F-INTEROP

HORIZON 2020
Information and Communication Technologies
Integrating experiments and facilities in FIRE+

Complementary report to deliverable D2.7

Complementary conformance test enablers final iteration

Grant Agreement number:	**687884**
Project acronym:	**F-Interop**
Project title:	**FIRE+ online interoperability and performance test tools to support emerging technologies from research to standardization and market launch** **The standards and innovations accelerating tool**
Type of action:	**Research and Innovation Action (RIA)**
Project website address:	**www.finterop.eu/**
Due date of deliverable:	**M33**
Dissemination level:	**PU**

This deliverable has been written in the context of the Horizon 2020 European research project F-Interop, which is supported 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.

Co-funded by the European Union

Co-funded by the Swiss Confederation

Document properties

Responsible partner	INRIA
Author(s)/editor(s)	Federico Sismondi, Tengfei Chang, Thomas Watteyne
Version	1.0
Keywords	Interoperability Testing, Remote Testing, Online, Platform, Testing components, Test enablers

Abstract

This short report corresponds to the deliverable **D2.7 – Complementary conformance test enablers final iteration**. It provides complementary information to the deliverable D.2.7 which was delivered in the shape of a source code package along with a video demonstration of the interoperability tool and enablers. It describes the advancement of the final state of the F-Interop online remote interoperability framework and the key enablers needed.

Table of Contents

List of Acronyms

6TiSCH	IPv6 over the TSCH mode of IEEE 802.15.4e
CoAP	Constrained Application Protocol
EC	European Commission
ETSI	European Telecommunications Standards Institute
EU	European Union
GPS	Global Positioning System
GUI	Graphical User Interphase
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technologies
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Standards Organization
IT	Information Technology
MAC	Media Access Control
OS	Operating System
R&D	Research & Development
RR	Resources Repository
SME	Small Medium Enterprise
SO	Session Orchestrator
TAT	Test Analysis Tool
TED	Test Extended Description
TL	Task Leader
TT	Testing Tool
URL	Uniform Resource Locator
WP	Work Package
W3C	World Wide Web Consortium

1 Introduction

1.1 About F-Interop

F-Interop is a Horizon 2020 European Research project, which proposes to extend the European research infrastructure (FIRE+) with online and remote interoperability and performance test tools supporting emerging technologies from research to standardization and to market launch. The outcome will be a set of tools enabling:

- Standardization communities to save time and resources, to be more inclusive with partners who cannot afford travelling, and to accelerate standardization processes;
- SMEs and companies to develop standards-based interoperable products with a shorter time-to-market and significantly lowered engineering and financial overhead.

F-Interop intends to position FIRE+ as an accelerator for new standards and innovations.

1.2 Deliverable Objectives

1.2.1 Work package Objectives

- Research and develop the online remote interoperability test key enablers
- Develop the conformance test enablers
- Implement and fine tune the requested tools with a modular architecture for extensibility

1.2.2 Task Objectives

1.2.2.1 T2.1: Online interop test core enablers M1-M33 (Task Leader: Inria)

Work. The main objective of this task is to define and implement the components of the F-Interop online remote interoperability-testing framework. This includes: the cloud-based interoperability test script repository (as well as its management), the test servers and test suites automation, as well as the libraries, adapters, API and hardware interfaces, and its reporting capability. This task includes developing new methods simplifying online remote interoperability testing. This task also considers associated security and authentication issues.

Roles: Inria will lead the task, and will integrate contributions from ETSI, UL and EANTC.

Outcome: All components and key enablers needed for online remote interoperability testing.

1.2.2.2 T2.2: Complementary conformance test enablers M1-M33 (Task Leader: Inria)

Work. Conformance is a pre-requisite for interoperability. This task will provide the enablers for online remote conformance testing which complement the key enabled developed in task T2.1. We will develop new methods and/or adaptations of existing conformance testing tools to take into account the specific case of interacting online and remotely with the implementation under test (IUT). Examples include sniffers platforms, tools for measuring end-to-end latency (for example based on GPS synchronization), a protocol dissector.

Roles: Inria will lead the task, and will integrate contributions from ETSI, UL and EANTC.

Outcome: Additional components needed for online remote conformance testing. These components will complement the key enablers developed in task T2.1.

2 Deliverables

The deliverable corresponds to "**Complementary conformance test enablers final iteration**". The deliverable itself is in the shape of source code and a video demonstrating a 6TiSCH conformance test session in action using these enablers. The video presents the execution of a 6TiSCH test.

The following resources compose the deliverable:

1. video introducing the setup of the 6TiSCH demo: https://youtu.be/r5SsCXgc4UU

2. the source code of the deliverable which has been tagged in F-Interop gitlab as 'video_demo': https://gitlab.f-interop.eu/f-interop/6tisch_testing_tool/tree/video_demo

3 Final iteration of the 6TiSCH Testing Tool

This section presents the 6TiSCH Testing tool and the current state of implementation of its features.

3.1 The standard

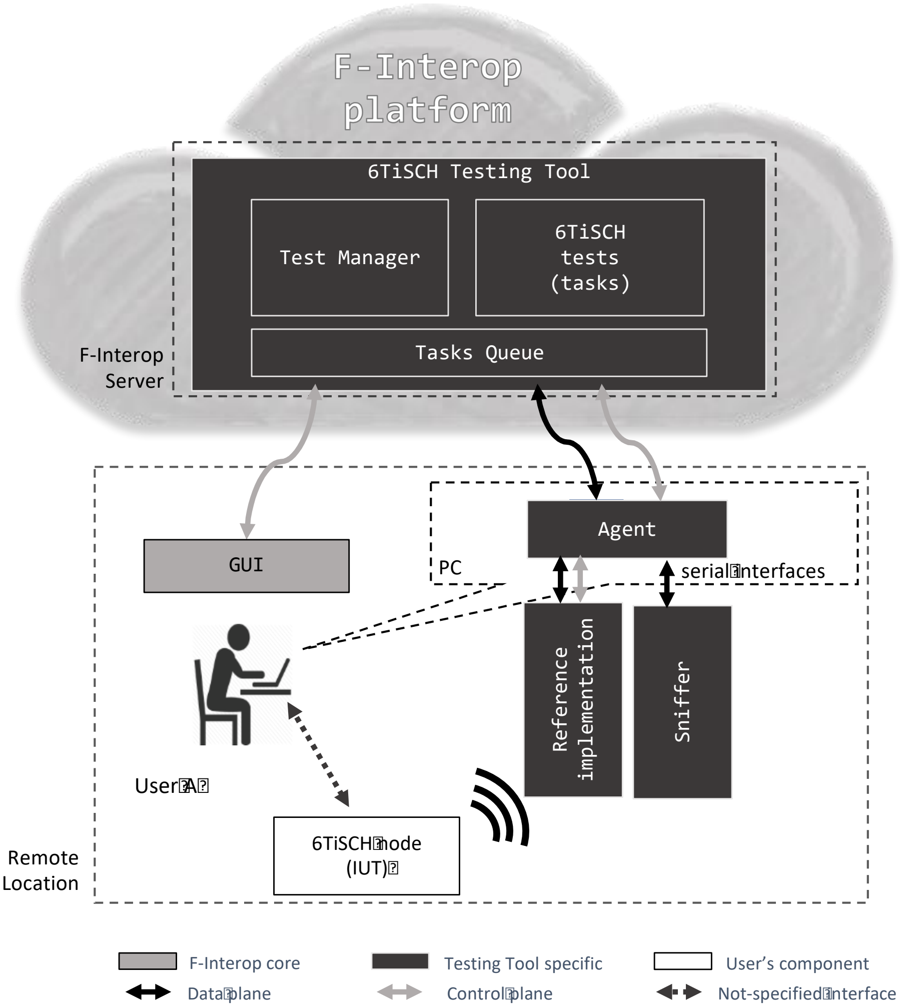
The 6TiSCH protocol stack, designed and standardized at IETF, glues together the IEEE802.15.4 PHY and MAC protocols with an IPv6- based upper stack. The latter includes the RPL routing protocol, the 6LoWPAN adaptation protocol, and the CoAP application protocol. Thanks to the Time Synchronized Channel Hopping (TSCH) scheme adopted at the MAC layer, 6TiSCH can offer low-latency, high reliability, scalability, and low power con- sumption, thereby meeting the requirements of Industrial Internet of Thing (iIoT) applications. 6TiSCH has defined the minimal configuration to run a 6TiSCH network (RFC8180), and a 6P sublayer that manages the allocation of MAC resources

3.2 The Test Description

The first 6TiSCH Interoperability TD was prepared for the first 6TiSCH Plugtest organised by ETSI in 2015, at IETF93 [3]. Since then, the TD has been updated several times, for each new ETSI Plugtest and Interop event organised in the framework of the F-Interop project. The latest version of the TD is available online 10 and includes four classes of tests: SYN tests (testing the synchronization status of device), MINIMAL tests (testing the minimal configuration of 6TiSCH), 6LoRH tests (testing IP packets in 6LoRH format, according to RFC8138), and 6P tests (testing format and behaviors of the 6P sublayer).

3.3 Implementation

The 6TiSCH testing tool is composed of three main protocol- specific components:



1. Test Manager: It is the brain of the 6TiSCH testing tool. To start the 6TiSCH manager, the user needs to select the 6TiSCH suite on the web GUI and start a session. Once it is running, the manger sends a message to the GUI asking which specific 6TiSCH test the user would like to execute. An AMQP consumer is up and running and listens to the choice made by the user. Once this information is received, the manager creates a test context which includes the sequence of steps required by that specific test, according to TD.

2. 6TiSCH Tests (tasks): It is the component dispatching STIMULI actions to users (start/shut-down IUT), controls to the packet sniffer (start/stop), and handling CHECK steps analysis after the test execution, using the Wireshark dissector.

3. Task Queue: It is in charge of executing the tasks generated by the manager and/or the 6TiSCH Tests (tasks) component. Once a task is received, the task queue assigns it to one of its workers that is currently idle and marks it as busy. Hence multiple tasks can be assigned at the same time using several workers. Once the task is completed, then the worker can get back to the idle status. The Task Queue is implemented in Celery.

3.4 Integration

Previous figure shows the general setup for running a 6TiSCH conformance test using the F-Interop Platform.

While running a test, the Manager generates three types of tasks for the Task Queue: step start, step end, testcase verdict. For all the tests, besides the specific feature for which interoperability is checked, the user has to perform some generic steps: turn on all the devices (the IUTs, and the packet sniffer); wait until the devices enter into a stable state; issue a stimulus to let the device perform a given action. As final step, to check the result of the test, the user uploads the Wireshark packet capture (pcap) file, which is analyzed by the Wireshark dissector. If the packet is found in the pcap file, and all the check fields are correctly dissected, then the manager gives a PASSED verdict. If the packet cannot be found or dissected, the test fails. The specific reasons why the test failed are shown inside the red bottom verdict.

3.5 Last fine-tuning actions

In future developments of the 6TiSCH tools, an Agent component will be integrated to interact with the IUT and automatize some steps of the test session. This includes the initial execution of the stimulus (e.g. send a 6P ADD packet), and the final upload of the pcap file. Currently, these operations are executed manually by the end user.

4 Demo

4.1 Goal of the demo

The goal of the demo is to show that the 6TiSCH Testing Tool can run a conformity test. For this demo, we will target the TD_6TiSCH_6P_01 test. We want to demonstrate in this demo that the core feature of the testing tool are all working reliably. Those features are:

- Packet sniffing
- Dissection of the captured frame
- Assertion upon those captured information
- Verdict emission

4.2 Set-up of the demo

The IUT used in this demo are OpenMote boards with the latest version of OpenWSN. OpenWSN is the reference implementation of 6TiSCH and it fully supports the 6P protocol that we intend to test.

4.3 Outcomes of the demo

When the demo is over, the user has witnessed a verdict (PASS/FAIL). The logs and traces (PCAP files) of its test session are also available for the user.

5 Annex

5.1 An example of 6TiSCH Test Description (YAML file)

```
TD_6TiSCH_6P_01:
  obj: check a 6P COUNT command is successful
  cfg:
    - topology => a DAGroot and a 6N within range (single hop)
    - all devices configured to communicate on a single frequency (frequency to be
allocated during the event)
    - link-layer security disabled
  ref:
    - IEEE802.15.4-2015
    - RFC8180
    - draft-ietf-6tisch-6top-protocol-05
  pre:
    - DAGroot and 6N are turned off
  seq:
    - s:
      - turn on the DAGroot
      - turn on the 6N
    - c:
      - the 6N synchronizes to the DAGroot

    - s:
      - have the 6N send a 6P ADD request to the DAGroot
        - asking for a single TX cells from 6N to DAGroot
        - with the candidate celllist set to [(4,5)]
      - check that the format of the request comply with draft-ietf-6tisch-6top-protocol-05
      - how this is done is implementation-specific
    - c:
      - the 6P transaction is succesful and both the DAGroot and 6N have added a cell from
6N to DAGroot at slotoffset=4 and channeloffset=5
      - check that the format of the reply comply with draft-ietf-6tisch-6top-protocol-05

    - s:
      - have the 6N send a 6P COUNT request to the DAGroot
      - check that the format of the request comply with draft-ietf-6tisch-6top-protocol-05
      - how this is done is implementation-specific
    - c:
      - check that, in the 6P response, the returned code SUCCESS
      - check that, in the 6P response, the count value is 1
      - check that the format of the reply comply with draft-ietf-6tisch-6top-protocol-05
```