
H2020-ICT-2018-2-825377

# UNICORE

## UNICORE: A Common Code Base and Toolkit for Deployment of Applications to Secure and Reliable Virtual Execution Environments

Horizon 2020 - Research and Innovation Framework Programme

# D6.2 Data Management Plan

Due date of deliverable: 30 June 2019

Actual submission date: 30 June 2019

| | |
|---|---|
| Start date of project | 1 January 2019 |
| Duration | 36 months |
| Lead contractor for this deliverable | NEC Laboratories Europe GmbH (NEC) |
| Version | 1.0 |
| Confidentiality status | Public |

**Abstract**

The goal of the EU-funded UNICORE project is to develop a common code-base and toolchain that will enable software developers to rapidly create secure, portable, scalable, high-performance solutions starting from existing applications. The key to this is to compile an application into very light-weight virtual machines – known as unikernels – where there is no traditional operating system, only the specific bits of operating system functionality that the application needs. The resulting unikernels can then be deployed and run on standard high-volume servers or cloud computing infrastructure.

This deliverable defines the project's data management plan. By and large, Unicore will not rely on any user or senstive data: where needed, tests will be run using already publicly available data sets or synthetic data; this deliverable explain, on a per use case basis, details regarding this. In addition, this deliverable details Unicore's commitment to making most of its output, be it source code, presentations or scientific papers, publicly available.

**Target Audience**

The target audience for this document is **public**.



**Impressum**

| | |
|---|---|
| Full project title | UNICORE: A Common Code Base and Toolkit for Deployment of Applications to Secure and Reliable Virtual Execution Environments |
| Title of the workpackage | D6.2 Data Management Plan |
| Editor | NEC Laboratories Europe GmbH (NEC) |
| Project Co-ordinator | Emil Slusanschi, UPB |
| Technical Manager | Felipe Huici, NEC |
| **Copyright notice** | © 2019 Participants in project UNICORE |

# List of Authors

| | |
|---|---|
| Authors | Felipe Huici (NEC), Gino Carozzo (NXW), Ioan Constantin (ORO), Stephen Parker (XLRN), Jesus Martin (CSUC), Gaylor Bosson (EPFL) |
| Participants | NEC, NXW, VUA, ORO, XLRN, CSUC, EPFL |
| Work-package | WP3 - Core Implementation |
| Security | PUBLIC |
| Nature | R |
| Version | 1.0 |
| Total number of pages | 9 |

# Contents

# 1 Overview

This deliverable provides the data management plan for open research data conforming to the guidelines of the H2020 framework programme. By and large, Unicore will not make use of any user or senstive data: where needed, the project will rely either on already publicly available datasets or on synthetic data or synthetic network traffic to conduct tests.

Apart from the use of data, Unicore will seek, as much as possible, to make its results and output publicly available:

- **Public deliverables** will be available on the project's website soon after their due date (http://unicore-project.eu/).
- **Scientific papers**: the Unicore partners are committed to, where permitted, making all of the scientific publications availabe publicly. This will be done through the conference websites for conferences that make them available on their sites, on the Unicore website, or through third-party sites such as https://arxiv.org/.
- **Public presentations, posters, videos and flyers**: public presentations will be available from the respective conference sites' where possible, or through the Unicore website otherwise (likewise for posters). Unicore videos of a dissemination nature will also be made available though the project's website, as will flyers or any other such dissemination material.
- **Source code**: The Unicore project has a strong commitment, as specified in the Description of Work, towards open source. In fact, desipte the project having started relatively recently, large portions of the code produced by it are already publicly available on the xenbits Unikraft repository (http://xenbits.xen.org/gitweb/?a=project_list;pf=unikraft. Generally, the code produced by this project will be found in this repository or on Github repos.

In the next chapter we provide more details regarding the project's data management plan on a per use case basis.

# 2 Data Management Plan

## 2.1 Serverless Use Case

This use case is focused basically in conversion of two kind of multimedia files: image files and video files. These files are served from digital institutional repositories so this conversion is needed because providers of the multimedia content upload it as raw format without compression.

The content of the different institutional repositories is public, so it is not restricted and everyone can watch the videos and the images, as well as reproduction, distribution and public difusion because they have creative commons license. A Creative Commons (CC) license is one of several public copyright licenses that enable the free distribution of an otherwise copyrighted "work". A CC license is used when an author wants to give other people the right to share, use, and build upon a work that they (the author) have created. One of the most representative institutional repositories that has this content is CALAIX:http://calaix.gencat.cat/locale-attribute=en.

The institutional repositories admins cap upload of their digital content to the appropiate repository, but some of them are so huge that it cannot be visualized directly. For this reason, we need a format conversion. In our use case, the media converter process checks if a digital content has been uploaded, and it can convert images and videos.

For the image conversion the media converter uses ImageMagick and for video conversion it uses the ffmpeg utility. The ffmpeg utility uses different profiles to convert the video files especifically to watch this converted content as lightweight distribution and broadcast, but without loss quality definition.

**Data Management Within the Use Case**: No senstive or public data will be used in this use case.

## 2.2 NFV Use Cases

### 2.2.1 Broadband Network Gateway (BNG) for Wired Internet Access

This use case will create a BNG as a unikernel VM, moving from t**Data Management Within the Use Case**:he current legacy infrastructure of a single physical BNG to a virtualized, monolithic BNG over Open Stack and finally to a unikernel-based BNG with fast service instantiation, dedicated instance (VM) per customer and scalability for thousands of customers. ORO will use its readily deployed testbed infrastructure to validate the use case. The testbed consists of PNF devices such as networking components, firewalls, load balancers and a physical BNG; and of compute, control and storage nodes serving a virtualized infrastructure openstack, vmware, docker, kubernetes, and ONAP capable.

**Data Management Within the Use Case**: Data will be used but will be synthetically generated.

### 2.2.2 Wireless 5G vRAN NFV Clusters

Accelleran develops software for small cells (base stations) for 4G mobile networks. To date, commercial releases of the software have been in the form of embedded software running on specialised Original Design

Manufacturer (ODM) hardware. With the advent of 5G networks, concepts such as Network Function Virtualisation and Radio Access Network (RAN) disaggregation have gained more prominence. These concepts necessitate the break down of monolithic networking software blocks into smaller components which can then be deployed in NFV environments. Accelleran will take aspects of its existing embedded system and adapt it to run on unikernels. In this scenario, the lower layers of the radio side protocol stack will continue to execute on ODM hardware and the Cell Control Plane software will execute remotely on general compute servers running unikernels. A more complete description of this use case can be found in deliverable D2.1

**Data Management Within the Use Case**: The testbed used for the execution of this use case consists of several standard network components operating in a controlled and isolated environment. In this environment no real end user data is used and all data are either or simulated or synthetically generated.

### 2.2.3 Customer Premise Equipment NFV and Micro Services

Ekinops has recently extended its products portfolio by developing the Open Virtualization Platform (OVP). The OVP is basically a universal Customer Premises Equipment (uCPE) that allows to instantiate and manage the life cycle of network services. A network service is a set of chained virtual machines and container-based embedded services (such as probes, NAT and DHCP).

The current implementation of OneOS6 based on Linux and DPDK is resource hungry: it requires at least two CPU cores and 2GB of RAM. This means that on a standard 8 cores machine we can only run three vRouters assuming that the hypervisor and the management plane require two CPUs.

In this use case, we will adopt the unikernel architecture to split our OneOS6 software into smaller micro services. Each micro-service handles a very specific network function such as NAT, DHCP, QoS, routing and would run as a unikernel. This allows to instantiate multiple VNFs on the same CPU core and increase the hardware efficiency, thus boosting the competitiveness of our products.

**Data Management Within the Use Case**: Data will be used but will be synthetically generated.

## 2.3 Home Automation/IoT Use Case

The Home Automation/IoT use case in Unicore consists of migrating some of the IoT middleware and gateway functions currently available in the Symphony platform by Nextworks (see deliverable D2.1 for more detailed description) to a unikernel-based implementation.

The use case will take place at Nextworks premises in Pisa, where Symphony is currently deployed and manages the entire office automation and domotic control of lighting, conditioning, and curtains, among others.

The testing of the use case will be mainly oriented towards assessing unikernel functionality and performance. A group of Nextworks researchers will be involved to anonymously operate the system and monitor performance of the deployed unikernels in the end-to-end service chain.

Up to 50 types of sensors and actuators are planned to be used in the Home Automation/IoT testing environment. During the tests, the following data, related to the home automation and power consumption aspects,

are expected to be collected:

- **Comfort**: Lights/bulbs, switches/dimmers, curtains/blinds, media systems, multisensor (motion detector, illumination, magnet and temp)
- **Savings**: thermostat managing, electricity automatic counter.
- **Security**: motion detection sensor, camera, indoor siren, smart floor sensor readings, CO and/or CO2 levels, NOx and/or NO2 levels
- **Other**: burglary security status, status of devices and sensors, control status of actuators, scheduler control commands

**Data Management Within the Use Case**: For the execution of this use case, logs related to sensing and actuation commands on the aforementioned sensors and actuators as well as within the Symphony middleware components will be collected. No personal identifiable and sensitive information will be used and stored in the system. A user of the Symphony platform will be identified by a generic userID, which is independent from personal identification data of the tester. Due to the scope and controlled testing nature and form of the planned use case experiments, no consent forms, nor specific data handling procedures need to be prepared and compiled by experimenters for the execution of the trials. Specifcally, (a) no personally-identifiable information will be used for the tests nor shared with project partners; (b) anonymized data is stored only in Symphony internal storage for debug purposes; and (c) data and logs are deleted after the execution of the test

## 2.4 Smart Contracts Use Case

Our current design uses a unikernel as a generic smart contract executor. The unikernel performs 4 steps. First, it loads a bytecode representation of the smart contract from the blockchain (e.g., the format could be the IR in LLVM). Then it statically checks that it can be executed properly, e.g., no infinite loops. Next it executes the smart contract using input state from the blockchain, and finally it stores the output state on the blockchain. The software we use is: LLVM tools, Go and its environment

**Data Management Within the Use Case**: There is no user data involved in this use case: we simply create synthetic transactions as input for smart contracts.

# 3 Conclusions

In this deliverable we detailed the project's data management plan. On a per use case basis, we explained that no sensitive or user data will be used in Unicore. In addition, we outlined our plans to make most of the project's materials and outcomes publicly available either though its official website, through conference websites, or through public open source code repositories.