Grant Agreement No.: 687871

ARCFIRE

Large-scale RINA Experimentation on FIRE+

Instrument: **Research and Innovation Action**
Thematic Priority: **H2020-ICT-2015**

D4.1 Data Management Plan

Due date of Deliverable: Month 6
Actual submission date: June 30, 2016
Start date of the project: January 1st, 2016. Duration: 24 months
version: V1.0

Project funded by the European Commission in the H2020 Programme (2014-2020)		
Dissemination level		
PU	Public	✓
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

FP7 Grant Agreement No.	687871
Project Name	Large-scale RINA Experimentation on FIRE+
Document Name	Deliverable 4.1
Document Title	Data Management Plan
Workpackage	WP4
Authors	Dimitri Staessens (iMinds) Sander Vrijders (iMinds)
Editor	Dimitri Staessens
Delivery Date	June 30th 2016
Version	v0.1

Table of Contents

List of Figures

List of Tables

1 Introduction

This Data Management Plan (DMP) intends to outline the handling of data gathered during the ARCFIRE project, from the point where it is gathered until the archiving process at the end of the project.

1.1 Purpose

This Data Management Plan ensures that policies are in place to:

- Facilitate the generation of data and analyses of that data by ARCFIRE;
- Outline the procedures and formats for transforming raw data into processed results.
- Ensure that data required to reconstruct published results are made available online in time to facilitate the peer review process, for instance on the ARCFIRE website.
- Ensure that raw and processed data sets, together with appropriate documentation, are released in timely ways as structured archive volumes to Open Access repositories for distribution to the FIRE+ community and others beyond the duration of the ARCFIRE project.

1.2 Scope

The scope of this data management plan focuses on:

- Timely reduction of raw data into structured results, along with documentation that determines when and where the data were acquired, and for what purpose.
- Timely generation and validation of archive volumes containing standard data products and documentation.
- Timely delivery of archive volumes to open repositories for distribution to the FIRE+ community and others.
- Timely posting of new and exciting data sets and results on the Internet for public access.
- Timely announcement of the availability of results via social media

1.3 Responsibilities

The development, maintenance and management of the Data Management Plan in is the responsibility of iMinds. The current responsible person is Dr. Dimitri Staessens (dimitri.staessens@intec.ugent.be). This Data Management Plan is not seen as a static document and will be updated during the project whenever appropriate.

1.4 Change Control

The validity of the Data Management Plan will be evaluated at least every six (6) months or whenever a revision is necessitated during the project. If changes are required, a new document will be prepared for internal use in the consortium, indicated with a minor version: D4.1.x. These documents can be made available upon request.

1.5 Relevant Documents

This document is structured according to the NASA *G*uidelines for Development of a Project Data Management Plan (PDMP).[1]

2 Project Overview

The ARCFIRE project will investigate RINA, the Recursive Internet Architecture at scale on FIRE+ testbeds. It will use publicly available Free/Libre Open Source Software (FLOSS) from a number of previous and currently running projects, including

- FP7-IRATI: This EC funded project developed a GPL/LGPL licensed implementation of RINA concepts for OS/Linux, called IRATI. It is available on Github and is seen as project background for ARCFIRE. ARCFIRE will contribute to the FLOSS developments of IRATI.
- GEANT-IRINA: This EC funded project developed a FLOSS traffic generation tool for the IRATI under the Geant outward software license. ARCFIRE may use this tool. ARCFIRE will develop a much richer tool set that will be made available under a different license (GPL).
- FP7-PRISTINE: This ongoing EC funded project develops a Management System for IRATI. ARCFIRE will use and contribute further to this software.
- FWO-RINAiSense: This Flemish Government funded project is developing a smaller scale user space RINA implementation aimed at constrained resource devices. ARCFIRE may make use of publicly available software from RINAiSense.

2.1 Project Objectives

ARCFIRE sets out to prove the effectiveness of the RINA architecture in mitigating key issues observed in the past decades in the deployment of TCP/IP as the underlying infrastructure of the global Internet. It will deploy key experiments on testbeds in Europe (provided by FIRE+)

[1] `http://nssdc.gsfc.nasa.gov/nssdc/pdmp_guidelines_march93.rtf`

and the U.S. (as made available by GENI). ARCFIRE will develop the tools necessary to deploy experiments using the RINA prototypes quickly and efficiently.

ARCFIRE will develop a framework for deploying test programs on a variety of testbeds and gather data regarding resiliency and manageability of RINA networks, as compared to TCP/IP-based networks.

3 Testbed description

The experiments in the ARCFIRE project will run on general purpose hardware and virtual machines running FLOSS. The four experiments will be conducted on selected testbeds available to the project consortium. The two currently available testbed infrastructures are provided by the Fed4FIRE project in the EU (`http://www.fed4fire.eu`) and GENI (`https://www.geni.net/`) in the United States.

A detailed testbed report will be delivered as part of ARCFIRE D4.2.

4 ARCFIRE experimentation framework

The ARCFIRE experimentation framework has to be developed during the project and its capabilities will impact the Data Management. We foresee that a number of existing tools will be integrated. These tools will serve as probes and be controlled from the ARCFIRE framework.

4.1 probes

We give a brief description of such probes below:

4.1.1 iperf

iPerf3 is a tool for active measurements of the maximum achievable bandwidth on IP networks. It supports tuning of various parameters related to timing, buffers and protocols (TCP, UDP, SCTP with IPv4 and IPv6). For each test it reports the bandwidth, loss, and other parameters. This is a new implementation that shares no code with the original iPerf and also is not backwards compatible. iPerf was orginally developed by NLANR/DAST. iPerf3 is principally developed by ESnet / Lawrence Berkeley National Laboratory. It is released under a three-clause BSD license. It can produce output in JSON format.

4.1.2 netperf

Netperf is a benchmark that can be used to measure the performance of many different types of networking. It provides tests for both unidirectional throughput, and end-to-end latency. The environments currently measureable by netperf include:

- TCP and UDP via BSD Sockets for both IPv4 and IPv6
- DLPI
- Unix Domain Sockets
- SCTP for both IPv4 and IPv6

4.1.3 tcpdump

Tcpdump prints out a description of the contents of packets on a network interface that match the boolean expression; the description is preceded by a time stamp, printed, by default, as hours, minutes, seconds, and fractions of a second since midnight. It can also be run with the -w flag, which causes it to save the packet data to a file for later analysis, and/or with the -r flag, which causes it to read from a saved packet file rather than to read packets from a network interface. In all cases, only packets that match expression will be processed by tcpdump.

The MIME type application/vnd.tcpdump.pcap has been registered with IANA for pcap files. The filename extension .pcap appears to be the most commonly used along with .cap and .dmp. Tcpdump itself doesn't check the extension when reading capture files and doesn't add an extension when writing them (it uses magic numbers in the file header instead). However, many operating systems and applications will use the extension if it is present and adding one (e.g. .pcap) is recommended.

4.1.4 wireshark

Wireshark is a GUI network protocol analyzer. It lets you interactively browse packet data from a live network or from a previously saved capture file. Wireshark's native capture file format is pcap format, which is also the format used by tcpdump and various other tools.

Wireshark can output XML, PostScript, csv, or plain text

4.1.5 RINA traffic generator (tgen)

The RINA traffic generator is a tool developed during the Geant3 IRINA project, designed to produce CBR and Poisson-distributed traffic for the IRATI implementation. It outputs periodic statistics in .csv format.

4.1.6 OML

OML is a generic software framework for measurement collection. It allows the developer of applications to define customisable measurement points (MP) within the application code. It consists of two main components:

- OML client library: it provides an API for applications to collect measurements that they produce. It exists for different languages including C and Python.
- OML Server: the OML server component is responsible for collecting and storing measurements inside a database. Currently, SQLite3 and PostgreSQL are supported as database backends.

MPs are usually defined within the application code in the form of tupples like: `("app_name", "measurement_name1:measurement_type1 measurement_name2:measurement_type2")` , depending on the binding used. The data recollected in the OML server will be saved in the configured database. The format of this data depends on the user after querying the database.

4.1.7 ARCFIRE tools development

ARCFIRE will develop a framework that will combine existing tools and develop a frontend to control and deploy these tools on the FIRE testbeds quickly and efficiently.

Scripts will be made public open source under appropriate software licenses. We don't plan for the tool to generate any data by itself.

4.2 Data formats

This section gives a brief summary of data formats that will be used for the output of ARCFIRE.

4.2.1 txt

Plaintext files will be used for the metadata accompanying any ARCFIRE data products.

4.2.2 pcap

Applications and libraries should use the pcap library to read savefiles, rather than having their own code to read savefiles. If, in the future, a new file format is supported by libpcap, applications and libraries using libpcap to read savefiles will be able to read the new format of savefiles, but applications and libraries using their own code to read savefiles will have to be changed to support the new file format.

"Savefiles" read and written by libpcap and applications using libpcap start with a per-file header. The format of the per-file header is:

- Magic number
- Major version

- Minor version
- Time zone offset
- Time stamp accuracy
- Snapshot length
- Link-layer header type

The magic number is used to discern the format (byte order and timestamp).
Other important parameters are:
A 4-byte number giving the ”snapshot length” of the capture; packets longer than the snapshot length are truncated to the snapshot length.
Following the per-file header are zero or more packets; each packet begins with a per-packet header, which is immediately followed by the raw packet data. The format of the per-packet header is:

- Time stamp, seconds value
- Time stamp, microseconds or nanoseconds value
- Length of captured packet data
- Un-truncated length of the packet data

All fields in the per-packet header are in the byte order of the host writing the file.
For a full description, see the manpage of pcap-savefile.

4.2.3 csv

RFC 4180 proposes a specification for the CSV format, and this is the definition commonly used. However, in popular usage ”CSV” is not a single, well-defined format. As a result, in practice the term ”CSV” might refer to any file that

- is plain text using a character set such as ASCII, various Unicode character sets (e.g. UTF-8), EBCDIC, or Shift JIS,
- consists of records (typically one record per line),
- with the records divided into fields separated by delimiters (typically a single reserved character such as comma, semicolon, or tab; sometimes the delimiter may include optional spaces),

- where every record has the same sequence of fields.

Within these general constraints, many variations are in use. Therefore, without additional information (such as whether RFC 4180 is honored), a file claimed simply to be in "CSV" format is not fully specified. As a result, many applications supporting CSV files allow users to preview the first few lines of the file and then specify the delimiter character(s), quoting rules, etc. If a particular CSV file's variations fall outside what a particular receiving program supports, it is often feasible to examine and edit the file by hand (i.e., with a text editor) or write a script or program to produce a conforming format.

ARCFIRE .csv files will be formatted according to RFC4180.

4.2.4 xml

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is defined by the W3C's XML 1.0 Specification and some associated open standards.

4.3 JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

4.4 Summary

A summary of the probes is given in Table 1.

Table 1: Summary of probes

Tool	type of data gathered	raw output format
iperf	bandwidth	txt, JSON
netperf	bandwidth	txt
tcpdump	raw packet data	pcap
rina-tgen	bandwidth	txt, csv

5 Project Data Flow

ARCFIRE will generate only computer science related data and measurements, so all processing can be done in situ after the experiment is done. ARCFIRE will generate large quantities of network traffic, that can not be stored in its totality and will be processed immediately after acquisition. For post-processing, relevant subsets of that data will be stored at the facility testbed (on the test machine or a centralised server provided by the testbed facility until these resources need to be released. In such cases the data may be moved to a central server at an ARCFIRE partner for further analysis. When all analysis is done, the necessary data to reproduce the results will be packaged and archived in a zip archive or tarball.

6 Products

Products resulting from the project include raw experiment data sets, and associated products such as statistical analysis data.

6.1 Experiment data Products

The project will generate raw data such as tcpdump traces, that could reach orders of magnitudes exceeding Terabytes of data very quickly (The iLab.t experimentation facility provides Gigabit links). Such data will not be archived or saved, but some traces may be filtered to illustrate key findings in smaller data sets (not exceeding a couple of megabytes in size).

6.1.1 ARCFIRE data product template

Data products for ARCFIRE will be accompanied by a metadata sheet including the information from Table 2.

7 Archive location

The selected archive for ARCFIRE is the Zenodo archive. All ARCFIRE data products will be grouped in a community, located at the following URL: `https://zenodo.org/collection/user-arcfire`.

8 Licensing

All open data from ARCFIRE contributed to the Zenodo repository is planned to be released under the Creative Commons CC-BY license: `https://creativecommons.org/licenses/by/4.0/legalcode`.

Table 2: ARCFIRE data set template

Data set reference and name	Identifier for the data set to be produced.
Data set description	Description of the data that will be generated or collected, its origin (in case it is collected), nature and scale and to whom it could be useful, and whether it underpins a scientific publication. Information on the existence (or not) of similar data and the possibilities for integration and reuse.
Standards and metadata	Reference to existing suitab le standards of the discipline. If these do not exist, an outline on how and what metadata will be created.
Data sharing	Description of how data will be shared, including access procedures, embargo periods (if any), outlines of technical mechanisms for dissemination and necessary software and other tools for enabling reuse, and definition of whether access will be widely open or restricted to specific groups. Identification of the repository where data will be stored, if already existing and identified, in dicating in particular the type of repository (institutional, standard repository for the discipline, etc.). In case the dataset cannot be shared, the reasons for this should be mentioned (e.g. ethical, rules of personal data, intellectual property, commercial, privacy related, security related).
Archiving and preservation (including storage and backup)	Description of the procedures that will be put in place for long term preservation of the data. Indication of how long the data should be preserved, what is its approximated end volume, what the associated costs are and how these are planned to be covered.