
**HORIZON 2020**

**Information and Communication Technologies**

**Integrating experiments and facilities in FIRE+**

# Deliverable D2.8

## Testing tools instantiations (Final report)

| | |
|---|---|
| **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**<br>**The standards and innovations accelerating tool** |
| **Type of action:** | **Research and Innovation Action (RIA)** |
| **Project website address:** | **www.finterop.eu** |
| **Due date of deliverable:** | **31/08/2018** |
| **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.*


## Document properties

| | |
|---|---|
| **Responsible partner** | ETSI |
| **Author(s)/editor(s)** | Ghada Gharbi, Federico Sismondi, Tangfei Chang |
| **Version** | 1 |
| **Keywords** | Interoperability Testing, Conformance Testing, Remote Testing, Online, Platform, Testing components, Test enablers |

## Abstract

This report corresponds to the deliverable: D2.8 – Testing tools instantiations – final report.

The deliverable D2.8 is the final report of the F-Interop testing tools instantiations. It is a stable version of how F-interop testing tools enablers described in deliverables D2.1 and D2.2 are instantiated to perform interoperability and conformance testing of CoAP and 6TiSCH protocols, and oneM2M standard for M2M communications. It describes the realized session models.

# Table of Contents

# List of Figures

# List of Acronyms

| | |
|---|---|
| ABC | Attribute Based Credential |
| CA | Consortium Agreement |
| CoAP | Constrained Application Protocol |
| ComSoc | Communications Society |
| DESCA | Development of a Simplified Consortium Agreement |
| DHCP | Dynamic Host Configuration Protocol |
| DHT | Distributed Hash Tables |
| DNS | Domain Name System |
| DNSSec | Domain Name System Security Extensions |
| DPA | Data Protection Authorities |
| DPO | Data Protection Officer |
| EC | European Commission |
| ENISA | European Union Agency for Network and Information Security |
| ETSI | European Telecommunications Standards Institute |
| EU | European Union |
| FP7 | Seventh Framework Programme |
| GA | Grand Agreement |
| GA | General Assembly |
| GPS | Global Positioning System |
| HTTPS | Hypertext Transfer Protocol Secure |
| ICT | Information and Communication Technologies |
| ID | Identifier |
| IEC | International Electrotechnical Commission |
| IEEE | Institute of Electrical and Electronics Engineers |
| IERC | European Research Cluster on the Internet of Things |
| IETF | Internet Engineering Task Force |
| IoT | Internet of Things |
| IP | Internet Protocol |
| IPC | Intellectual Property Committee |
| IPM | IPR Monitoring and Exploitation Manager |
| IPR | Intellectual Property Rights |
| IPSEC | Internet Protocol Security |
| IPv4 | Internet Protocol version 4 |
| IPv6 | Internet Protocol version 6 |
| ISO | International Standards Organization |
| ISP | Internet Service Provider |
| IT | Information Technology |
| ITU | International Telecommunication Union |
| KPI | Key Performance Indicator |
| LSPI | Legal, Security and Privacy Issues |
| MAC | Media Access Control |
| MSc | Master of Science |
| M2M | Machine to Machine |
| OASIS | Organization for the Advancement of Structured Information Standards |
| OECD | Organization for Economic Cooperation and Development |

| | |
|---|---|
| OS | Operating System |
| OSN | Online Social Network |
| PC | Project Coordinator |
| PCP | Partner Contact Person |
| PDPO | Personal Data Protection Officer |
| PERT | Program Evaluation Review Technique |
| PhD | Doctor of Philosophy |
| PM | Person Month |
| PMB | Project Management Board |
| PPR | Periodic Progress Report |
| PRAAT | Privacy Risk Area Assessment Tool |
| P&T | Post & Telecom |
| QoS | Quality of Service |
| RAND | Reasonable and Non Discriminatory |
| RFC | Request For Comments |
| R&D | Research & Development |
| SME | Small Medium Enterprise |
| SMS | Short Message Service |
| SOTA (or SoA) | State Of the Art |
| SSL | Secure Sockets Layer |
| TC | Technical Coordinator |
| TCP | Transmission Control Protocol |
| TL | Task Leader |
| TLS | Transport Layer Security |
| Tor | The Onion Router |
| TRL | Technology Readiness Level |
| UK | United Kingdoms |
| UN | United Nations |
| UNCTAD | United Nations Conference on Trade and Development |
| UPRAAT | Universal Privacy Risk Area Assessment Tool |
| URL | Uniform Resource Locator |
| US | United States |
| VoIP | Voice over Internet Protocol |
| WES | Women's Engineering Society |
| WiTEC | Women in science, Engineering and Technology |
| WoT | Web of Trust |
| WP | Work Package |
| WPL | Work Package Leader |
| W3C | World Wide Web Consortium |
| XML | Extensible Markup Language |

# 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 T2.3: Testing tools instantiations M7-M34 (Task Leader: ETSI)

The purpose of this task is to verify the applicability of the tools developed in T2.1 and T2.2 to real standards. We therefore have identified the following emerging standards which we believe are of particular importance for the IoT: 6TiSCH, 6LoWPAN, CoAP, IPv6, oneM2M, Web of Things. For each of them, we will instantiate generic concepts developed in tasks T2.1 and T2.2. For each, we will comply with the following 2-step approach:

1. Develop the test specifications and conduct remote on-line testing in simple scenarios with a limited number of communicating devices.
2. Extend the testing to large scale remote on-line testing and experimentations with complex scenarios. This will include adding the relevant test cases.

**Roles:** ETSI will lead the task, and will integrate contributions from Inria, UL and EANTC. **Outcomes:** remote online testing tools and test suites for 6TiSCH, 6LoWPAN, CoAP, IPv6, oneM2M, Web of Things.

### 1.2.3 Deliverable Objective and methodology

#### 1.2.3.1 Deliverable objective

The objective of the **D2.8 Testing tools instantiations – final report** is the description of stable F-Interop testing tools instantiations as output of the task T2.3. This report illustrates the instantiations of the F-Interop online remote interoperability-testing framework components while considering test suites for emerging IoT standards such as 6TiSCH, CoAP and oneM2M.

#### 1.2.3.2 Deliverable methodology

The deliverable extends the information presented in Deliverable D2.3 Testing tools instantiations (initial report), where a first iteration of testing tools instantiations has been presented. Deliverable D2.8 focus on the new achievements in CoAP, 6TiSCH and oneM2M testing tools since the first iteration.

6TiSCH and oneM2M testing tools were at early stage. Now, they reached a beta version.

# 2 F-Interop platform enablers

We describe in this section F-interop core enablers which are:

(i) the event bus responsible of exchanging messages between F-Interop platform,
(ii) session orchestrator which plays an administrative role,
(iii) resource repository containing a list of all devices and their properties that can be used for F-interop tests,
(iv) result repository to store test results,
(v) User Interface (UI) the first point of contact for an end-user, and
(vi) agent running near devices (e.g. at the user premises) to communicate with the orchestration and analysis components. The interaction between components is based on the AMQP message protocol and an API was elaborated to define AMQP message formats.

Through standard security mechanisms, the F-Interop platform ensures the authentication of different users, and the confidentiality of test results.

## 2.1 The “Event Bus” software Design Pattern

The F-Interop architecture, show in Figure 1, is composed of different components. The components of this architecture are responsible of managing the testing infrastructure, including provisioning the underlying network, capturing trace, starting/stopping the different tests, and reporting the verdicts. Through standard security mechanisms, the architecture ensures the authentication of the different users, and the confidentiality of test results. F-Interop components exchange messages through an “Event Bus”. All communication is done through this mechanism, including control messages, raw data packets and logs. We use RabbitMQ as the underlying message-passing mechanism. It acts as a secure message broker between all the components through encrypted channels.

This architecture enables many independent components to talk to each other and avoid a monolith that would be difficult to manage as the number of components increase. Each message contains a routing key which indicates how to route this message to the relevant input queues of the components. It also shipped with a web interface for monitoring the load and enable easy debugging. Isolation of test session is performed using virtual hosts. The exchanged messages and the routing keys are standardized by F-Interop, following an API.

The messaging patterns is used by:

- Components which can publish events (the component is a producer): commonly used for sending debug logs, send data to other IUTs or controlling the execution of a test
- Components can subscribe to events: (the component is a consumer): e.g. UI reads test informative messages and prints them in the graphical interface.
- Components can provide or call services (components are consumer/producer at the same time for this type of interaction): we basically have two types request/replies or action/confirmation and they consume certain messages, and reply by producing others as a result.

The communication between F-Interop testing enablers is detailed at http://doc.f-interop.eu/.

Each message contains a routing key and a topic which indicates how to route this message to the relevant input queues of the components. Messages are of two types: ***control plane*** and ***data plane***. The ***control plane*** messages relate to the management of an ongoing test session: e.g. start a sniffer, signal the start/end of a test case, etc. The ***data plane*** messages contain the raw data exchanged between the IUTs.

We notice that in the case of performance testing, only control traffic is managed. In this deliverable, only interoperability and conformance testing tools are considered.

An F-Interop-User conducts remote interoperability tests in independent sessions. We use the virtual host mechanism of RabbitMQ to ensure isolation between concurrent sessions. This architecture is modular and scalable by design. Components can be added/deleted from the event bus without requiring further coordination. Different components can be run on different (virtual) machines to ensure scalability. Different components can be written in different programming languages.

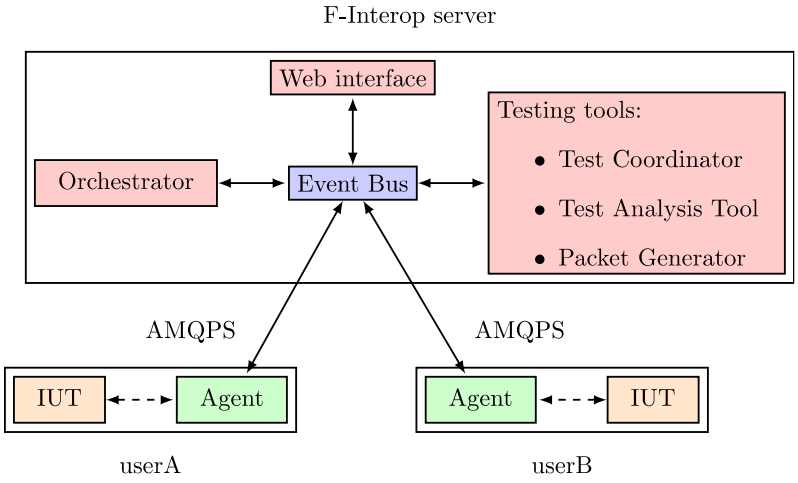


**Figure 1: F-Interop remote interoperability testing architecture**

# 2.2 Components description

## 2.2.1 Agent: connecting F-Interop users to the F-Interop Platform

An **agent** is a program that an F-Interop-User downloads from the F-Interop website, and which allows him/her to connect an IUT to the F-Interop server. Communication between the agent and the server is authenticated and secure. Through the agent, the F-Interop server can (remotely) interact with the IUT, for example by changing configuration or injecting packets. Similarly, the agent reports events to the server, such as sniffed packets.

## 2.2.2 The orchestrator

The orchestrator plays an administrative role. The orchestrator main roles are:

- monitoring the users that are connected,
- activating the rooms currently in use and starts/stops the test sessions. It is also in charge of provisioning the message broker and updating Technical Overview of F-Interop firewall rules when test sessions are activated. It does so by spawning/killing the processes of the different components connected to the event bus. It uses the supervisor process control system.
- publishing signalisation events to the event bus:
    - configuration message of the session to the event bus on session start,
    - terminate message on session close.

The test orchestrator can be run as a central service for a specific F-Interop test (think a user just running a test), or can be deployed on a testbed virtual machine (think a user wanting to adapt a test or develop a new test).

The orchestrator relies on RESTful API to communicate with F-Interop components. Possible operations are:

- Creating a new session,
- Starting a session,
- Stopping a session,

- Deleting a session,
- Creating a new user,
- Getting information about a user,
- Deleting a user,
- Posting an event on the event bus.

The orchestrator exposed RESTful API is detailed at http://doc.f-interop.eu/#session-orchestrator.

### 2.2.3 Testing tool

The **testing tool** handles the components throughout an F-Interop session. Each testing tool test either a particular protocol (e.g. CoAP Testing Tool) or a set of protocols used in a certain architecture (OneM2M Testing Tool). It can be started once the different users are connected and the necessary components are provisioned by the orchestrator. The role of the testing tool is to perform the tests between IUT(s) and generate a verdict for each test case. It operates as a black box which emits coordination messages towards agents (accurately in case of interoperability and conformance testing), and generates test verdicts and a final report. For interoperability and conformance testing, F-Interop provides a reference implementation for testing tools and documents the format of the messages exchanged between components (see doc.f-interop.eu). This description of messages exchanged between internal components of the testing tool can be seen as a second level (or extension) of the previously mentioned API specification. The purpose of this is to ease development and integration of new testing tools in F-Interop environment. This reference testing tool architecture and implementation includes components such as a test coordinator, a test analysis tool, a sniffer, a packet router (which intentionally can drop packets for simulating lossy links) and a packet generator (mainly for conformance testing). You can find more details in deliverable **D1.3_v1_0: Initial architecture design** and in doc.f-interop.eu. This reference testing tool architecture is the one used for CoAP interoperability testing tool, and OneM2M interoperability testing tool.

### 2.2.4 GUI

The F-Interop **Web interface** allows the user to select a test description from a list of available tests, start the execution of the test description and follow the execution of the different test cases. In some cases, the web interface can request the user to take some action (e.g. switch off a node). The web interface also allows the user to retrieve the test report. The web interface communicates with the rest of the system by sending/receiving message over the Event Bus.

# 3 Status of work

## 3.1 Status of work for T2.3 – Deliverable D2.8

The work accomplished for the first iteration of the task T2.3 - Testing tools instantiations - is to verify the applicability of the tools developed in T2.1 and T2.2 to real standards. Following the F-Interop architecture, the defined interfaces and the messaging API, interoperability, conformance, and performance testing tools are developed by the F-Interop partners. This document focuses on work done in interoperability and conformance testing. Emerging IoT Standards such as 6TiSCH, CoAP, and oneM2M were studied to illustrate the instantiations of the F-Interop remote testing framework components.

CoAP testing tool had reached its beta version. It is used as the reference implementation for testing tool testing protocols which sit top of IP. Its architecture and protocol agnostic implementation is documented and provided for other partners and contributors in the form of an enabler for interoperability testing.
Integration of CoAP testing tool to the other F-Interop enablers and ecosystem (UI, resource repo, results repo, orchestrator, etc) has been achieved.

Details about 6TiSCH, CoAP and oneM2M testing tools are given in the following sections.

## 3.2 CoAP interoperability testing tool

### 3.2.1 Overview

The CoAP Testing Tool offers to end users a controlled and feature-rich environment that eases the execution of the online and remote standard-based interoperability test procedure.

The following figure shows the setup for running a CoAP remote interoperability test between two IUTs. Two protocol-specific components (the Agent and the Testing Tool) have been implemented for supporting CoAP tests.

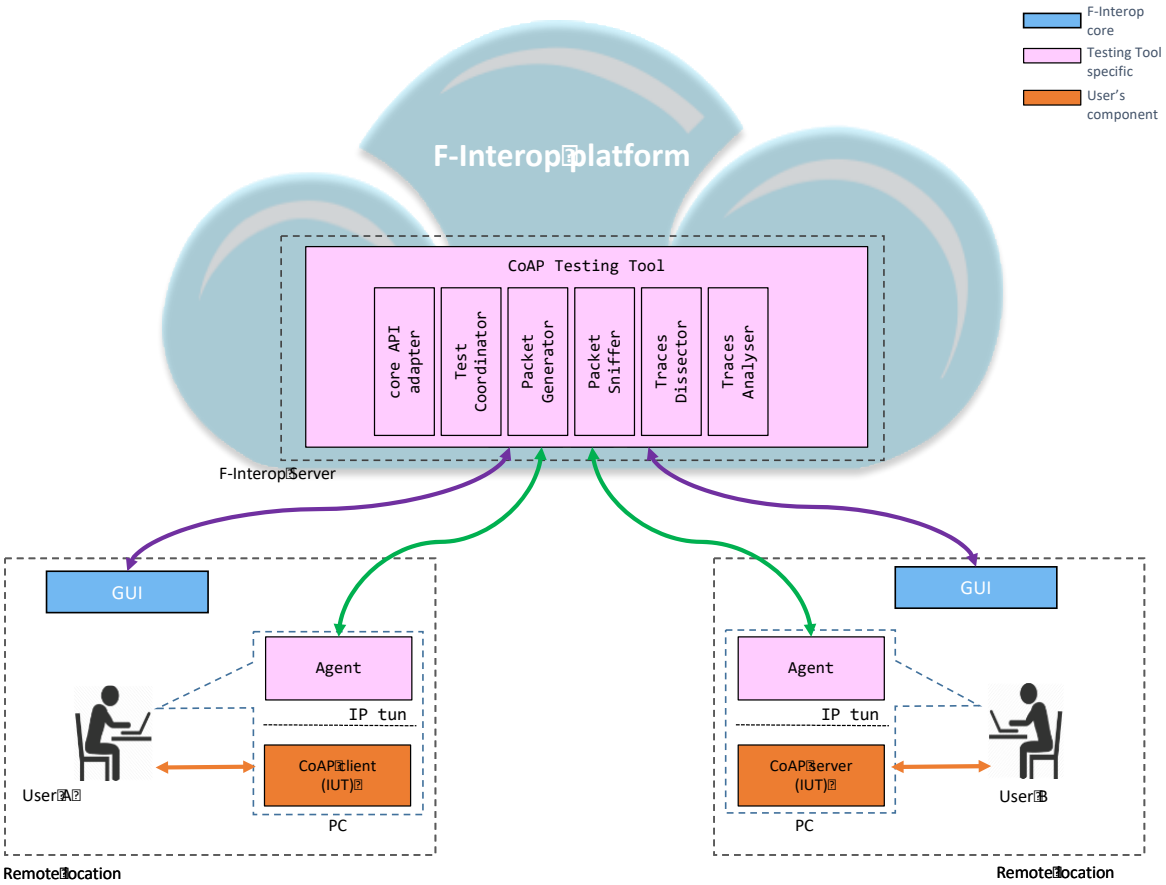


**Figure 2: CoAP interoperability testing tool: user-to-user session**

The main components of the CoAP Testing Tool are:

- **Packet Router and Agents**: establishes a VPN-like setup between IUTs. The controlled environment helps users bypass UDP-blocking firewalls and other middle boxes installed in their facilities. The setup creates IPv6 network interfaces bound to the VPN, therefore users can test their implementations over IPv6 regardless the IP version supported by their internet provider. The Packet Router is the middle-box between IUT1's interface and IUT2's interface, it can act as a lossy gateway for test scenarios which require simulating a lossy context.
- **Test Coordinator**: coordinates the entire interoperability test. It iterates over the test steps described in the test description. It dispatches commands to users through the GUI, based on the TD (e.g. ''user1: CoAP Client is requested to send a GET request''). The user is guided to perform the test remotely.
- **Packet Sniffer:** sniffs the traffic exchanged between IUTs and generates PCAP files records. The component enables the export feature for network traces so users can analyses the exchanged frames using tooling outside of F-Interop e.g. wireshark or some analysis scripts code.
- **Traces Dissector**: dissects the exchanged messages between IUTs and provides a human readable representation of the packets. It provides a wireshark-like view to helps users find problems encountered during the interoperability test execution.
- **Traces Analyzer:** analyses the traffic exchanged between IUTs during a test case. The tool

automatically issues PASS, FAIL, INCONCLUSIVE verdicts after each test case. The analysis is based on the CHECK steps of the test cases description.

### 3.2.2 Integration

The core API adapter module handles the integration with F-Interop's core services such as GUI and Results Repository.

The integration enables two interoperability test use cases:

- remote user-to-user interoperability session, as described in previous figure,
- single-user sessions to test interoperability against reference implementations.

### 3.2.3 Last fine-tuning actions

For the last tuning actions, we will focus on fine-tuning the test cases implementation for the analysis for avoiding any false positive issued verdict.

## 3.3 6TiSCH conformance testing tool

### 3.3.1 Overview

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

- **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.
- **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.
- **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 (http://www.celeryproject.org).

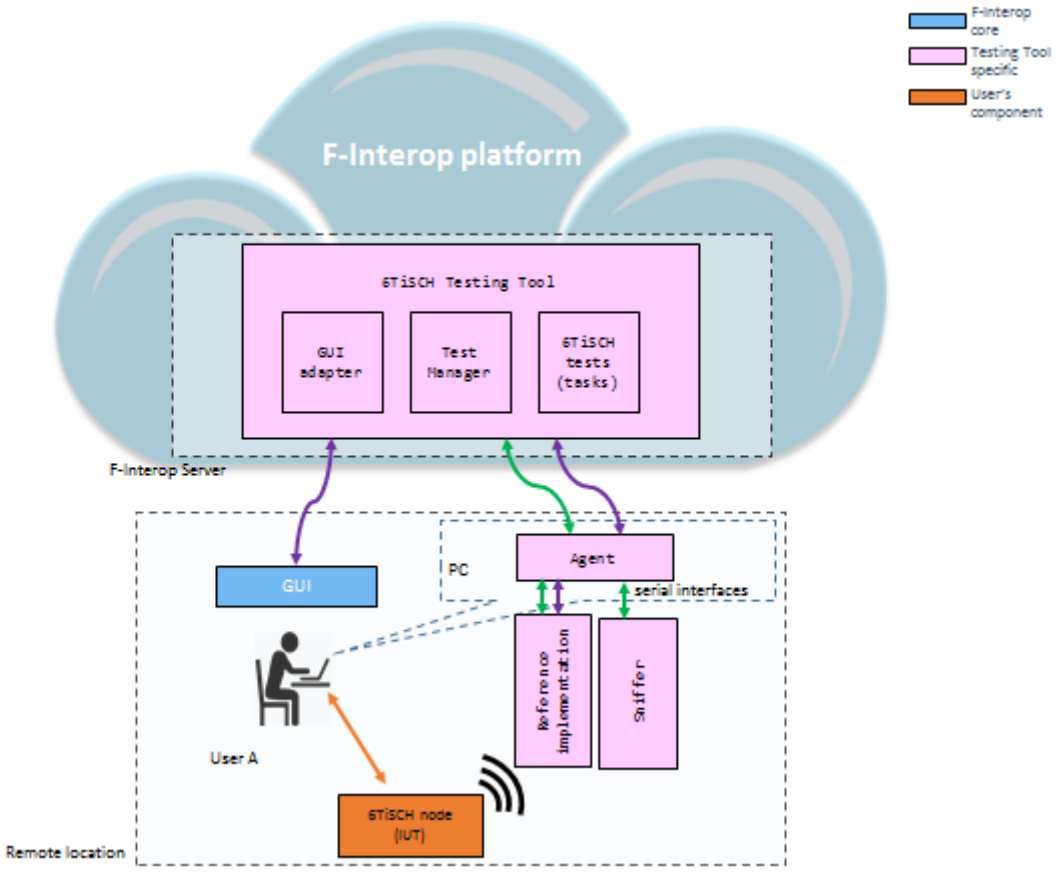


**Figure 3: 6TiSCH conformance testing tool**

## 3.3.2 F-Interop Integration

The figure above 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** and **testcase verdict**. For all the tests, besides the specific feature for which interoperability is checked, the user has to perform some generic steps:

**1)** turn on all the devices (the IUTs, and the packet sniffer);

**2)** wait until the devices enter into a stable state;

**3)** 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 analysed 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.

## 3.3.3 Last actions

For the last actions for 6TiSCH testing tool, we will focus on implementing the feature to update the result to result repository feature. Meanwhile, we will fine-tune the test cases implementation for avoiding issuing any verdict with false status.

## 3.4 oneM2M interoperability testing tool

### 3.4.1 Overview

As described in Figure 4, the oneM2M interoperability testing tool includes the following specific components:

- The oneM2M interoperability testing tool which is composed of the subsequent modules:
  - The Traces Dissector for decoding network traces captured by the sniffer.
  - The Packet Sniffer for sniffing the traffic between the oneM2M interoperability testing components during a session.
  - The Packet generator for the generation and the dispatch of packets towards the client or server endpoints.
  - The Traces Analyzer for the verification of traces of the test session. The verification is post mortem.
  - The Test Coordinator responsible of the management of oneM2M test suites achievement during oneM2M session.

  All the communication between the components respect the API available at http://doc.f-interop.eu/.
- An agent to establish a VPN-like setup between oneM2M endpoints (oneM2M client, namely ADN (Application Dedicated Node), and oneM2M Server, namely CSE (Common Service Entity)).

These components interact with the F-Interop core components such as the Resource repository, the Orchestrator and the GUI to conduct remote interoperability testing with respect to F-Interop API. The core API adapter module is responsible of the integration of oneM2M testing tool components and F-Interop core services.

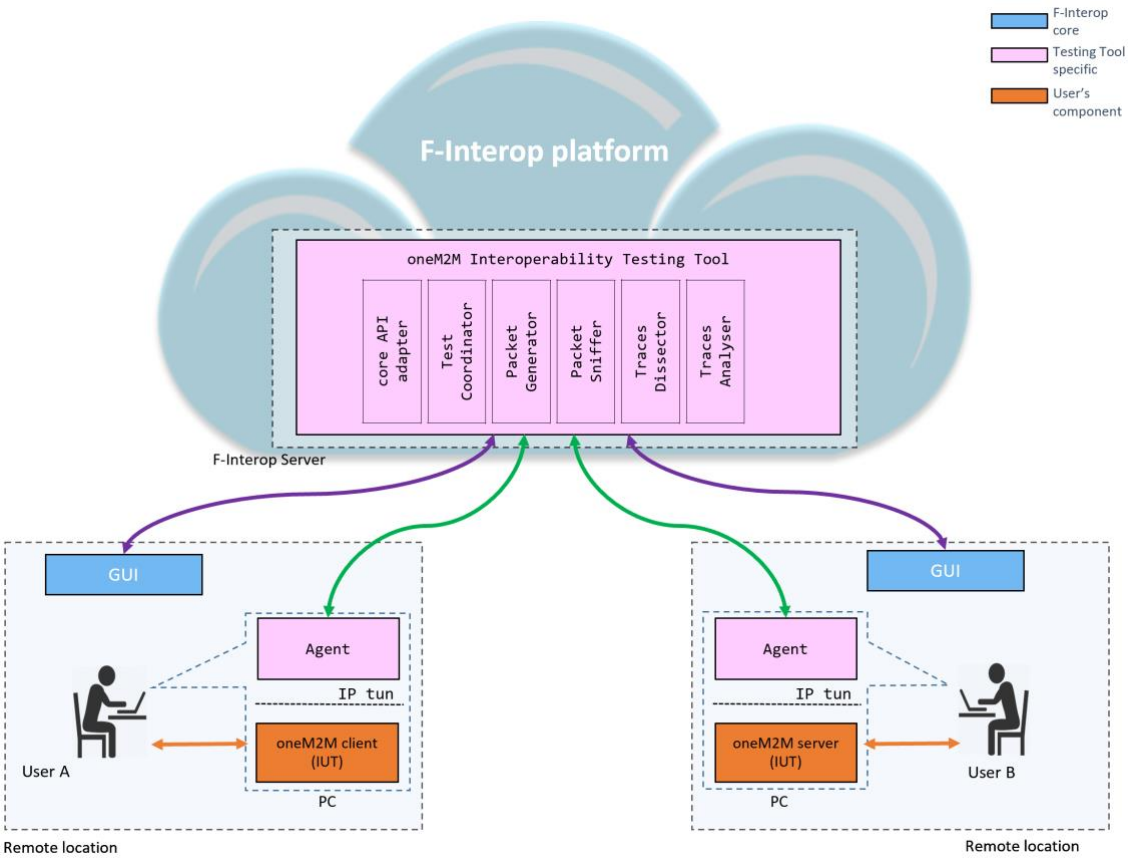


**Figure 4: oneM2M interoperability testing tool: user-to-user session model**

### 3.4.2 Integration

Two testing session models are handled in oneM2M testing tool:

- User-to-user session which is described in the figure above.
- Single user session to enable reference-based oneM2M interoperability testing. A oneM2M client was developed in this purpose. We use OM2M open source project as reference oneM2M server implementation,(http://www.eclipse.org/om2m/).

### 3.4.3 Last actions

For the last actions, we will focus on the integration of single user session model and fine-tuning the test cases implementation.

# 4 Conclusion

This document presented the work related to task T2.3. We present indicators on the status of the work of the final iteration of interoperability/conformance testing tools instantiations. We described the outcome of the work achieved in the development of F-Interop interoperability/ conformance test suites experiences on IoT emerging standards namely CoAP, 6TiSCH, and oneM2M.