenWRT image include kernel configuration for built-in USB and ext-fs drivers, default networking and file system structure (deployed on USB instead of flash). The toolchain is available on request to experimenters who want to create their own image.

e. *Features*

In addition to the default images, it is also possible to deploy custom images, which have to be provided by the experimenter via an HTTP server. The link can be passed as part of the Rspec. To be able to debug the boot process, the netconsole feature of the Linux kernel can be used to stream the kernel output to a user provided UDP client that needs to be specified in the Rspec.

A sample request Rspec can be found in Figure 2. In this case, the experimenter points to a custom image and want to stream the output to an UDP client running on some other host.

To ensure portability, twist-AM runs in a self-contained Docker container that can easily be deployed on a different host. As the configuration is not part of this container, it can be adapted to a different testbed with ease by editing a configuration file and providing an Ansible environment for the target testbed.



```
<?xml version='1.0'?>
<rspec xmlns="http://www.geni.net/resources/rspec/3" type="request" generated_by="jFed RSpec Editor"
generated="2015-12-10T15:26:26.132+01:00" xmlns:emulab="http://www.protogeni.net/resources/rspec/ext/emulab/1"
xmlns:jfedBonfire="http://jfed.iminds.be/rspec/ext/jfed-bonfire/1"
xmlns:delay="http://www.protogeni.net/resources/rspec/ext/delay/1" xmlns:jfed-
command="http://jfed.iminds.be/rspec/ext/jfed-command/1"
xmlns:client="http://www.protogeni.net/resources/rspec/ext/client/1" xmlns:jfed-ssh-
keys="http://jfed.iminds.be/rspec/ext/jfed-ssh-keys/1" xmlns:jfed="http://jfed.iminds.be/rspec/ext/jfed/1"
xmlns:sharedvlan="http://www.protogeni.net/resources/rspec/ext/shared-vlan/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.geni.net/resources/rspec/3
http://www.geni.net/resources/rspec/3/request.xsd ">
  <node client_id="node0" exclusive="true" component_manager_id="urn:publicid:IDN+twist.tu-berlin.de+authority+am"
component_id="urn:publicid:IDN+twist.tu-berlin.de+tplink+db998a27-d5d2-4e24-879c-69f0c881317f">
    <sliver_type name="raw-pc">
      <disk_image name="http://crewserver1/openwrt/openwrt-v0.4.0-3-g8592801.base-ar71xx-generic-TLWDR4300-
rootfs.tar.gz"/>
      <netconsole arg="@192.168.10.101/,6892@192.168.10.207/"/>
    </sliver_type>
    <location xmlns="http://jfed.iminds.be/rspec/ext/jfed/1" x="536.0" y="170.5"/>
  </node>
</rspec>
```

Figure 2 Sample request Rspec

2.6 Portable testbed

The portable testbed supports all three federation protocols: SFA, FRCP and OMSP. Emulab is used to implement the SFA AM. The client tools that can be used are respectively jFed, OMF6 and OML. The documentation of the portable testbed still needs to be extended to include all steps regarding the setup of the portable testbed. Usage of the client tools is described in chapter 3. Continuous facility monitoring is not foreseen due to the portable nature of the testbed; when moving the testbed, or installing it in a location without internet up-link, the testbed would always be marked as 'down'.

3 Deployment and extension of existing Fed4FIRE tools

Fed4FIRE tools make use of three federation protocols: SFA, FRCP and OMSP. These protocols were described in detail in D5.1 chapter 3. For every federation protocol, a testbed can chose to support one of the client tools that implement these protocols. The table below shows all WiSHFUL testbeds and lists the tools that are supported on the testbed.

	w-iLab.t (IMINDS)	**ORBIT (Rutgers)**	**FIBRE Island @UFRJ**	**IRIS (TCD)**	**TWIST (TUB)**	**Portable testbed**
SFA	jFed	XML/ jFed planned	MySlice / jFed planned	jFed planned	jFed	jFed
FRCP	OMF6	OMF6	OMF5.4	OMF6 planned	SSH/Custom	OMF6
OMSP	OML	OML	OML	OML planned	Custom measurements	OML

For SFA, most testbeds make use of the jFed tool. FRCP is supported on most testbeds by the use of OMF6. Nearly all testbeds provide OMSP support by using OML. For every testbed a simple example is shown for every tool. This document can serve as a guideline for experimenters.

Based on feedback from open call experimenters, other tools might be chosen or modifications will be made to the existing tools.

Example configuration files for the tools listed in this chapter (jFed, OMF6, OML) are available through the WirelessTestbedsAcademy GitHub account, in the ExperimentationTools repository [1].

3.1 w-iLab.t (IMINDS)

To run an experiment on the w-iLab.t testbed, the experimenter needs to register for an account at the iLab.t authority [2]. Once the account is approved, the experimenter can use his certificate to access the testbeds. The certificates issued by a Fed4FIRE authority give access to all testbeds in the federation. In addition to Fed4FIRE certificates, also users with a GENI certificate are allowed to use the w-iLab.t testbed.

The w-iLab.t offers an inventory tool [3] that can help the experimenter to determine the exact location of testbed nodes and their configuration. The tool also offers some filters to make it easier to search for suitable nodes for the experiment.

Nodes of the testbed can be reserved on the w-iLab.t reservation page [4]. The user has to import his certificate into the browser so that he can be authenticated. On the reservation page, the user can select a suitable time slot for his experiment and chose the nodes on which he wants the experiment to run.

Once the nodes have been reserved, the experimenter can use the following Fed4FIRE tools to conduct his experiment:

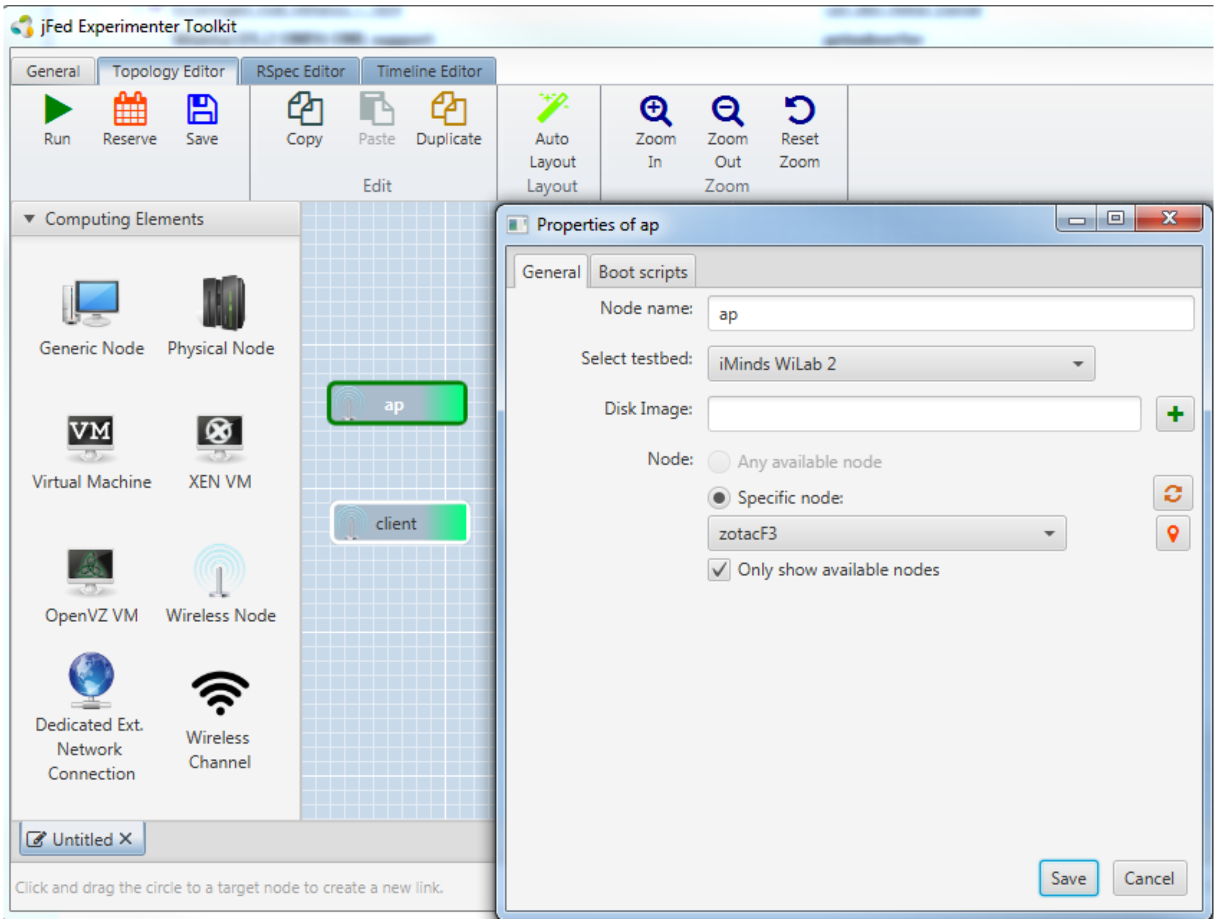
- Provisioning of the experiment is done using jFed (see section 3.1.1).
- Experiment control can be done using SSH or OMF6 (see section 3.1.2).
- Measurement collection is possible with OML (see section 3.1.3).

More extensive documentation can be found on the w-iLab.t documentation page [5].

3.1.1 Provisioning: jFed

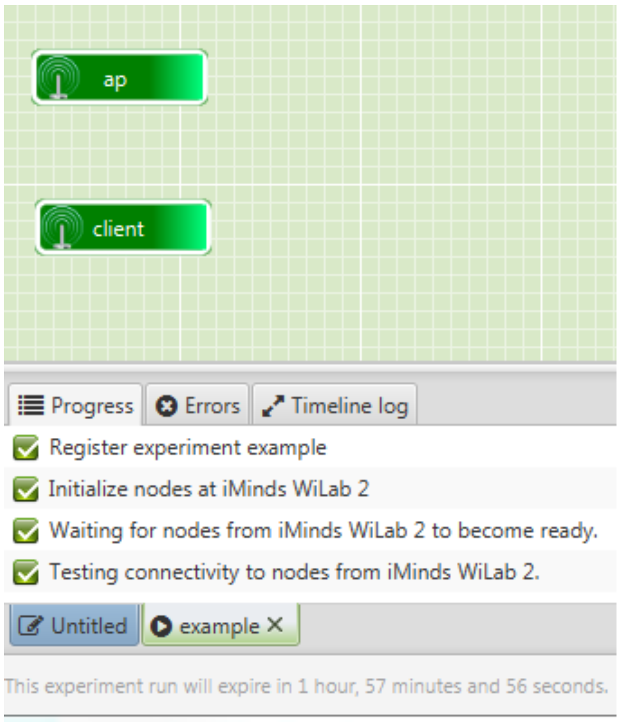
jFed can be used to provision the nodes. jFed can be started from the jFed website [6]. The following steps show how to use the w-iLab.t testbed from jFed:

- At start up, the user will be prompted to provide her user certificate, obtained while registering for an account.
- The user can start a new experiment by clicking *'New'*.
- Now the user must drag 2 wireless nodes on the canvas and right-click them to make additional configurations. The user should then name one of the nodes *'ap'* (for access point) and the other one *'client,'* for this example.
- The user should then select 'IMINDS WiLab 2' as testbed.
- Finally, the user selects her specific reserved nodes from the dropdown list. If the 'Disk Image' field is left blank, the nodes will be loaded with the default operating system provided by the testbed. The resulting configuration is shown in Figure 3.
- The user may click 'Run' to start her experiment and fill in a unique name for your slice.
- When all nodes turn green, the user's experiment is successfully activated, as shown in Figure 4.
- Simply double-clicking the nodes in jFed will open up a terminal that gives the user full SSH access to the nodes.



Node name: ap
Select testbed: iMinds WiLab 2
Disk Image:
Node: Any available node [✓ Specific node:] zotacF3
[✓] Only show available nodes

Figure 3 jFed experiment at w-iLab.t



Register experiment example
Initialize nodes at iMinds WiLab 2
Waiting for nodes from iMinds WiLab 2 to become ready.
Testing connectivity to nodes from iMinds WiLab 2.

This experiment run will expire in 1 hour, 57 minutes and 56 seconds.

Figure 4 jFed active experiment at w-iLab.t

jFed orchestrates the provisioning by contacting the SFA AM at w-iLab.t [7]. The XML code below shows the request Rspec for this experiment. It contains two nodes from the example above (ap and client) and the node id's of the specific testbed nodes on which the experiment should run (zotacF3 and zotacF4).

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rspec generated="2014-02-23T20:07:09.675+01:00" type="request">
<node client_id="ap"
       component_id="urn:publicid:IDN+wilab2.ilabt.IMINDS.be+node+zotacF3"
       component_manager_id="urn:publicid:IDN+wilab2.ilabt.IMINDS.be+authority+cm"
       exclusive="true">
              <sliver_type name="raw-pc"/>
  </node>
  <node client_id="client"
       component_id="urn:publicid:IDN+wilab2.ilabt.IMINDS.be+node+zotacF4"
       component_manager_id="urn:publicid:IDN+wilab2.ilabt.IMINDS.be+authority+cm"
       exclusive="true">
             <sliver_type name="raw-pc"/>
  </node>
</rspec>
```

3.1.2 Experiment Control: OMF6

Experiment control on w-iLab.t can either be done manually through SSH, or automatically by using OMF6. The OMF6 deployment at w-iLab.t is shown in Figure 5.

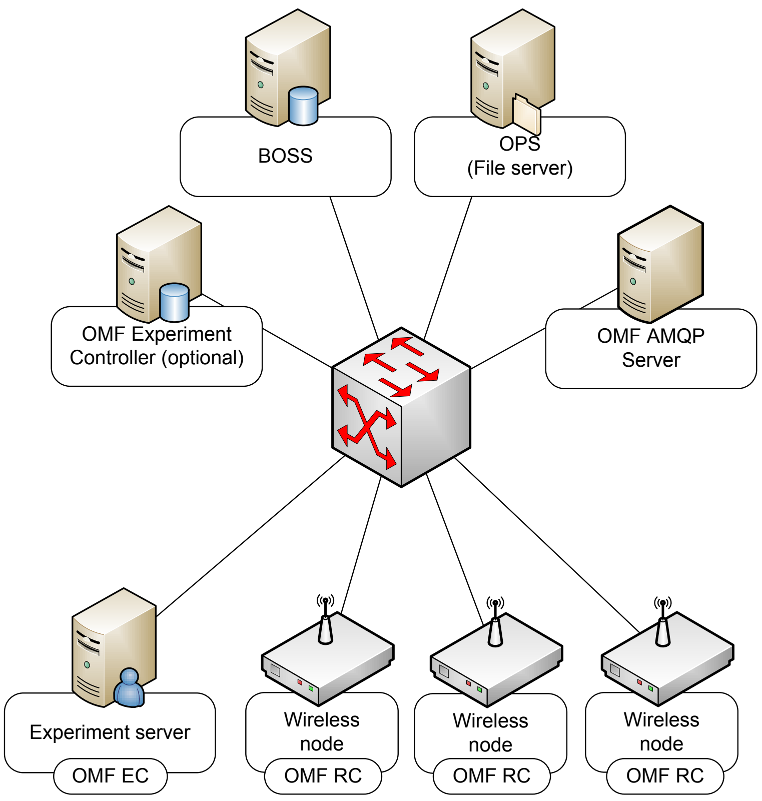


Figure 5 OMF6 deployment at w-iLab.t

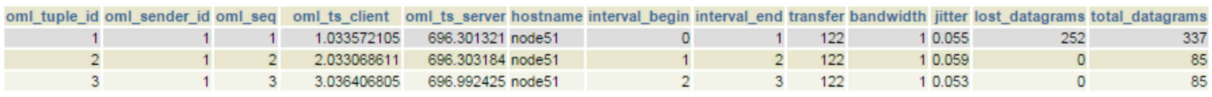
When using OMF6, the experimenter has to define all details of the experiment in an OEDL file. This file contains a listing of the resources, the applications and a time line for the experiment. The OEDL file is parsed by an OMF Experiment Controller (OMF EC) and sent to OMF Resource controllers (OMF RC) that are running on the testbed nodes. This communication is done using AMQP. The following setup can be used on w-iLab.t:

- The AMQP server is accessible at amqp://labwiki.atlantis.ugent.be
- A random w-iLab.t node can act as OMF EC. Install guides can be found on the w-iLab.t documentation page [5].
- The default operating system on w-iLab.t nodes already have an OMF RC pre installed. If you want to install one yourself, see the w-iLab.t documentation [5]. The OMF RC of a specific node can be addressed by using the hostname of that node. The hostnames of the nodes in a jFed experiment are composed as follows:
    - *<nodeName>.<sliceName>.<projectName>.<testbed>*
    - Example hostnames could be:
      ap.example.wall2-ilabt-IMINDS-be.wilab2.ilabt.IMINDS.be
      client.example.wall2-ilabt-IMINDS-be.wilab2.ilabt.IMINDS.be

The full OEDL example for this experiment can be found on the w-iLab.t documentation page [5].

3.1.3 Measurements: OML

To support OML, the w-iLab.t testbed offers one publicly available OML server [8]. All testbed nodes are pre-installed with the OML client library. The example OMF6 experiment using OML measurements can be found on the w-iLab.t documentation page [5]. A sample of the resulting OML database is shown in Figure 6.



oml_tuple_id	oml_sender_id	oml_seq	oml_ts_client	oml_ts_server	hostname	interval_begin	interval_end	transfer	bandwidth	jitter	lost_datagrams	total_datagrams
1	1	1	1.033572105	696.301321	node51	0	1	122	1	0.055	252	337
2	1	2	2.033068611	696.303184	node51	1	2	122	1	0.059	0	85
3	1	3	3.036406805	696.992425	node51	2	3	122	1	0.053	0	85

Figure 6 Sample OML database at w-iLab.t

3.2 ORBIT (Rutgers)

ORBIT facilities primary method of access relies on the use of public key authentication. Account creation is governed through a distributed approval system where the PI for the institution/group has full authority and is in charge of account management. Once the account is approved, an experimenter can use it to create reservations and access the testbed. The account system is fully federate with GENI enabling any/all registered users' access to all of the ORBIT resources with their GENI credentials.

Reservations for any of the 11 ORBIT domains can be made on the ORBIT Schedule page that is part of the Control Panel (user portal). The scheduler screen is illustrated in Figure 7**.** The reservation approval process is fully automated and is based on a two stage algorithm. In the first (pre-approval) stage, a scheduling request received before noon is pre-approved for the following day. Users are limited to two hours a day of pre-approved time on the main grid. For the reservations that are made less than twelve hours in advance or for the ones that are more than 2 hours in duration (i.e. multiple slots), the slots will be automatically approved at the beginning of the slot (second or just-in-time approval stage).

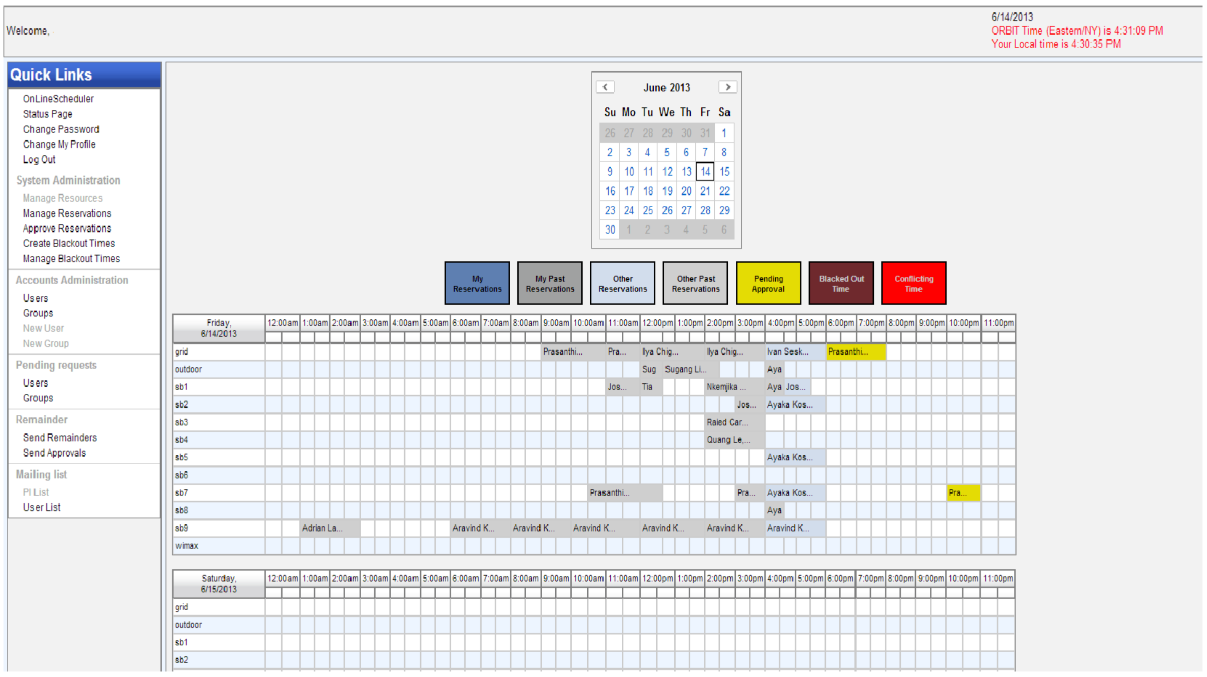


Figure 7 ORBIT Scheduler Page

The scheduler allows creation of conflicting requests (i.e. requests from multiple users for the same time slot) which are automatically resolved before the beginning of the slot based on past usage (with the observation window over the past two weeks).

Once domain reservation is approved, the experimenter can access corresponding console to conduct his experiment:

- Provisioning of the experiment is done using OMF5.5 (see section 3.3.1).
- Experiment control can be done using SSH or OMF5.5/6 (see section 3.2.2).
- Measurement collection is possible with OML (see section 3.2.3).

More extensive documentation can be found on the ORBIT wiki pages [9].

3.2.1 Provisioning: OMF5.5

The ORBIT testbed uses OMF5.5 for imaging (i.e. provisioning) a set of nodes. The actual sets can be defined either with command line options through OEDL based configuration files, can include a number of built-in topologies and/or use inventory database queries or corresponding OMF Inventory Service responses (XML). Users can load/save a large number of pre-existing (a.k.a. baseline) images as well as install custom images using OMF.

3.2.2 Experiment Control: OMF5.5/OMF6

The primary means of experiment control in ORBIT are through OMF. Both version 5.5 and 6.0 are supported on each of the 10 domain consoles (main grid, outdoor deployment and 9 sandboxes) that are used as experiment controllers (ECs). The experimenters using OMF for execution are required to use OEDL for experiment specification.

3.2.3 Measurements: OML

The measurement collection facilities in ORBIT are provided with two sets of OML collection servers: a.) each domain console acts as local OML server for experiments that are executed with nodes in a single domain and b.) a global OML server (oml.orbit-lab.org) that is connected to all domains and can be used for experiments spanning multiple domains or the ones that are using external resources.

3.3 FIBRE Island @ UFRJ

3.3.1 Provisioning: MySlice / jFed Planned

The MySlice [10] portal for the FIBRE testbed @ UFRJ can be accessed here: https://portal.ufrj.fibre.org.br/LS-WEB/. Users first have to sign up for an account, after which they can select resources from the testbed and reserve them for a given time period. Once resources have been provisioned, the users are able to SSH into the nodes.

The entire provisioning of the experiment is done using the MySlice portal.

The support for jFed is planned by June 2016.

3.3.2 Experiment Control: SSH / OMF5.4

Experiment control on this FIBRE Island is done manually through SSH, or automatically using OMF5.4. Users can define their entire experiment in an OEDL file to allow for repeatable uniform experimentation.

3.3.3 Measurements: OML

All testbed nodes are pre-installed with the OML client library. In addition, the testbed also offers an OML server to the experimenters.

3.4 IRIS (TCD)

The development of TCD facility access is currently on-going. Mechanisms for accessing the TCD testbed will largely follow the approach for accessing the w-iLab.t at IMINDS. Anticipated steps for an experiment with the TCD testbed include:

- Register for a F4F account with the iLab.t authority, as described above. The facility at TCD accepts valid F4F certificates.
- TCD will offer a web-based reservation tool to be released with SFA support.
- Provision of the experiment to be done with jFed (see section 3.4.1)
- Experiment control can be done with SSH currently or OMF6 (see section 3.4.2) in the future
- Measurement collection to be possible with OML (see section 3.4.3) in the future
-

3.4.1 Provisioning: jFed

As discussed in Section 3.1.1, jFed provides resource provisioning functionality in a drag and drop interface. TCD provides resources in the form of experimentation units, which consist of computational support and flexible radio hardware. Users will have the ability to select a particular radio node and computational disk image.

Once resources are provisioned, users will be able to double click an experimentation unit to gain SSH access.

3.4.2 Experiment Control: OMF6

OMF support is currently being designed for TCD’s facility. Experimentation units will have an OMF RC installed on their default computational image.

3.4.3 Measurements: OML

In order to support monitoring, TCD will offer access to an OML server and pre-install experimentation units with OML a client library.

3.5 TWIST (TUB)

In order to authenticate to the twist-AM the user needs to sign up for an account at the IMINDS authority. On the right side of the registration form, the user should select 'Join Existing Project' and enter 'TWIST' as project name. This will trigger a request to the projects lead and the user will need to be approved before his account is being enabled. After the user has received notice of his activation, navigate to the page again and login. The user now has the opportunity to download a certificate. The user should then select the first option to download the certificate to his local machine.

3.5.1 Provisioning: jFed

a. Downloading and initializing the jFed tool

There is a version of jFed-experimenter-GUI built for use with the TWIST testbed. Currently users must download the corresponding jar archive (still available on request), the TWIST testbed will be available in the standard jFed in future. After downloading, the user may start the GUI by executing java -jar jFed-experimenter-GUI.jar.

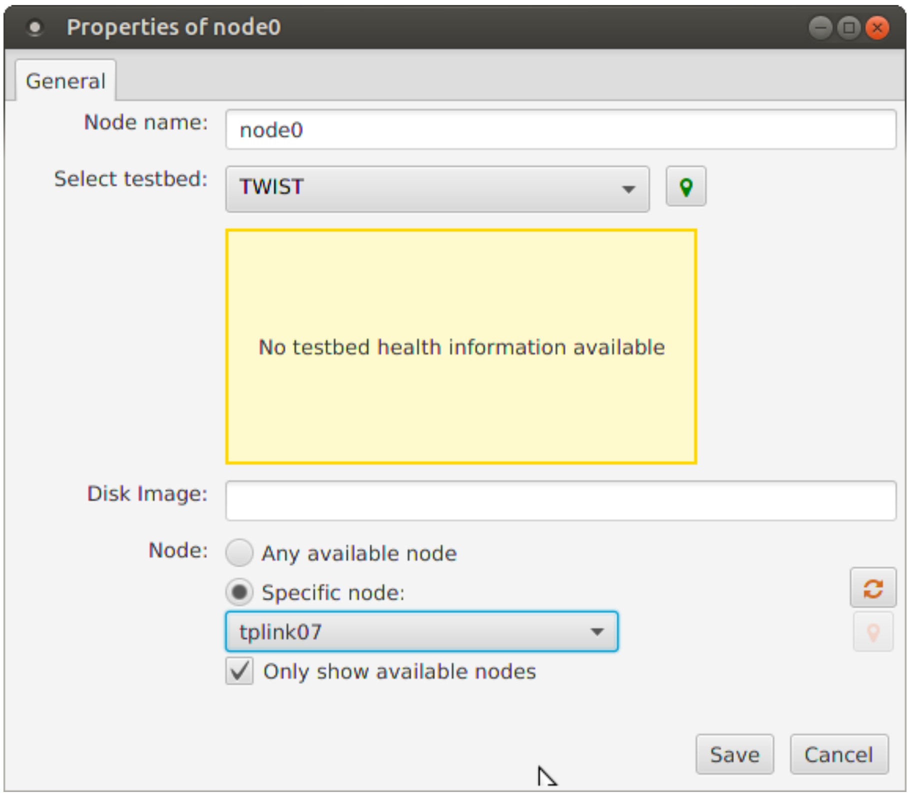
In the login dialog, the user should select the certificate downloaded earlier and enter his IMINDS password. After clicking on login the user will be asked to make initial settings. For automated upload of the public key to the reserved nodes, the user should add his key in the settings.

- Under 'SSH Authentication' the user should select 'Use custom key-pair' and browse for his private SSH key

After clicking on save, the GUI will appear and the user can start to setup his first experiment.

b. Creating a first experiment

When the user clicks New, he will get a blank canvas where he can draw your experiment. To do so, the user should drag in a Wireless node from the left side to the canvas.



Properties of node0

General

Node name: node0

Select testbed: TWIST

No testbed health information available

Disk Image:

Node: Any available node [✓ Specific node:] tplink07

✓ Only show available nodes

Figure 8: Specifying TWIST as node testbed

To select 'TWIST' as the testbed, the user should right click on the node and click 'Configure Node'.

- In the 'Properties' dialog, the user should select 'TWIST' as the testbed from the dropdown menu
- In the 'Node' selection dialog the user can either select a specific node or just let the AM assign any available

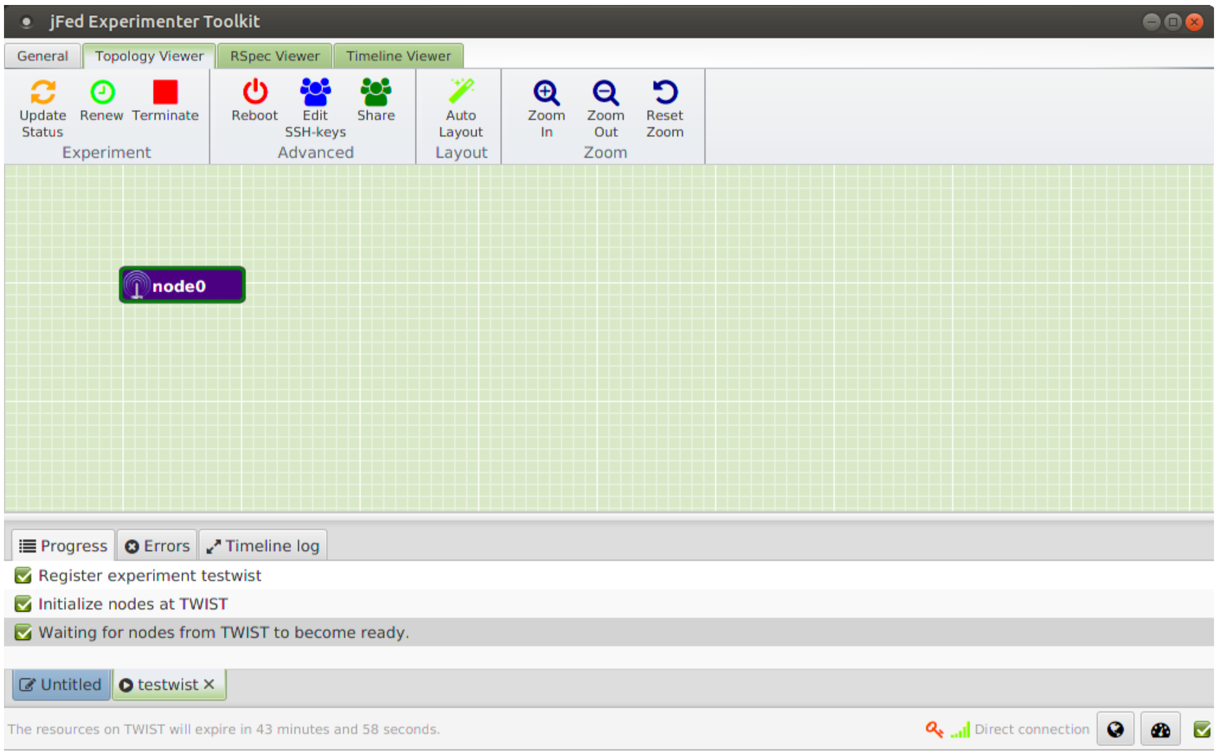
After clicking on save, the user can repeat this procedure and select as many nodes as his experiment requires. To avoid having to reassemble his setting every time, the user can always 'Save' your experiment as an Rspec file and reload it the next time he opens the jFed-tool.

c. Running the experiment

To run this experiment, the user begins by clicking the tab General at the top, and then the Run button. The user will now have to choose a name for the experiment (= slice name) and choose a maximum duration.

It will now take a couple of minutes to get the node prepared.

When a green check box in front of 'Waiting for nodes from TWIST to become ready' appears, the user can log in to the nodes as root using his private key.



✓ Register experiment testwist
✓ Initialize nodes at TWIST
✓ Waiting for nodes from TWIST to become ready.

The resources on TWIST will expire in 43 minutes and 58 seconds.

Figure 9: The experiment setup has finished

d. Releasing the nodes

When the user has finished his experiment, he should not forget to click the red 'Terminate' button to release his nodes. This way they are available for other experimenters again. However, the user must keep in mind that this will reboot the node and all changes he may have made are lost irreversibly. Also the user's nodes will expire together with his slice. The IMINDS authority will send the user a notification and he can renew your slice, if his experiment takes longer than expected though.

3.5.2 Experiment control: SSH / Custom

Currently, only experiment control over SSH is supported. The necessity of an automated experiment control framework, such as OMF6, will be investigated. The biggest concern is the ruby support on the routers (MIPS platform), which complicated integration. The OMF6 based solutions is for those routers suboptimal. Namely running RCs on the remote machine for each router, this creates additional execution delay and can be source of problems.

3.5.3 Measurements: Custom

Currently, only custom measurements are supported, set up by the experimenter himself. The necessity of an automated measurement collection framework, such as OML, will be investigated.

3.6 Portable testbed

To run an experiment on the portable testbed, the experimenter needs to register for an account at the iLab.t authority [11]. Once the account is approved, the experimenter can use his certificate to access the testbed. The certificates issued by a Fed4FIRE authority give access to all testbeds in the federation. In addition to Fed4FIRE certificates, also users with a GENI certificate are allowed to use the portable testbed.

The experimenter can use the following Fed4FIRE tools to conduct his experiment:

- Provisioning of the experiment is done using jFed (see section 3.6.1).
- Experiment control can be done using SSH or OMF6 (see section 3.6.2).
- Measurement collection is possible with OML (see section 3.6.3).

3.6.1 Provisioning: jFed

The Emulab framework takes care of resource discovery, reservation and provisioning. It allows experimenters to design complex topologies and transparently takes care of all network configurations. The Emulab framework also provides a GENI AMv3 SFA interface. The architecture of the portable testbed is shown in Figure 10.

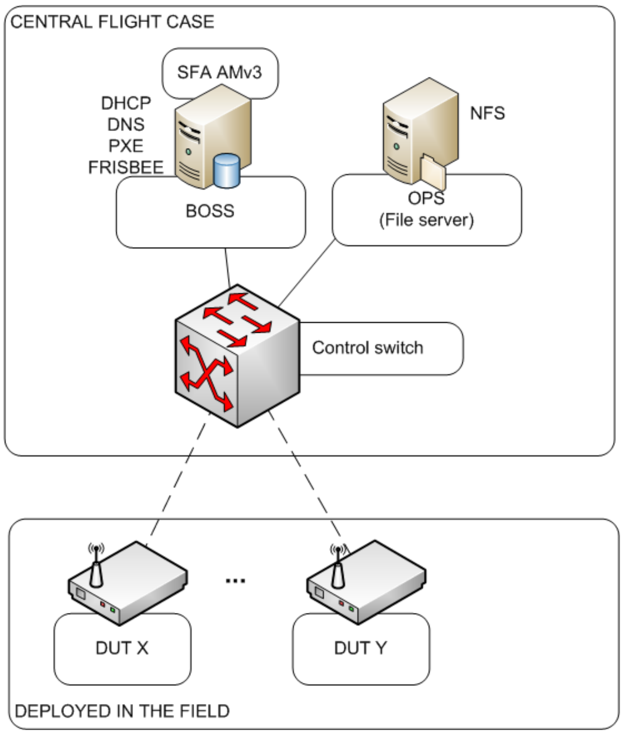


Figure 10 Portable testbed SFA architecture

The usage of jFed in the portable testbed is identical to the usage of the w-iLab.t testbed (see section 3.1.1). The experimenter just has to select 'Portable testbed' from the available list of wireless testbeds.

3.6.2 Experiment Control: OMF6

Experiment control on the portable testbed can either be done manually through SSH or automatically by using OMF6. The OMF6 deployment at w-iLab.t is shown in Figure 11.

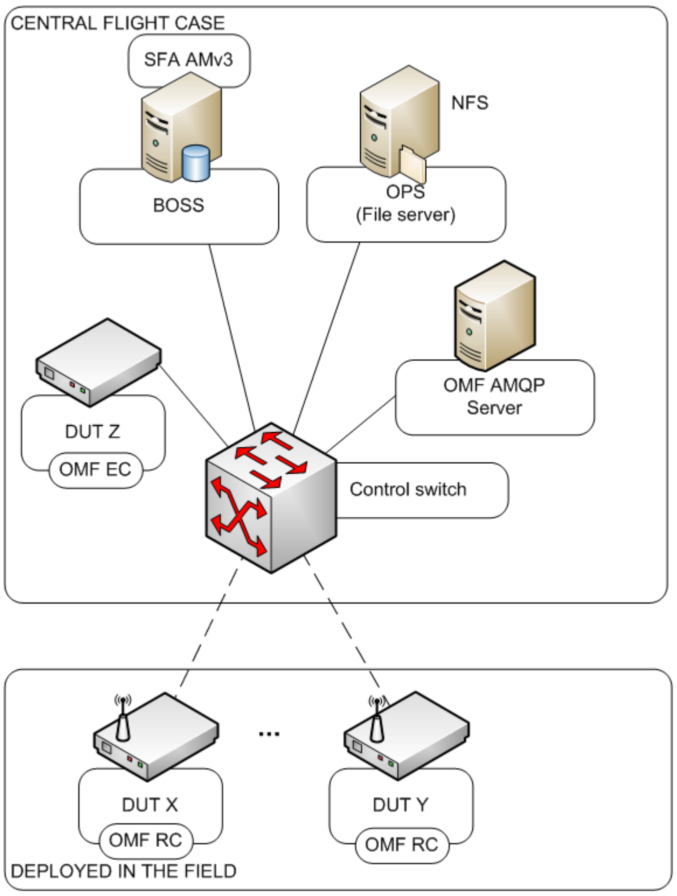


Figure 11 Portable testbed OMF6 installation

The following setup can be used in the portable testbed:

- The AMQP server is accessible at amqp://amqp.portable.test
- A random portable testbed node (DUT) can act as OMF EC. Install guides for the OMF EC can be found on the w-iLab.t documentation page [5].
- The default operating system on portable testbed nodes (DUT) already have an OMF RC pre installed. If the user wants to install one herself, she should see the w-iLab.t documentation [5]. The OMF RC of a specific node can be addressed by using the hostname of that node. The hostnames of the nodes in a jFed experiment are composed as follows:
    - <nodeName>.<sliceName>.<projectName>.<testbed>
    - Example hostnames could be:
    - ap.example.wall2-ilabt-IMINDS-be.portable.test
      client.example.wall2-ilabt-IMINDS-be.portable.test

3.6.3 Measurements: OML

To support OML, the portable testbed offers one pre-installed OML server [12]. All testbed nodes are pre-installed with the OML client library. The complete architecture of the portable testbed is shown in Figure 12.

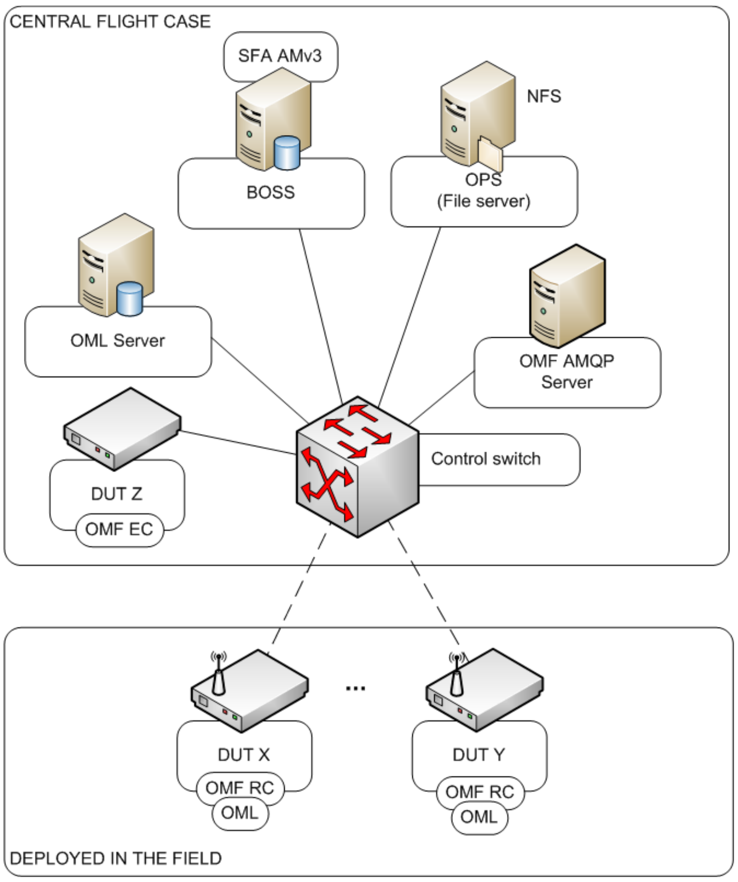


Figure 12 Portable testbed architecture

4 Development & Fed4FIRE compatibility of new tools

4.1 Tools for the integration of WiSHFUL UPIs in testbeds

The WiSHFUL UPIs developed in WP3 and WP4 will allow experimenters to implement control programs allowing fine-grained control of their solutions under test. The control programs can run on each node offering node-local control via UPI_R and UPI_N and management via the UPI_M. They can also enable network-wide or global control via UPI_G. This interface provides functions for discovering the nodes under control. Since this information is already available via the testbed management services, it makes sense to provide an interface between the testbed management services and the global monitoring and configuration engine implementing UPI_G. Figure 13 illustrates how this interface is integrated in the high-level WiSHFUL architecture.

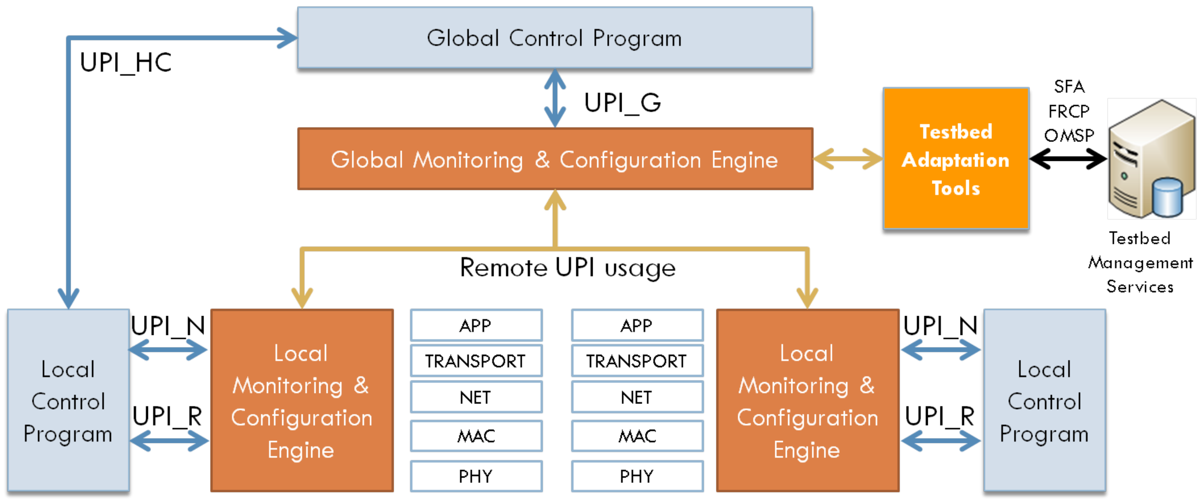


Figure 13 WiSHFUL UPIs in testbeds

Because all Fed4FIRE compliant testbeds have a specific implementation (e.g. Rspec extensions) for the Fed4FIRE interfaces (SFA, FRCP, OMSP), testbed adaptation tools need to be developed that offer a generic interface to the global monitoring and configuration interface.

The following abbreviations are used below:

- Global MCE: Global Monitoring & Configuration Engine
- Local MCE: Local Monitoring & Configuration Engine

The following requirements were indentified for successful integration of WiSHFUL UPIs in testbeds:

- Node discovery (see section 4.1.1): The Global MCE requires knowledge on the number and type of nodes that are part of an experiment. This information can be obtained from the testbeds' SFA AM.
- Experiment setup (see section 4.1.2): The installation of Global and Local MCE's on testbed nodes should be automated.
- Experiment control (see section 4.1.3): OMF6 (FRCP) allows the configuration and orchestration of an experiment. OMF6 can also be used to configure and start the Global and Local MCE's on testbed nodes.
- WiSHFUL control (see section 4.1.4): All tasks regarding WiSHFUL control should be cleanly separated from end-user applications (experiment control).

- Monitoring & measurements (see section 4.1.5): OML (OMSP) can be used for gathering monitoring results from each testbed node. This information can be fed via UPI_G to the Global MCE to steer the decision making.

The following sections will serve as a guideline for implementation during the second year of the project. The integration between testbeds and the application API described in D10.1 will follow the same approach as described below.

4.1.1 Node discovery

SFA can be used for resource discovery, e.g. knowing which nodes are in an experiment. Using the resource descriptions advertised by the testbeds AM (in Rspec format) the type of nodes and their capabilities can be retrieved. JFed can be used to select & provision the discovered resources (nodes) that are needed to conduct the experiment.

After activation of the experiment, the testbeds AM can be queried for the list of nodes (or other types of resources) that are part of a certain experiment. This information can be used by OMF6 to feed the list of experiment nodes to the global MCE at start-up. This is explained in detail in section 4.1.3.

It should however also be possible to do WiSHFUL experimentation on hardware that is not part of a testbed, without the usage of SFA and FRCP. Therefore a simple node discovery mechanism should also be implemented into the Global & Local MCE's.

4.1.2 Experiment Setup

Scripts can be configured in jFed so that on start-up extra software is installed on the nodes. If this software is complex to install and depends on lots of libraries, tools like Ansible could be used. This feature will be exploited to install the software components for the following control functionalities:

- Experiment Control (see section 4.1.3): An Ansible script will be provided to install an OMF Experiment Controller (on one node) and OMF Resource Controllers on all other nodes.
- WiSHFUL Control (see section 4.1.4): An Ansible script could be made to allow installation of WiSHFUL software onto testbed nodes. This might install the Global MCE on one node (could be on the same node as the OMF EC) and Local MCE on all other nodes.

4.1.3 Experiment Control

OMF6 is used to define the functionality of the nodes in the experiment and to construct a timeline (time and/or event based) of the experiment. In practice, this will result in giving the testbed resources a meaningful name and assigning applications to run on them. It is possible to retrieve the nodes in an experiment from SFA and use this information in the OMF experiment description.

The *getResources* command [13] retrieves the list of resources which are associated with the current slice used by the experimenter. The result of this command is an array where each element is a resource record (hash describing the resource). The experiment script can then filter the list of resources based on specific keys and values in such a record. This command will have to be extended to support testbeds running Emulab or testbeds running the GCF wrapper. An example of a resource record is shown below for a wireless node in the w-iLab.t testbed (zotacB1).

"client_id": "zotacB1",
"status": "ready",

```
"omf_id": "the_id_by_which_that_node_can_be_used_in_an_OEDL_script",
"sliver_id": "urn:publicid:IDN+wilab2.ilabt.IMINDS.be+sliver+1234",
"ssh_login": {
 "hostname": "node1.wilab2.ilabt.IMINDS.be",
 "port": "22"
},
"interfaces": {
 "zotacB1:if0": {
  "client_id": "zotacB1:if0",
  "sliver_id": "urn:publicid:IDN+wilab2.ilabt.IMINDS.be+sliver+1234",
  "mac_address": "01234567890a",
  "ip": [
   {
    "address": "10.10.1.1",
    "type": "ipv4"
   }
  ]
 }
},
"urn": "urn:publicid:IDN+wilab2.ilabt.IMINDS.be+sliver+1234",
"type": "node"
```

Once all experiment nodes have been retrieved, the following software can be started by OMF:

- Experimenter software: All end-user (experimenter) applications should be defined and started in the OMF experiment description.
- WiSHFUL software: Global & Local MCEs should be started on the testbed nodes.

By using this approach, a clean separation is made between experiment control and WiSHFUL control.

Note that the OMF functionality to set up experiment network interfaces (wired or wireless) will not be used, as this will be covered by the WiSHFUL software.

4.1.4 WiSHFUL Control

To the experimenter, the WiSHFUL software can be considered as extra functionality that can be included in his experiment to allow for more advanced control strategies.

The global and local control programs define the network/radio control logic. The Global MCE should be able to retrieve the nodes in the experiment. Since the Global MCE is also started by OMF, the following approach can be used:

- OMF retrieves the nodes in Rspec format from the testbeds AM (*getResources*) (see section 4.1.3).
- When starting the Global MCE, this list of nodes is passed from the OMF EC to the Global MCE as a command line argument (hash).
- Interaction between the EC and applications started by the EC (e.g. the Global MCE) are possible through the passing of STDIN commands at run-time.

An example OMF application definition for the WiSHFUL Global MCE is shown in the next section (see section 4.1.5).

4.1.5 Monitoring & measurements

The OMF experiment controller can define events based on measurement information stored by OML. By default, the OML clients send their data to an OML server, which stores the data into a database. The OMF EC can query this database and then act if the measurements exceed a certain threshold. However, the delay that is introduced by this method might be too high to quickly react to changing situations.

To solve this issue, the WiSHFUL Global MCE will serve as an OML proxy server. This way, it will be able to parse all data before passing it on to the real OML server. The Global MCE will not store the data in a database, but keep the last received measurements available in RAM (e.g. Python associative array). Using this solution, no modifications are needed to the OML framework. Instead, the WiSHFUL Global MCE will be extended to support parsing and forwarding of OML measurement streams.
Information about the type of measurements is defined in the OMF experiment description (*defMeasurement*). This information should be passed to the WiSHFUL controller at start-up.

An example OMF application definition for the WiSHFUL Global MCE:

```
defApplication('WiSHFUL_Controller') do |app|
        app.description = 'Application Definition for the WiSHFUL controller software'
        # Define the path to the binary executable for this application
         app.binary_path = '/usr/bin/wishful-controller.py'
         # Define the configurable command line parameters for this application
         app.defProperty('resources', 'List of WiSHFUL enabled resources', '-l', {:type => :string})
         app.defProperty('oml-server', 'IP:PORT of real OML server', '-o', {:type => :string})
         app.defProperty('measurements', 'Description of measurement streams to analyze', '-m', {:type =>
:string})
end
```

4.1.6 Summary

This section serves as a summary by listing all the steps in the experiment life cycle.

- The experimenter uses his F4F certificate to start jFed.
- The experimenter chooses a selection of resources from a certain testbed and activates his experiment using jFed.
- By using simple Ansible install scripts, the user enables both experiment control (OMF6) and WiSHFUL control on the nodes in his experiment. The install scripts are automatically executed by jFed at start of the experiment.
- The user logs into the OMF EC and starts an OEDL script that reflects the detailed configuration of the experiment:
    - The nodes of the experiment are retrieved by the getResources() call.
    - The resulting node list is used to map applications onto resources:
        - Start the Global MCE on node1 (pass the node list, OML server and measurement descriptions as arguments)
        - After X seconds, start the Local MCEs one by one (with 2 seconds interval) on the other nodes.
        - After Y seconds, start end-user applications (e.g. throughput measurements over Wi-Fi) on the other nodes.

- The user creates/modifies/activates the WiSHFUL Global and/or Local Control Programs. Interaction with the WiSHFUL control is possible through passing of STDIN parameters from OMF EC to Global MCE at run-time.
- OML monitoring/measurement streams originating from end-user applications or Local MCEs are parsed by the Global MCE before being forwarded to the OML server. The Global MCE can take decisions based on thresholds defined for these measurements.

4.2 Advanced tools for automation of testbed/software deployment

4.2.1 Ansible

Ansible is an open source and free to use IT automation engines that simplifies the installation process for complex software suites. It is able to deploy applications and configure entire systems based on a uniform description, called a playbook. More information on the Ansible tool was given in D5.1 section 4.3.1.

Ansible is currently used by TUB to install the OpenWRT routers in the TWIST testbed. While the GCF framework provides the northbound SFA interface, Ansible takes care of the southbound interface towards the testbed specific hardware. More detailed information can be found in section 2.5 .

Ansible can also be used by the experimenter to easily install extra software or tools onto testbed nodes. Ansible playbooks are provided to enable the OMF6 framework (OMF EC, RC and AMQP server) and the OML instrumentation tool (both OML server and client libraries). For installing jFed, no Ansible playbook is available because it is limited to installing Java [14].

Example configuration files for the tools listed in this chapter (jFed, OMF6, OML) are available through the WirelessTestbedsAcademy GitHub account, in the ExperimentationTools repository [15].

5 Conclusion

This deliverable shows that the majority of the testbeds in WiSHFUL (w-iLab.t, ORBIT, FIBRE Island @ UFRJ, TWIST and the Portable testbed) have reached Fed4FIRE compliance on the level of the AM API. This API enables the use of existing Fed4FIRE tools on WiSHFUL testbeds. It also enables the use of a single user certificate to access all testbeds in WiSHFUL and Fed4FIRE. The use of one user account in combination with uniform tools greatly simplifies the experimentation life cycle for experimenters.

For every testbed, an example experiment life cycle is illustrated, making use of existing Fed4FIRE tools. The jFed tool is supported by most testbeds and allows the user to very easily provision resources on different testbeds. For automated experiment control, the experimenters can make use of the OMF6 framework. Finally, to collect measurements in a uniform way, the OML instrumentation tool can be used. In order to simplify the installation procedures for these tools, the IT automation tool Ansible is used. Sample Ansible playbooks are uploaded to the WirelessTestbedsAcademy GitHub account for all proposed tools. This deliverable can be used as a reference for experimenters to guide them through the entire experimentation process on WiSHFUL testbeds.

To conclude this deliverable, a detailed description regarding the integration of WiSHFUL UPIs in testbeds is given. The requirements for integration were split up into different categories: node discovery, experiment setup, experiment control, WiSHFUL control and monitoring & measurements. The approach defines a loose coupling between the testbed interfaces and the WiSHFUL UPIs, while allowing for a clean separation between experiment control and WiSHFUL control. To the experimenter, the WiSHFUL software can be considered as extra functionality (on top of existing testbed functions) that can be included in his experiment to allow for more advanced control strategies. The detailed description in this document will serve as a guideline for implementation during the second year of the project.

6 References

[1] ExperimentationTools repository on the WirelessTestbedsAcademy Github account, available at https://github.com/WirelessTestbedsAcademy/ExperimentationTools, last access December 16th 2015

[2] iLab.t authority, available at https://authority.ilabt.IMINDS.be/getcert.php, last accessed December 10th 2015

[3] W-iLab.t inventory page, available at http://inventory.wilab2.ilabt.IMINDS.be/, last accessed December 10th 2015.

[4] The w-iLab.t reservation page, available at https://www.wilab2.ilabt.IMINDS.be:12369/reservation/sfareservation.php3, last accessed December 10th 2015

[5] w-iLab.t documentation website, available at http://doc.ilabt.IMINDS.be/ilabt-documentation/wilabfacility.html, last accessed December 10th 2015.

[6] jFed home page, available at http://jfed.IMINDS.be, last accessed December 10th 2015.

[7] W-iLab.t GENI AM v3, accessible through https://www.wilab2.ilabt.IMINDS.be:12369/protogeni/xmlrpc/am/3.0

[8] W-iLab.t OML server, accessible at tcp:am.wilab2.ilabt.IMINDS.be:3004

[9] ORBIT home page, accessible at http://www.orbit-lab.org, last accessed December 15th 2015

[10] MySlice home page, accessible at https://myslice.info/, last accessed December 22nd 2015

[11] iLab.t authority, available at https://authority.ilabt.IMINDS.be/getcert.php, last accessed December 10th 2015

[12] W-iLab.t OML server, accessible at tcp:am.wilab2.ilabt.IMINDS.be:3004

[13] OEDL documentation for OMF6, available at https://omf.mytestbed.net/projects/omf6/wiki/OEDLOMF6, last accessed December 11th 2015.

[14] Java home page, available at https://www.oracle.com/java/index.html, last accessed December 16th 2015.

[15] ExperimentationTools repository on the WirelessTestbedsAcademy Github account, available at https://github.com/WirelessTestbedsAcademy/ExperimentationTools, last access December 16th 2015