Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications

Deliverable D4.1

Edge/Cloud continuum management framework report (I)

ACCORDION receives funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871793

DOCUMENT INFORMATION

PROJECT	
PROJECT ACRONYM	ACCORDION
PROJECT FULL NAME	Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications
STARTING DATE	01/01/2020 (36 months)
ENDING DATE	31/12/2022
PROJECT WEBSITE	http://www.accordion-project.eu/
TOPIC	ICT-15-2019-2020 Cloud Computing
GRANT AGREEMENT N.	871793
COORDINATOR	CNR
DELIVERABLE INFORMATION	
WORKPACKAGE N. | TITLE	WP 4 |Edge/Cloud continuum management framework
WORKPACKAGE LEADER	TID
DELIVERABLE N. | TITLE	D4.1: Edge/Cloud continuum management framework report (I)
EDITOR	Zinelaabidine Nadir (AALTO)
CONTRIBUTOR(S)	Tarik Taleb (AALTO), John Violos (ICCS), Stylianos Tsanakas (ICCS), Tita Pagoulatou (ICCS), Theodoros Theodoropoulos (HUA), Massimo Coppola (CNR), Patrizio Dazzi (CNR), Luca Ferrucci (CNR), Ferran Diego (TID), Eduard Marin (TID), Nicolas Kourtelis (TID)
REVIEWER	K. Tserpes (HUA)
CONTRACTUAL DELIVERY DATE	02/2021
ACTUAL DELIVERY DATE	28/02/2021
VERSION	1.0
TYPE	Report / Websites, patents filling, etc.
DISSEMINATION LEVEL	Public / Private
TOTAL N. PAGES	55
KEYWORDS	Edge, Cloud, Compute and Network Orchestration, Resilience, Privacy, Security

EXECUTIVE SUMMARY

This deliverable provides the first report summarizing the scientific advancements, during the first year of the project, achieved by WP4 Tasks. Work Package (WP) 4, dubbed **Edge/Cloud continuum management framework,** is organized around 6 Tasks is to develop a framework that efficiently manages the deployment and runtime of ACCORDION applications on the continuum.

The first section of this deliverable provides the objectives of this deliverable and its relation to other deliverables of the projects. The second section gives an overview of WP6 Tasks. The following sections, Sections 3-7, are dedicated respectively to tasks 1-6. Section 3, dubbed ACCORDION Intelligent orchestrator (AIO), combines the first two tasks of the projects namely: 1) Task 4.1 Intelligent, adaptive resource orchestration, and 2) AI-based network orchestration. In Section 4, Resilience policies & mechanisms task is presented. Section 5 is dedicated to T4.4, Security-enhanced application development & deployment, while the following section is devoted to privacy-preserving mechanisms. The last section, Intelligent edge/cloud continuum orchestrator, reports the work achieved in order to integrate all the components, that are being developed by each task, of this WP.

In this first release of the deliverable, D4.1, it did not consider how this framework will be integrated with the other frameworks of the project, nor how each component communicates with the other components of the ACCORDION framework. For each task, involved partners provide the objectives targeted by the task for building a framework that efficiently orchestrates the federation resources and optimizes ACCORDION applications' deployment and runtime. For each section, sections 3-7, the first subsection provides a description and objectives of the task followed by its requirements. Furthermore, the scientific challenges & advancements achieved during the first cycle of the project are presented in the next subsection. The penultimate subsection describes the output of the task. Finally, the last subsection provides plans for future work.

DISCLAIMER

ACCORDION (871793) is a H2020 ICT project funded by the European Commission.

ACCORDION establishes an opportunistic approach in bringing together edge resource/infrastructures (public clouds, on-premise infrastructures, telco resources, even end-devices) in pools defined in terms of latency, that can support NextGen application requirements. To mitigate the expectation that these pools will be "sparse", providing low availability guarantees, ACCORDION will intelligently orchestrate the compute & network continuum formed between edge and public clouds, using the latter as a capacitor. Deployment decisions will be taken also based on privacy, security, cost, time and resource type criteria.

This document contains information on ACCORDION core activities. Any reference to content in this document should clearly indicate the authors, source, organisation and publication date.

The document has been produced with the funding of the European Commission. The content of this publication is the sole responsibility of the ACCORDION Consortium and its experts, and it cannot be considered to reflect the views of the European Commission. The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.

The European Union (EU) was established in accordance with the Treaty on the European Union (Maastricht). There are currently 27 members states of the European Union. It is based on the European Communities and the member states' cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice, and the Court of Auditors (http://europa.eu.int/).

REVISION HISTORY LOG

VERSION No.	DATE	AUTHOR(S)	SUMMARY OF CHANGES
0.1	25/11/2020	Zinelaabidine Nadir (Aalto University) Tarik Taleb (Aalto University)	First draft of the ToC
0.2	30/1/2021	All	First round of contributions
0.3	22/02/2021	Zinelaabidine Nadir (Aalto University)	First revision, checking Headings. figures, references,
0.4	22/02/2021	John Violos (ICCS)	Revision section 4
0.5	23/02/2021	Konstantinos Tserpes (HUA)	Internal review
0.6	26/02/2021	All	Updates afted internal review
1.0	28/02/2021	Zinelaabidine Nadir (Aalto University)	Final revision

GLOSSARY

EU	European Union
EC	European Commission
H2020	Horizon 2020 EU Framework Programme for Research and Innovation
QoS	Quality of Services
RPM	Resilience Policies & Mechanisms
EM	Edge Miniclouds
CC	Central Clouds
AIO	ACCORDION Intelligent Orchestrator
QoE	Quality of Experience
VM	Virtual Machine
RPM	Resilience policies & mechanisms
PoI	Point of Interest
RNN	Recurrent Neural Networks
CNN	Convolutional Neural Network
LSTM	Long short-term memory
PSO	Particle Swarm Optimization
BPSO	Bayesian Particle Swarm Optimization
GA	Genetic Algorithm
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error.
FT	Fault Tolerance
ANN	Artificial Neural Networks
HBES	Hybrid Bayesian Evolution Strategy
MRMOGAP	Multi-resource Multi-Objective generalized assignment problem
NCO	Network Resource Orchestrator
CRO	Computing Resource Orchestrator
DS-CRO	Domain Specific Compute Resource Orchestrator
E2E-CRO	End-to-End Compute Resource Orchestrator
DS-NRO	Domain Specific Network Resource Orchestrator
E2E-NRO	End-to-End Network Resource Orchestrator
CSA	Cloud Security Alliance
CIS	Center for Information Security organization
SAST	Static Application Security Testing
DAST	Dynamic Application Security Testing

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Scope and objectives of the document

This deliverable, D4.1, represents the first release of a series of reports that summarizes the scientific advancements achieved, during a cycle of the ACCORDION project. This release describes the 5 (five) tasks of WP4, their objectives and their output for building a framework that efficiently orchestrates the federation resources and optimizes ACCORDION applications' deployment and runtime. For each task, involved partners provide the research challenges and advancements achieved during the first cycle of the project, (M04-M13). They also provide their plans for the next cycle of the project. Furthermore, at this stage of the project, this deliverable did not consider how this framework will be integrated with the other frameworks of the project, nor how each component communicates with the other components of the ACCORDION framework.

1.2 Structure of the document

This deliverable contains 6 (six) sections. The following section, Section 2 gives a brief description of WP4 tasks. Section 3 provides a description of WP4 tasks and how they relate to ACCORDION architecture. The following sections, Sections 4-8, describes respectively tasks 1-5 of WP4 as follows:

- Section 3: ACCORDION Intelligent Orchestrator, Task 4.1 and 4.2, led by CNR and Aalto
- Section 4: Resilience policies & mechanisms, Task 4.3, Led by ICCS
- Section 5: Security-enhanced application development & deployment, Task 4.4, led by HPE
- Section 6: Privacy-preserving mechanisms, Task 4.5, Led by TID
- Section 7: Intelligent edge/cloud continuum orchestrator, Task 4.6, Led by TID

For each section, the first subsection provides a description and objectives of the task followed by their requirements in the next section. Furthermore, the scientific challenges and advancements achieved during the first cycle of the project are presented in the next subsection. The penultimate subsection describes the output of the respective task, whereas the last subsection provides plans for future work. Finally, Section 8 concludes this deliverable.

1.3 Relationship with other documents

All the tasks of WP4 have a relationship with WP2, WP3 and WP4. This document is tightly linked to four (04) deliverables of WP2 that have preceded this deliverable. Each one of these deliverables describes: **1)** User requirements **D2.1-M08**, **2)** State of the art report **D2.2-M10**, **3)** Architecture design **D2.3-M12** and **4)** Data requirements analysis and collection **D2.4-M12**. This deliverable is also accompanied with **D4.2, Edge/Cloud continuum management framework implementation,** which provides the first version of each component of this WP.

These WP tasks relate also to other tasks of WP3 and WP5. However, this deliverable did not consider how this framework will be integrated with the other frameworks of the project, nor how each component communicates with the other components of the ACCORDION framework. But we may refer to other components of WP3 and WP5 which are described in D3.1 and D5.1 respectively. For instance, both ACCORDION Intelligent Orchestrator (AIO) and Resilience policies & mechanisms (RPM) are both linked to components of WP3. AIO is also related to WP5 as AIO takes part in the management of the ACCORDION application' deployment and runtime. Furthermore, both T4.4, Security-enhanced application development & deployment and T4.5, privacy-preserving mechanisms relate to WP5 as they develop tools for making the system more secure by inspecting the application before its deployment into the federation.

2 Architecture

2.1 General Architecture

The aim of WP4, dubbed **Edge/Cloud continuum management framework**, is to efficiently manage the deployment and runtime of ACCORDION applications on the continuum. Thus, its main objectives are:

- Create an orchestrator that will be able to manipulate the underlying edge VIMs, public clouds and network resources, as well as applications' components and ACCORDION services in order to optimize the deployment and runtime of ACCORDION applications on the continuum.
- Infuse applications with end-to-end resilience, security, and privacy properties.

This framework is organized around the underlying five (05) components, where each component has a specific task in the management of the deployment and runtime of ACCORDION applications:

- ACCORDION Intelligent Orchestrator (AIO)
  - ACCORDION Compute Resource Orchestrator (CRO)
  - ACCORDION Network Resource Orchestrator (NCO)
- Resilience policies & mechanisms (RPM)
- Security-enhanced application development & deployment
- Privacy-preserving mechanisms (PPM)
- Intelligent edge/cloud continuum orchestrator

Each component of this framework is briefly described in the following subsections, while the following sections are dedicated to each component.

2.2 ACCORDION Intelligent orchestrator

The ACCORDION Intelligent Orchestrator (AIO) is at the heart of the ACCORDION platform. Its objective is to provide the necessary functionalities for managing all types of ACCORDION resources and provides their allocation plans for deploying ACCORDION applications while considering their KPIs. AIO is composed of two modules: i) **ACCORDION Compute Resource Orchestrator** for managing the ACCORDION compute resources and ii) **ACCORDION Network Resource Orchestrator** for managing the network resources. AIO relies on monitoring the ACCORDION resources and applications to achieve the desired applications' KPI (QoS/QoE) and optimize resources utilization.

2.2.1 ACCORDION Compute Resource Orchestrator (CRO)

This component, Compute Resource Orchestrator (CRO), is responsible for allocating of the available compute resources to deploy applications while considering their KPIs by providing Service Allocation Plans (SAP). This component also considers modifying or composing a new SAP when a QoS/QoE violation or another type of disrupting event is detected and signalled by the VIM Monitoring Agents or other ACCORDION services and executing the suitable recovery actions (migration, scale up/down, etc.)

2.2.2 ACCORDION Network Resource Orchestrator (NRO)

The second component of AIO, Network Resource Orchestrator (NRO), is responsible for managing the network resources such, as latency and bandwidth, and allocate to ACCORDION application and services according to their network requirements. The role of this component is to provide a multi-domain AI-based orchestration framework of the network by ensuring reliability and latency. It considers AI-techniques (reinforcement learning and Federated learning) to automate management operations of the network slices and facilitate their lifecycles management.

2.3 Resilience policies & mechanisms

The Resilience policies & mechanisms component provides a proactive fault tolerance mechanism using continuous resource usage features by the resource monitoring system. Next, a pre-process and a timely analysis takes place for each hybrid edge minicloud in order to reveal at run-time, potential QoS degradation. An innovative hybrid Bayesian Evolutionary Optimization algorithm is used to make a smart search in the hypothesis space and converge to a close to optimal Time Series Deep Learning topology. This model can evaluate if the deployed resources cannot satisfy the increasing workload and resource utilization on the next time steps and if the resource orchestrator should trigger mitigation policies such as hot- and cold-migration between neighbouring hybrid edge miniclouds and processing edge nodes.

2.4 Security-enhanced application development & deployment

The main objectives of this component are to provide technologies and tools applicable to the Edge/Cloud continuum framework to improve the security of CI/CD development pipelines in a Security by Design approach. The tools include services for Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST), static container security testing and finally secret management services.

2.5 Privacy-preserving mechanisms

The main goal of this component is to build the necessary mechanisms to preserve the users' privacy within the Accordion infrastructure. This component is responsible for (i) the secure and private execution of containers, (ii) the creation of privacy-preserving machine learning models and (iii) the detection and prevention of user data leakage.

2.6 Intelligent edge/cloud continuum orchestrator

The ACCORDION Intelligent Orchestrator (AIO) is the core component to manage the “continuum” of resources belonging to Edge Miniclouds (EM) or public Central Clouds (CC). The main objective is the delivery of allocation plans that matches applications requirements (e.g., in terms of QoE/QoS, etc.) according to the resource conditions and that adapts the network capabilities that ensures reliability and latency requirements. The process followed in this year is the definition of a baseline AIO mainly with the computational resource orchestration and network orchestration. More detailed descriptions are in Section 4 and Section 9. The other components that would help the AIO will be integrated smoothly during the next steps since the system is complex without adding any additional feature.

3 ACCORDION Intelligent orchestrator

3.1 Description and objectives

The ACCORDION federation is composed of a "continuum" of resources belonging to Edge Miniclouds (EM) or public Central Clouds (CC). This continuum enables the convergence of heterogeneous infrastructures, stretching all the way from cloud to edge devices, including intermediate steps such as ISP gateways, cellular base stations, and private cloud deployments. The ACCORDION Intelligent Orchestrator (AIO) component is at the heart of the ACCORDION platform.

The role of the AIO is to manage all types of resources of the ACCORDION Federation, to allow their allocation to the applications running on the Federation. Ultimately, the actual objectives of AIO are to deliver allocation plans matching application requirements (e.g., in terms of QoE/QoS, etc.) while at the same time achieving an efficient exploitation of these resources. Such, potentially conflicting, objectives are particularly complex to achieve in the highly dynamic, distributed and heterogeneous environment, targeted by the ACCORDION project. In order to be properly instrumented to manage such a complexity, AIO is a complex service. It has a compound and hierarchical structure suitable to offer solutions that are tailored to a specific type of resources, while providing a flexible tool enabling the definition and enforcement of high-level management policies across the entire continuum of resources.

AIO achieves its goal by organizing its activity in two major steps: (i) end-to-end cloud/edge selection, and (ii) domain specific allocation. The first step is aimed at identifying the collection(s) of resources that are most suitable to host (portions of) the application in input. The second step is devoted to the actual selection of resources to host the artifacts composing the (portion of) application that has been assigned for execution to a specific collection.

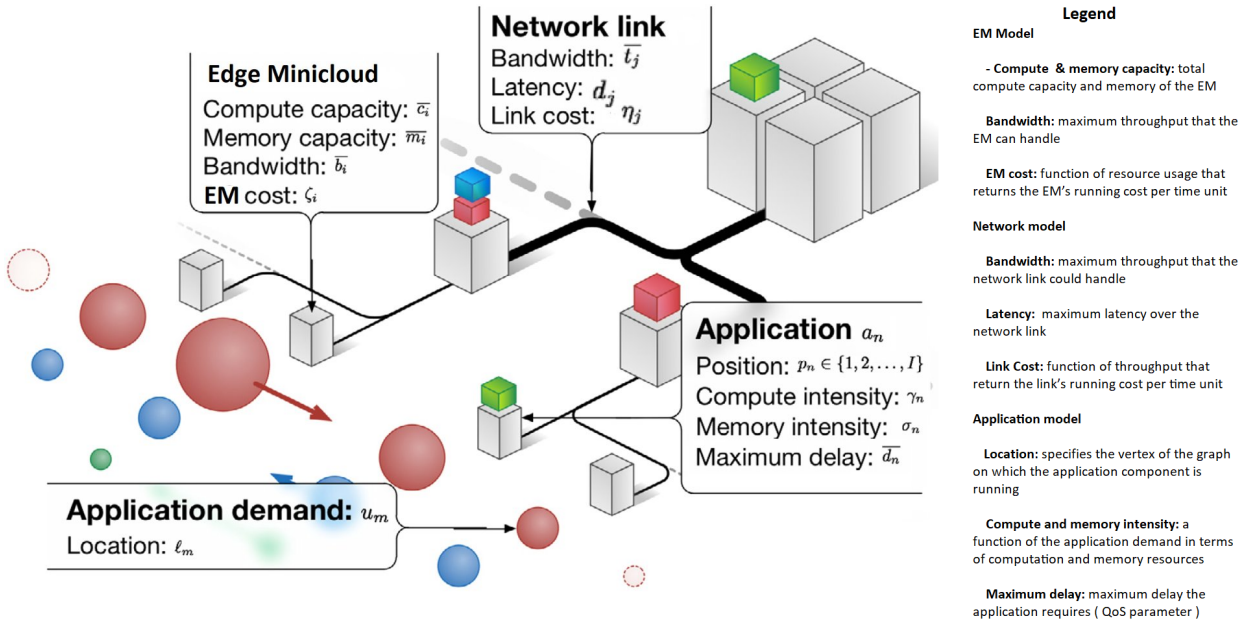

Figure 1:Conceptual representation of EM, Network and Application models

3.1.1 A Blueprints for Applications, a model for Resources

AIO takes in input applications descriptions, each organized in the form of a *Blueprint*. The blueprint defines all the entities associated with an application: the modules composing the applications and the links among these modules. To each entity are also associated its own specific requirements, e.g., the requested network bandwidth, the computing capacity, specialized hardware devices, etc. Additional requirements can be also specified in the blueprint, e.g., which part of the application should be executed at the edge, maximum network delay that an application can tolerate.

AIO builds an internal representation of the resources belonging to the ACCORDION federation. A definitive formal version of such representation will be defined later in the project. The work conducted so far has been based on a conceptual representation that considers the following items, which is represented graphically in figure 1:

3.1.1.1 COMPUTE RESOURCES:

- **Compute capacity**: total compute capacity of an EM or CC
- **Memory capacity:** total memory capacity of an EM or CC
- **Bandwidth**: maximum throughput that an EM or CC can handle

- **Cost**: function of resource usage that returns the EM or CC running cost per time unit

3.1.1.2 NETWORK RESOURCES:

- **Bandwidth**: maximum throughput that the network link could handle
- **Latency**: maximum latency over the network link
- **Link Cost**: function of throughput that return the link's running cost per time unit

ACCORDION Intelligent orchestrator achieves its goals finding proper allocation schemas for the application blueprint by leveraging the information collected on the available resources belonging to the ACCORDION federation. To this end, AIO provides specialized approaches and modules that are differentiated on the basis of the resources actually exploited. In fact, it is composed of two main modules: the **ACCORDION Compute Resource Orchestrator** (CRO) and the **ACCORDION Network Resource Orchestrator (NRO)**.

3.1.2 ACCORDION Compute Resource Orchestrator (CRO)

ACCORDION Compute Resource Orchestrator, CRO, is the module of the Intelligent Orchestrator that focuses on the assignment of applications to the computational resources belonging to the ACCORDION federation. CRO defines an internal representation for the Federation resources and an application blueprint internal representation. Such representations are needed to approach the problem of allocating application to resources as a **Multi-resource Multi-Objective generalized assignment problem (MRMOGAP)**. To solve the MRMOGAP, various types of algorithms have been investigated to establish a trade-off between the execution complexity of the algorithm and its precision to find a suitable optimized deployment solution. Such approach defines multiple objectives, i.e., reduce latency between application components or with the end-user, reduce resources utilization, reduce costs, etc...

Another function of the CRO is to proactively avoid the degradation of the QoS/QoE levels of an application by taking into consideration a set of indices. These indices are provided by other ACCORDION components, such as:

- Privacy preserving application component
- Failure detection and recommendation component

Finally, if the degradation of QoS/QoE cannot be avoided, the CRO has to cope with the runtime violation of the QoS/QoE of an application or other disrupting events by preparing and executing a set of recovery

actions, such as the scaling up or the scaling out by increasing/decreasing the amount or size of instances of components.

3.1.3 ACCORDION Network Resource Orchestrator (NCO)

The second component of AIO, the Network Resource orchestrator (NCO), also takes part in the deployment and runtime of ACCORDION applications. NCO assures that the deployment and runtime of applications do not violate the network requirements of these applications. As depicted in figure, this component, ensure two functionalities (i) End-to-End network orchestration and (ii) Domain specific orchestration. The role of this component is to provide a multi-domain AI-based orchestration framework of the network elements by ensuring reliability and latency. This component provides automated orchestration and intelligent management operations and facilitates the life cycle management of the network slices with the aim of rapid slice creation and activation, enabling application developers, Use Case owners (In the case of ACCORDION: ORAMA VR, ORBK and PLEXUS) to define blueprints for their VR/AR ready slices. Furthermore, this component relies on monitoring the network resources at the edge/public cloud for any potential QoS degradation (e.g., congested links, node capacity excess, etc.) and accordingly plan operations to fix these issues (e.g., service migration, remove congested links, scaling, etc.) and guarantee the network requirements of these applications. These events could be provided through monitoring agents deployed on each domain or by the application itself. To enable self-configuration and self-optimization capabilities of network resources, this component considers the exploitation of machine learning techniques and their integration in the ACCORDION framework.

3.2 Requirements

The ACCORDION Intelligent Orchestrator is connected to many different components of the ACCORDION Platform (see Figure 2). Such interactions are required to feed the orchestrator with all the information needed to drive the allocation process. Interactions with components developed by WP3 are needed to receive information on the federation resources. From WP5, AIO components are derived the data related to the applications. Finally, the interplay with the other modules developed within WP4 are aimed at ensuring resilience, privacy and security.

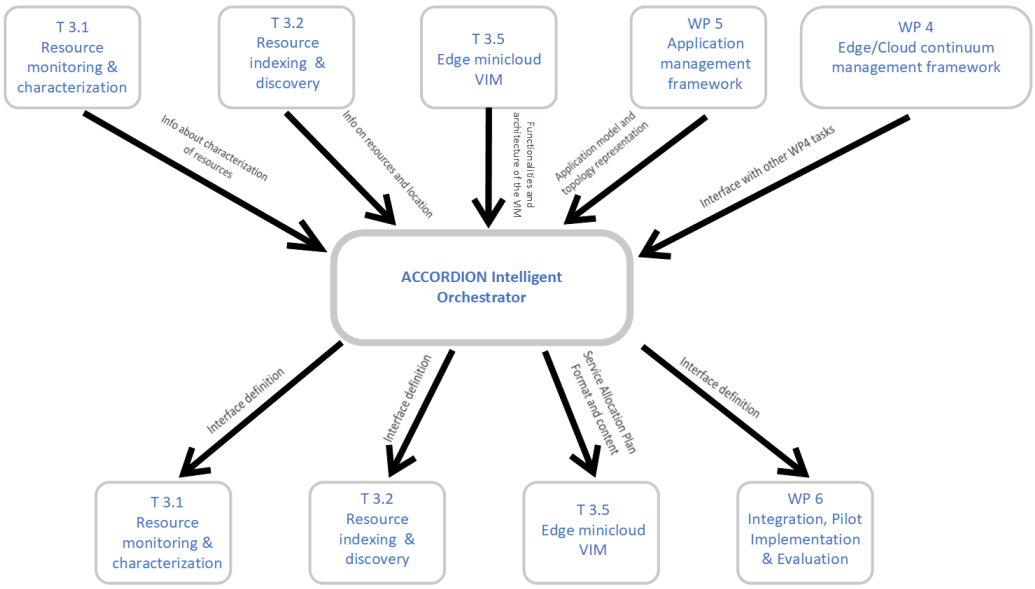

Figure 2: Interactions in terms of requirements and capabilities versus other tasks and work packages of ACCORDION

3.3 Research Challenges & Advancements Achieved

There are several research challenges related to the actual objectives of this component. Some of these challenges are common to the two orchestrators whereas others are specific to computing and network aspects.

3.3.1 Computing Resource Orchestrator for a heterogeneous Cloud/Edge continuum

The computing continuum is, potentially, highly heterogeneous. In a cloud, computing resources are typically provided through virtualization and there is an illusion of infinite resource availability as a consequence of the horizontal scaling. In contrast, in edge computing, computational resources are scarce and then have to be managed very efficiently. Cloud resources are accessible by clients across countries and continents; conversely, in edge computing, a client typically access resources that are under the same network coverage, be it cellular (e.g., 5G) or local (e.g., domestic or office). While the cloud can provide vast computing capacity through elasticity, accessing these resources may involve multiple hops of network communication, leading to prohibitive latency in the processing of client requests. Indeed, one of the main motivations for introducing edge-based computation is to mitigate network latency. So, one of the first research challenges is how to design the architecture of the CRO such as it could manage such heterogeneity in resources and could be

adapted indifferently to EM and CC, taking in consideration their different peculiarities, in particular the limited resource capacity of Edge cloud environments.

To maintain a global vision over the entire set of resources of the federation and of the positions of all the application components, the CRO is logically view as **centralized**, but the potentially large size of an ACCORDION Federation and the number of resource management parameters render a fully centralized resource allocation strategy practically infeasible. A distributed design for the orchestrator will provide some clear advantages, for example:

- **Fault-tolerance:** a decentralized solution is more resistant to failure with respect to a centralized one
- **Increase scalability:** a centralized solution does not scale respect to the size of the ACCORDION federation
- **Reduce network congestion:** a centralized solution could be a bottleneck for network traffic

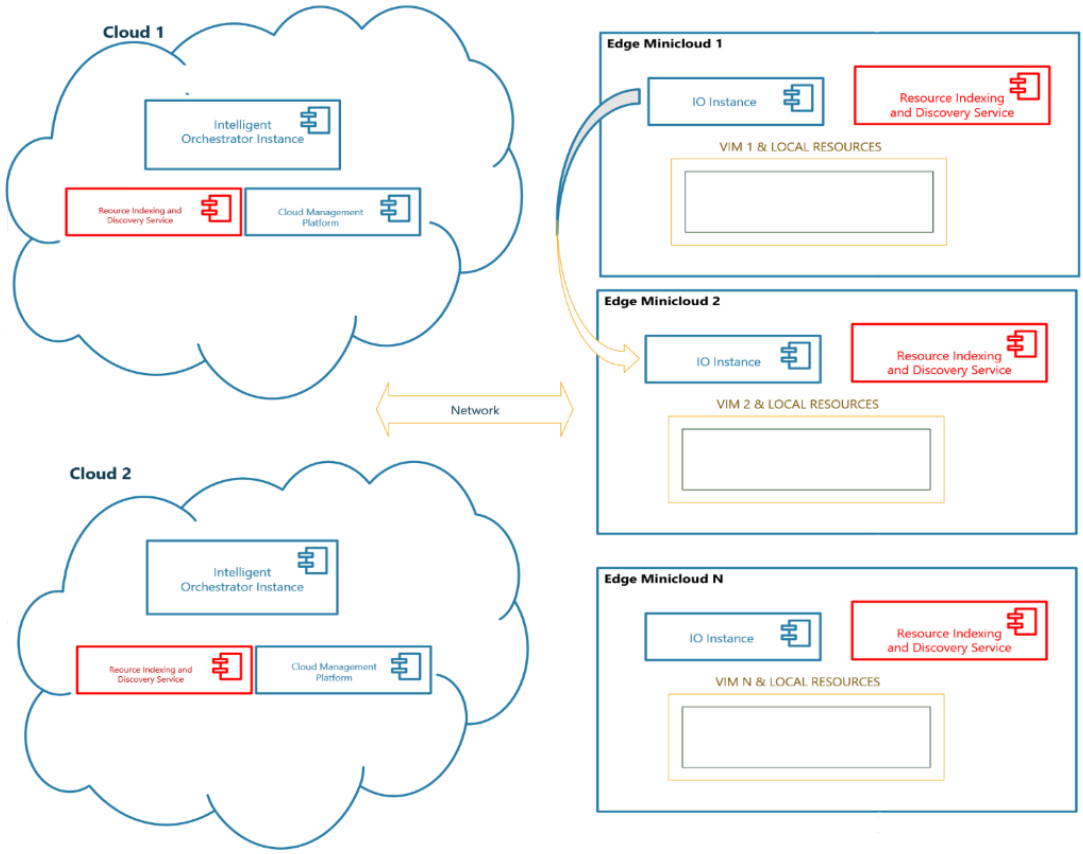

Figure 3: Decentralized solution for the CRO for an ACCORDION Federation with 2 CCs and {1..N} Ems

Such distributed design is shown in Figure 3. For these reasons, the CRO is logically separated in two components: **i)** a Domain Specific Compute Resource Orchestrator (DS-CRO) component, and **ii)** an End-to-End Compute Resource Orchestrator (E2E-CRO) component.

The E2E-CRO component is a centralized component, deployed in a unique instance in a Central Cloud of the Federation. The DS-CRO component is instead a domain specific component which is in charge of performing a matchmaking within the application resources request, its QoS/QoE indicators and the availability of the resources of a specific EM/CC domain. The DS-CRO component has a full and detailed knowledge of these resources, their availability and their localization.

To realize this matchmaking, the DS-CRO needs the specification of the application blueprint as well as the representation of the resources and their actual availability; due to the high dynamicity of the environment and the high volatility of the availability of the resources, DS-CRO queries the Resource and Indexing service component of ACCORDION to obtain information about resource availability in an online fashion. The application blueprint instead is retrieved from the Application Registry.

The DS-CRO then approach the MRMOGAP problem; the multi-resource generalized assignment problem is encountered when a set of tasks have to be assigned to a set of agents in a way that permits assignment of multiple tasks to an agent subject to the availability of a set of multiple resources assigned to that agent, as it happens with our orchestrator.

The MRMOGAP could be modelled mathematically in various ways: however, in any case, finding its solution can be very complex from a computational perspective, being a well-known NP-hard problem.

The models that have a prominent role in literature are:

- The Mixed Integer Linear Programming model [1]–[4]
- The Mixed Integer Non-Linear Programming model[5]
- Graph models[6], [7]

As described in [1], at the DS-CRO level each vertex of the graph is an EM (Edge Minicloud) or CC (Central Cloud) able to host applications using a finite set of resources. Each EM is modelled as a single administrative domain composed by a set of hosts, that can run one or more virtualized resources (VMs for CC and Containers for EMs) depending on their availability of resources, but the view of E2E-CRO over the set of resources of each EM and CC is limited to aggregated values. Examples of such information are:

- Number of CPUs
- Total amount of RAM
- Total amount of Storage
- Presence (or not) of a determined type of a special resource, such as a particular type of GPU.

Furthermore, each CC/EM specifies its cost model, to allow an evaluation of the overall cost to deploy an application, and to minimize the use of resources (if specified in the Objective function). We consider providers adopting a per-resource cost mode: in the per-resource case, we model EM/CC providers as capable of running each type of VM/Container provided by customers, up to availability of resources limit, and charging them per unit of used resources over time. For this reason, we introduce a set of variables $\mathbf{c_{ir}}$, which is the unit cost for the resource **r** of the EM/CC associated to the vertex of index **i**.

The logical network links connecting CCs and EMs are represented as edges of the graph. So far, we worked under the assumption to work with undirected edges, but it is also possible to model the directed edge case, by assigning the proper values to the resources in the two directions. Examples of such network resources are:

- Aggregated maximum latency over the link
- Aggregated maximum bandwidth estimations over the link
- An indicator for the level of congestion of the link, in percentage and normalized to 1: **LC**, $0 \leq LC \leq 1$
- Other similar indicators

Also, in this case, to each CC/EM is associated a cost for the communications inter-providers. Such value could be a function of the different cost specified by the providers at the vertices connected by a given edge. The representation graph is updated exploiting the information obtained by the Resource and Indexing service as well as the Monitoring and Network.

At the DS-CRO level, each vertex of the graph represents a single host of an EM/CC, while each edge represents the internal network connections between hosts of the referred EM/CC

The application, which is structured accordingly with the model defined by Task 5.1, is logically represented with an **application blueprint** annotated by the set of requested resources, QoS and QoE constraints.

The following are the classes of algorithms used to solve the MRMOGAP service allocation problem which had been investigated so far:

- **Exact algorithms**: these algorithms produce an exact solution to the problem. We will not take them into consideration for the ACCORDION since they are too expensive in terms of computation complexity.
    - CPlex
- **Approximated algorithms:** these algorithms produce a near-optimal solution with a limited computational effort. This is achieved by exploiting heuristic, random processes or other types of techniques to reduce the available solution space.
    - **Centralized:**
        - Simulated annealing: the method models the physical process of heating a material and then slowly lowering the temperature to decrease defects, thus minimizing the system energy. At each iteration, a new point in the reference space is randomly generated. The distance of the new point from the previous point is based on a probability distribution that is proportional to the simulated temperature. The temperature progressively decreases from an initial positive value to zero. Then, it measures the quality of the solution, and moves to it according to the temperature-dependent probabilities of selecting better or worse solutions, which during the search respectively remain at 1 (or positive) and decrease towards zero. The algorithm accepts all new points that foster the achievement of the objective, but also, with a given probability, those points that move away from the objective, avoiding being trapped in local minima.
        - Iterative local search[1]: the method tries to apply a local search routine, starting from a candidate “local” solution, to find a better “neighbour” solution heuristically, exploiting only local information to the set of neighbours. If no solutions improving the current one are found, to avoid being trapped in local minima, the algorithm iteratively applies the local routine from a different local solution, which is decided without taking in consideration the information collected by neighbours.
    - **Distributed**
        - Distributed version of simulated annealing
        - Voting-based consensus algorithms[3]: In this type of algorithms, a set of agents try to find a consensus over a certain decision, applying one of a large set of distributed voting techniques such as the Boyer–Moore majority vote algorithm, which is an exact algorithm, and other approximated algorithms. In case of orchestration, each

agent is represented by an application, which starts a voting procedure with the aim of acquiring the resources needed to deploy its assignment vector, participating to a resource election protocol.

- Hierarchical, two level genetic algorithms[5]: A genetic algorithm is an iterative process, in which each iteration is a generation composed of a population of individuals that represents a candidate solution. Each individual is represented by its own chromosome, which is characterized by a set of variable components known as genes. The quality of an individual in the population is determined by a fitness function the resulting value specifies how an individual compares to others in the population. By applying genetic operators an old generation can evolve into a new one until convergence is reached. In two-level genetic algorithms, two GA are nested to solve an assignment problem: the first level applies a distributed GA at the federation level; the second level is executed locally for each EM.

Graph coloring techniques[7]: In this type of algorithm, a distributed version of the well-known Graph Coloring Problem is applied to a bipartite graph, with the two set of vertices constituted by, respectively, the application components and the EM (or the hosting nodes on EM), solving node load conflicts over resources of EM: each feasible coloring represents a set of no conflicting node.

The output of the DS-CRO is a collection of **SAP** (Service Allocation Plan): each plan, which exact structure and internal data format should be still refined, is composed by a vector of couples (c_i, x_j), where $c_i \in Ap$ is the i-th component of the application Ap, while x_j is a hosting server belonging to a specific domain.

DS-CRO select the best SAP plan, among the ones calculated, during the optimization process: in case a QoS/QoE violation is detected or a disrupting event occurred, e.g., a security violation, a failure of a hardware component or a risk of a privacy data leakage, DS-CRO can consider the possibility of executing another SAP, if available.

When an SAP enactment fails, the DS-CRO could execute a set of one or more recovery actions. The QoS/QoE violation or disrupting event can happened also during the lifecycle of the application, after the deployment has been done with success: in this case, a modification to the executed SAP or the execution of another one could also be triggered and a set of recovery actions could be taken, taking in consideration the failed SAP.

Forecasting techniques to proactively detect failures or QoS/QoE parameters deterioration could be employed which exploit ML/AI algorithms over a set of indicators provided by the proper ACCORDION services; these techniques will be further investigated.

If no SAPs are computed or all the SAPs failed to be executed, the DS-CRO is in charge to try to find additional resources to satisfy the application deployment request from other group of remote resources, belonging to other EMs or CCs, managed by a different DS-CRO.

3.3.2 Network Resource Orchestrator for a heterogeneous Cloud/Edge continuum

Alongside NCO, the main role of task 4.2 is to develop a multi-domain AI-based orchestration framework, dubbed Network Resource Orchestrator NRO, of the network resources by ensuring reliability and latency of ACCORDION deployed services, enabling applications' providers (ACCORDION use cases' owners) to define blueprints for their slices, and facilitating the efficient life cycle management of the slices with the aim of rapid slice creation and activation. This poses several scientific challenges that should be taken into consideration for a reliable and efficient network orchestration.

As depicted in Figure 4 [9], this architecture depicts a Network Slice Instance (NSI) stretched across multiple administrative domains. A NSI consists of many Network Slice Subnet Instances (NSSIs) belonging to a different domain. Each NSSI is formed by a set of slates, where each slate represents an abstraction of specific type of resources (Virtual Network Function (NFV), Compute, Network). The role of this component is to provide a multi-domain AI-based orchestration framework of the network elements by ensuring reliability and latency. As ACCORDION federation is built on top of Edge miniclouds (EM) and public Central Clouds (CC). ACCORDION applications may need resources that are only available only on some EMs due to their limitations in terms of compute resources. Such deployment of applications may be stretched across many EMs/CC, and it may require efficient network management, to provide the required bandwidth and latency. This component ensures two functionalities **(i)** End-to-End network orchestration and **(ii)** Domain-specific orchestration.

3.3.2.1 E2E NETWORK ORCHESTRATOR (E2E-NRO)

An end-to-end (E2E) orchestration provides a service delivery all the way from a service provider to the end users. This orchestration stretches across different domains (Public cloud, Mini Cloud) unifying various network layers and heterogenous technologies enabling an overlaid layer providing an efficient convergence

of networking and services [8]. An E2E-NCO is responsible for defining the slice blueprint in terms of network resources.

3.3.2.2 DOMAIN-SPECIFIC NETWORK ORCHESTRATOR (DS-NRO)

After defining the slice blueprint, E2E-NCO issue a request from the corresponding DS-NROs. Each DS-NRO, on his turn forwards the request to the corresponding entity (e.g :NFVI controller, Domain-specific SDN controller) to handle the requests and allocate the resources.

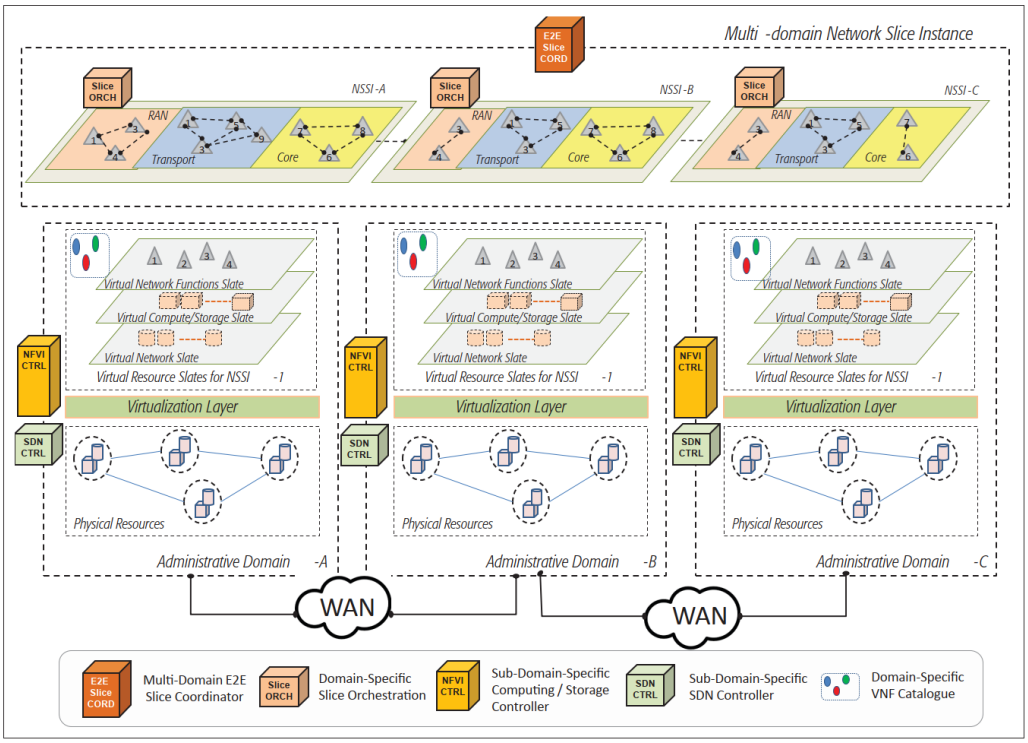

Figure 4 Federated network slice across multiple administrative domains[9]

ACCORDION use cases' applications (ORAMA VR, ORBK, Plexus) have strict network requirements in terms of bandwidth and latency. NRO relies on closed-loop automation to ensure the applications' KPIs.

A network slice is composed of several VNFs interconnected and optimally placed to meet specific KPIs defined by the application provider (latency and bandwidth). A key challenge for developing this component is how to map an application blueprint as described by the application developer (Component developed by task 5.1 will carry the definition of the application blueprint) to a concrete slice blueprint that defines the compute and network resources, and network functions[10].

Identify the infrastructure-domains with the required resources that can meet the application requirements is also challenging. Most importantly, ACCORDION use case applications require strict KPIs in term of bandwidth and latency, and compute resources which could be accommodated by creating customize slices across the multi-domain architecture (Public Cloud, Edge mini cloud) that could be composed of several network instances where each slice is on its turn composed of network resources and network functions.

Another challenge is to design network that can optimize resources' allocation and ensure the lifecycle management of NS (instantiation, runtime, resources availability, slice scaling). Federated learning is considered to be a game-changer for an intelligent and efficient resource management. FL is a machine learning technique that enables the distribution of the learning process among different Edge nodes (e.g : Edge Mini Clouds) for two main reasons 1) ensure privacy and 2) reduce latency. without the need of sharing local data, these nodes will collaborate by sharing their models to a central node to train their model. This latter, the central node, will learn from these models in order to help improving their local models [11]. However, due to the limitation of resources of MC (Compute and network), this is very challenging to leverage intelligence at the edge.

3.4 Technical Challenges and Mitigation (if applicable)

When a DS-CRO fails to execute an SAP during its deployment, or when a recovery action is required, due to a QoS/QoE detected violation or similar disrupting events, it could be not possible to undertake recovery actions limiting the scope of actions to the local resources belonging to an edge minicloud. If the "royal road" is to let this violation or issue to emerge to a higher-level (E2E-CRO), we are already investigating the possibility of creating point-to-point decentralized interactions between DS-CROs to address these kinds of issues. Especially when the extent of the violation is limited, to redeploy the entire application, which is often a very expensive action, both in computational and network terms, leading to making the E2E-CRO a bottleneck for the entire Federation. As an example, please consider the need for a migration: in this case a possible solution technique is to contact a selected group of DS-CROs belonging to "neighbours" domains, chosen following certain selection criteria, creating what we called an **Aggregation of DS-CROs.**

The creation and management of these Aggregations introduces a set of new technical challenges:

- *Discovering of "neighbours"*: to solve this problem, it is possible to exploit an Overlay Network, maintained between the various DS-CRO of the Federation. A plethora of technological solutions are available to enable this kind of interaction, which will be investigated to identify some candidate

techniques, such as gossip-based protocols. Another possibility is to create this Overlay Network "on the fly" exploiting the E2E-CRO and its information about the geographical location of the EMs.

- *Management of the Aggregation:* when a set of candidate neighbours DS-CRO are identified, a new MRMOGAP must be solved over the union of all the resources of the DS-CROs of the Aggregation. The new aggregation could be managed by:
    - One of the DS-CROs, chosen with a distributed election algorithm
    - A new ad-hoc DS-CRO, different from all the DS-CRO of the **Aggregation**
    - All the DS-CROs of the **Aggregation,** exploiting a distributed algorithm to coordinate the solution of the MMORGAP
- *MMORGAP resolution process:* distributed algorithms are well suitable to solve this problem efficiently. Such class of algorithm has been briefly explained and analysed in Section 3.3.1. The process for creating an Aggregation could be iterated if the MRMOGAP could not be solved, or the distributed algorithm raises exceptions during execution or for example a set of resources are lost due to network connection failures or disruptive events happened: in this case, it is possible to expand the actual Aggregation with additional neighbours, or create a new Aggregation which is a superset of the old one, creating a hierarchical structure with multiple levels of orchestration. The Aggregation could either persist or be destroyed after the resolution of the application request.

Similar techniques have been also applied in Fog computing for an automatic and spontaneous creation of virtual clusters, similar to our Aggregations, often exploiting algorithms based on the concept of betweenness centrality to measure the centrality of a node in a graph where each vertex represents a Fog device, and techniques such as Spectral clustering, as in SmartFog[12].

3.5 Future Work

In the second cycle of the project, there are many advancements and refinements we plan to develop on the ACCORDION Intelligent Orchestrator, both from an implementation and algorithmic viewpoint. Besides of a well-defined interplay between the ACCORDION Compute Resource Orchestrator and ACCORDION Network Resource Orchestrator, resulting in a fully integrated ACCORDION Intelligent Orchestrator module, we have specific plans for the development of the two orchestrators.

3.5.1 ACCORDION Compute Resource Orchestrator

We plan to enable replaceable E2E-CRO and DS-CRO implementations, easing the integration of alternative approaches and solutions. In particular, it is our intention to investigate ML-based approaches, both at the level of the E2E-CRO and at the level of DS-CRO. Regarding the E2E-CRO, in the next future we will deal with definition of its API to improve the interaction of the orchestrator with the other components of the ACCORDION Platform. In particular, we plan to intensify the joint work with the modules developed within the WP4, the application model (T5.1) and WP3.

We also plan to develop a network interaction protocol, and the relative software modules, between distributed DS-CRO instances to enable a decentralized orchestration of resources between different edge miniclouds.

3.5.2 ACCORDION Network Resource Orchestrator (NCO)

We plan in the second cycle of the project, the following actions:

- Investigate on mapping ACCORDION applications models developed in task 5.1 into network slice configurations considering ACCORDION federation resources and their heterogeneity in term of compute and network resources.
- Collaborate with use case owner to deploy their applications and gather data for training this component models.
- Start designing and implementing both E2E-NRO and DS-NRO and accordingly plan integration them with the other components of the project (WP5 and WP3)

4 Resilience policies & mechanisms

4.1 Description & Objectives

Given that the modus operandi of Edge Computing consists of a vast number of compute nodes operating simultaneously, it is extremely important to regard component failure as an inevitability. By doing so it is possible to ensure that the infrastructure shall keep operating without interruption and Quality of Services (QoS) deterioration. The main way of ensuring that a cloud/edge computing infrastructure will be able to keep carrying out its operation even after the event of a critical failure, is by triggering migration policies and utilizing backup components, which automatically replace the failed ones in a manner which guarantees the QoS.

The migration and the reactive creation of a new Virtual Machine (VM) requires a certain waiting period, which would provoke QoS degradation. Thus, fault tolerance should be achieved by following a proactive approach. At any given time, the network should contain a specific number of virtual computational nodes, which remain idle until one of the already working components ceases to function properly. Given that additional Cloud-based computational nodes come on demand, it is important to keep this number to a minimum. By utilizing machine learning algorithms, it is possible to extract information regarding the behaviour patterns of the services and the faults that occur. This enables the fault tolerance functionality to take place in a manner which will ensure that the networks operations will continue to take place uninterrupted and that the overall cost will be kept to a bare minimum.

The Resilience policies & mechanisms (RPM) component provides a proactive fault tolerance model using data features related with the resources usage in a highly geographically distributed cloud environment. The Resource utilization aware Fault Tolerance (FT) mechanism monitors the resources usage that runs on each hybrid edge minicloud in order to reveal, at run-time, potential QoS degradations. In case that the deployed resources cannot satisfy the increasing amount of resources usage on the next predicted time steps, then the middleware will trigger mitigation policies such as hot- and cold-migration between neighbouring hybrid edge miniclouds and processing edge nodes.

In case that the Accordion platform will cover highly geographically distributed areas and the edge devices that generate tasks will be moved out of the coverage of the Edge infrastructure or there will be a high distribution of devices around a specific Point of Interest (PoI), then the FT mechanism will perceive an increase in the workload, predict potential QoS deterioration and trigger proactive measures [12], [13].

4.2 Requirements

This Proactive Fault Tolerance mechanism leverages historical monitoring data decoupling the prediction from the application implementation and the resource characteristics. The Resilience policies & mechanisms (RPM) component provides next time-step predictions based on the current and previous resource usage metrics. The quality of training data is the main requirement of the proposed approach. In order to provide accurate predictions, the input observations should have the same statistical properties with observations of the training data. This characteristic exists in all data driven methods.

The RPM component has the requirement to communicate with the resource monitoring and profiling mechanism that will be implemented in the Task 3.1 in order to receive timely and accurate information about the performance of the processing edge nodes and the VMs. In addition, historical data are gathered and stored in a time series format and they will be used for the training of the RPM. The RPM component will also communicate with the Intelligent, adaptive resource orchestration component of the Task 4.1 in order to inform it regarding the potential QoS degradation and the proactive VM migration that should take place.

4.3 Research Challenges & Advancements Achieved

The RPM component uses a resource utilization predictor [14] in order to predict the potential QoS deterioration and trigger proactive measures and intelligent replication. The resource utilization predictor involves two pipelines as depicted in Fig. 2. The first pipeline as depicted in Fig. 2a is the training process which builds a multi-output regression model based on historical data. Specifically, the edge devices profile the resource usage during the execution of the deployed service. The profiling output contains CPU, RAM, Disk and network parameters retrieved by the Resource monitoring mechanism. The profiling dataset is pre-processed using a normalization method and piped to the hyper-parameter optimization process which trains multiple neural networks in order to identify an optimal topology.

The second pipeline, Figure 5b, is the Inference process which takes the current status of resource usage, after it is pre-processed like the training dataset. Afterwards, the Deep Learning (DL) model provides a resource utilization prediction for the next timestep and QoS deterioration. Proactive FT measures can be triggered in order to mitigate the QoS deterioration and specifically the latency in tasks executions based on an optimal and cost-effective usage of the Edge resources.

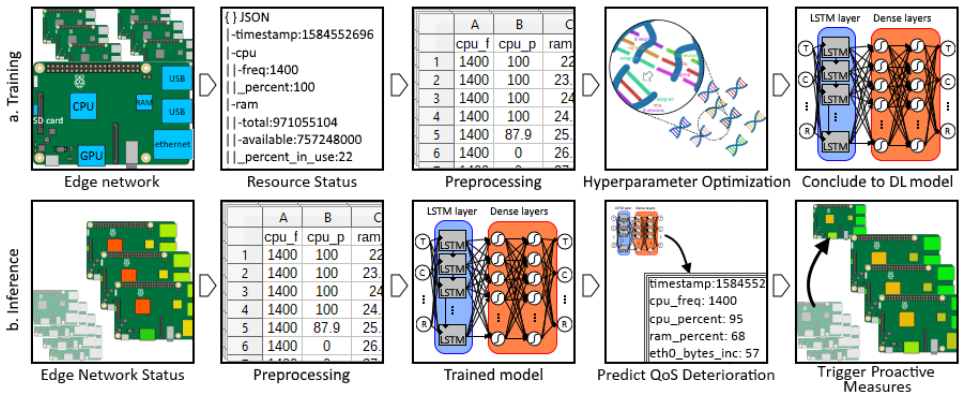

Figure 5 :Training & inference of the proactive FT mechanism

The sequence of resource utilization metrics exhibit predictability as we have seen in our experiments. These metrics have high serial and cross correlation values making rational a time series approach. A Recurrent Neural Network (RNN) multi-regression model which leverage time series characteristics can be a prominent solution for the prediction of these resource metrics. Time sequences can be effectively modelled and be forecasted by Recurrent Neural Networks (RNN) and one dimensional Convolutional Neural Network (CNN) as they will be presented in the next paragraphs. The following paragraphs will be present the different DL layers that can be used in time series problems. The main DL architecture decisions and systematic methodologies in order to find a close to optimal DL architecture with evolutionary computation algorithms.

4.3.1 LSTM Neurons Architectural Usage

Long short-term memory (LSTM) is a type of RNN which contains memory blocks in the recurrent hidden layers. Specifically, LSTM units include the cell, the input gate, the output gate and the forget gate, which ensure the storage and access in previous data, even after the passage of large temporal periods. Thanks to its ability to capture long-term temporal dependencies, LSTM is suitable for time series problems. It is also capable of tackling two common problems of RNNs: the short-term memory and the vanishing gradient problem. LSTM is a non-linear time series model that allows information to persist, using a memory state, and is capable to learn the order dependence between observations in a sequence. LSTM originates from RNN but outperforms it by resolving two weaknesses: the vanishing gradient problem and the short-term memory. The former makes RNN earlier layers incapable of being trained sufficiently. The latter makes RNN incapable of carrying information from earlier time observations to later ones and, as a result, RNN forgets faster.

4.3.2 One-dimensional Convolutional Neural Network

A Convolutional Neural Network (CNN) is a type of feed forward neural network which is suitable for processing data which have a grid structure. Its architecture usually consists of one or more convolutional layers with subsampling stages and one or more fully connected layers as it happens in common multi-layer neural networks. CNNs use an optimizable feature extractor named kernel, which enable CNNs to easily memorize spatial order of features. The latter helps CNNs to perform an important accuracy and efficiency. A convolution is performed between the data matrix and the kernel matrix, with the convolution window moving **n** elements for each multiplication, **n** parameter called strides. After the convolutional layer, a common practice is to add a pooling layer, either a max or an average one. The purpose of the pooling layer is to reduce dimensionality and thus the computational power required to process the data, as well as to supress the noise. Following the convolution and pooling layers, the data is flattened, so it can be used as input to a fully connected feed forward network, which can help learn nonlinear patterns and features.

In the case of resource usage prediction, as we can see in Figure 6, CNNs featuring one-dimensional convolutional layers with max pooling layers[15]. The input data comprises of a time-series that show the percentage usage of every resource available to our system, such as cpu, ram, bandwidth. The notion is that one-dimensional convolutional layers will help find patterns that refer to the usage of our resources, performing the convolution on the time dimension. Particularly, after splitting our dataset timeseries into its train and test sets, we transform it into samples of shape (n, features). This is fed into our network, which performs one dimensional convolution on each sample, the idea being that resource usage of a specific timestep has a much higher correlation to more recent measurements of the resource usage compared to earlier ones. This way the network is trying to predict the feature values of the next timestep using the last n values of the dataset.

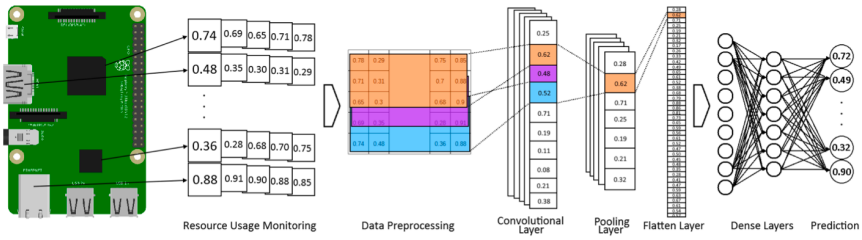

Figure 6 :Deep learning CNN model leveraging sequence of resource usage metrics

4.3.3 Regularization

The DL model should be regularized and generalized for new instances. One common practice to achieve this is the Dropout technique, in which a percentage of the recurrent connections are excluded from activation during each learning iteration. Early stopping is a regularization strategy in which we evaluate the model in each training iteration and when the accuracy is decreased in the testing dataset, the training stops.

4.3.4 Learning Process

The parameters of an LSTM-RNN can be optimized by minimizing an objective function. The objective function expresses the difference between the predicted output and the actual output such as Mean Absolute Error (MAE) L1 and Mean Squared Error L2. Objective functions can also be combined with penalization weight techniques in order to regularize the model. Based on our experience the most widely used optimization techniques for time series are the Root Mean Square Propagation (RMPSrop) and the Adaptive Moment Estimation (Adam). RMSProp and Adam are both Stochastic Gradient Descent approaches with an adaptive learning rate. RMSProp uses the Momentum approach, in which the gradient in every iteration is the sum of the current gradient and the previous gradients, which results in restricting the oscillation. Adam, similarly uses the Momentum technique, but it also finds an invariant direction of slope in contrast to the other oscillating directions, with the navigation through saddle points.

4.3.5 Architecture Design

The main DL architecture decisions namely hyper-parameters are the number and type of layers, the number of units in each layer. In addition, it may also include the activation function type and dropout percentage. In many cases, hyper-parameters are defined based on domain expertise and manual tweaking. A large number of layers and units may increase the capacity and memorization of a model, but it may also decrease the generalization. However, a small number of layers and units decreases the learning abilities of the model. The decision between a large topology that overfits the data and a small topology that underfits is known as the bias - variance trade-off. An automatic, systematic search through architecture space will be performed by the evolutionary computation algorithm such as Genetic Algorithm (GA), Hybrid Bayesian Evolution Strategy (HBES) and Hybrid Bayesian Particle Swarm Optimization (BPSO). These two algorithms, GA and the BPSO, are described in the next two paragraphs. Figure 7 provides the HBESin pseudocode and the Figure 8 provides the output close to optimal topology.

The main steps of Generic Algorithms [16] [17] are summarized as follows. Firstly, we pick a population size (N) and generate said population of Artificial Neural Networks (ANN) with random hyperparameters. Secondly, we pick a number of generations for the algorithm. We iterate over the generations performing the following steps: i) We train and test the entire population and then sort it by the fitness function which is the MSE of the individual ANN; ii) The (B) best ANN advance to the next generation along with some randomly chosen (R), while we are being careful to avoid converging too fast; iii) We generate the remaining (N-R-B) members of the population by picking the hyper-parameters of the new networks choosing from the hyper-parameters of two parent ANN, chosen at random* from (B U R) with the probability of a random hyper-parameter changing to a completely different value (mutation).

The BPSO uses both the PSO [18] and the BO [19] with Gaussian Process for the hyper-parameter selection. The former was responsible for the numerical hyper-parameters and the latter for the nominal. The reason that separation occurred is due to PSO's inability to manage nominal hyper-parameters easily, whereas BO is able to handle both numerical and nominal hyper-parameters. The numerical hyper-parameters defined by PSO are units, layers, dropout rate, lookback, learning rate, epochs, batch size and the number of LSTM/CNN layers. In order to have a complete deep neural network, the building of the network requires optimizers and activation functions for each layer of the network. This process is implemented by the BO which functions inside the PSO algorithm and searches the optimal nominal hyper-parameter set given a numerical hyper-parameter set from PSO. Once the optimal nominal hyper-parameter set is found, the deep network gets trained and returns the loss value from the evaluation method.

Algorithm 1 Hybrid Bayesian & Evolution Strategy

Step 1: Initialization of Evolution Strategy
Set the starting search point of the algorithm. Usually a_1=[0.5, 0.5, ..., 0.5] since we have already scaled our hyperparameter options down to [0,1]

Step 2: for i = 1, 2, ..., n_{pop}:
i)add some random noise to the search point
ii)Scale back from [0,1] to the hyperparameter search space to create the ordinal hyperparameter values for the network to be trained
iii)Bayesian Optimization with GP
1) Apply a Gaussian Process prior on f
2) Observe f at n_0 points according to an initial experimental design
3) Initialize $n = n_0$
4) Repeat while $n \leq N$
a) Update the posterior probability distribution on f using all available data
b) Let x_n be a maximizer of the acquisition function over x.
c) Observe $y_n = f(x_n, x_i(t+1), v_i(t+1))$
d) $n \leftarrow (n+1)$

Step 3: Sort the results and its corresponding hyperparameters

Step 4: Calculate the new search point by averaging the points of the top_n networks

Step 5: Go to Step 2 until desired number of iterations is completed

Figure 7 :Smart search in the hypothesis space

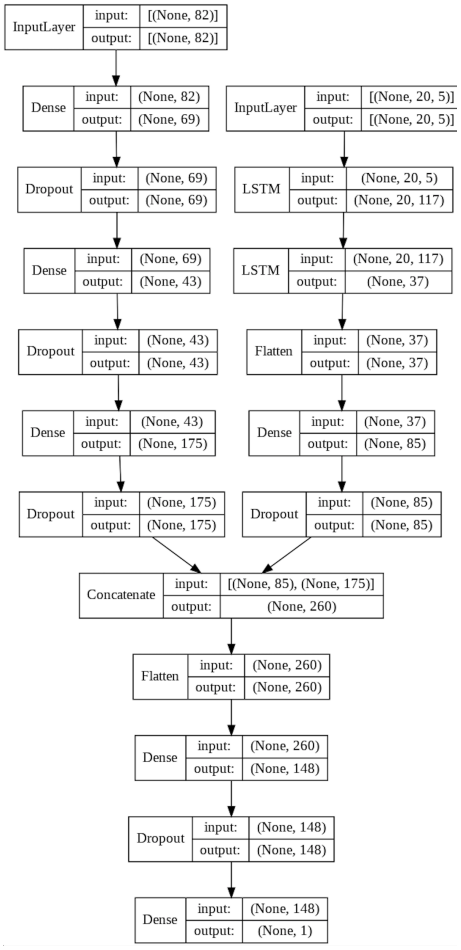

Figure 8: Close to optimal DL topology

4.3.6 Evaluation of the proposed RPM model

The RPM proposed model made a smart search in the hypothesis space and concluded to the close to optimal architecture depicted in Figure 8. This architecture depicts two towers. The first tower captures the global status of the processing edge nodes. This means that it perceives the information of the entire Edge infrastructure. The second tower which leverage the LSTM layers, focuses on a specific processing edge node. This type of architectures is used for every processing edge nodes in order to capture the local and the global status.

The proposed RPM model is compared to three state of the art predictors, AUCROP [20], XGBoost [21], Auto-sklearn [22]. AUCROP is the abbreviation of Application and User Context Resource Predictor which is a meta-model predictor which utilizes common machine learning algorithms for resource usage predictions. Auto-sklearn is a general-purpose automated state of the art machine learning meta model used for data

pre-processing, regression and hyper-parameter tuning through the Bayesian Optimization algorithm. Its efficiency is attributed to its possibility of saving previous optimization runs which provides the opportunity for the training of data with previously saved settings. Auto-sklearn has won the prestigious ChaLearn AutoML challenge, it is widely usedin research papers and it is considered as one of the best AutoML frameworks by the data scientist community.

XGBoost is an open-source software library which provides a gradient boosting framework and is often used for the Gradient boosted decision trees. Its availability in several programming languages and its integration in Python's libraries, such as Scikit-learn, have contributed to its popularity. XGBoost has been used a lot in Kaggle and KDD Cup winning solutions. The GA-LSTM, HBPSO-LSTM, and BPSO-CNN models are our resource usage prediction models in edge devices that leverage evolutionary computation algorithms for the estimation of the hyper-parameters and a close to optimal use of the , feedforward, LSTM, and CNN layers.

Method	RMSE	MAE	CPU-1 RMSE	CPU-1 MAE	RAM-1 RMSE	RAM-1 MAE	Train Time	Infer. Single Time	Infer. Batch Time
BPSO-CNN	0.0631	0.0254	15.932	12,898	1.416	0.587	1692	0.036	0.039
BPSO-LSTM	0.0661	0.0276	16.088	12.691	1.479	0.613	1042	0.034	0.035
AUCROP	0.0814	0.0414	17.235	14.009	2.480	1.482	522	0.004	0.011
XGBoost	0.1139	0.0599	16.457	13.569	1.515	0.472	181	0.060	0.010
Auto-sklearn	0.1055	0.0243	52.659	17.856	1.546	0.526	1338	0.263	0.572
GA-LSTM	0.0674	0.0338	16.099	12.838	1.746	0.917	574	0.020	0.024

Table 1 : Evaluation of the prediction of the heterogeneous edge resource utilization

The proposed RPM model uses the out-of-sample evaluation technique which preserves the sequence of the observations. Therefore, we did not apply any shuffling method in the context of time-series applications. We split the dataset into two parts, the training sequence which was the 66% of the observations and the testing sequence which was the rest 34% of the data. The proposed RPM model was evaluated with the evaluation metrics Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). We recorded the training time and the inference times for a single prediction and for a batch of 100 predictions. We conducted experiments of the resource utilization of edge devices with a dataset constructed by Raspberry pies. Table 1, shows the evaluated results.

In order to make a large-scale simulation of the proposed RPM model in an edge computing environment, we set an experiment scheduled to last one week in which generated and offloaded tasks are close to 1,500,000 tasks. The task generation used a sum of Gaussian distributions in order the workload to model the working behaviour of employees who have peak of service requests at 11:00 in the morning. The number of the main processing edge nodes, namely hosts, is five and there are 15 more backup hosts in which potential fault tasks can be replicated. When a QoS deterioration is predicted then the intelligent VM replication takes place proactively in order to service the fault tasks. The predictions of the RPM model have a time frame of 10 minutes. We compared - the reactive baseline FT mechanism against the proactive Intelligent FT mechanism and we see an improvement in the tail latency of task execution from 61 sec to 2 sec. The tail latency measure is the 98% percentile of - execution time and it is a representative measure for the QoS deterioration. The simulation took place with CloudSim Plus and the Table 2 summarizes the results. The execution time values and latency are in sec.

	Number of tasks	Tail Latency	Avg. Exec. time	Std. Exec. time	Median Exec. time	Min Exec. time	Max Exec. time	Skewness Exec. time	Kurtosis Exec. time
Baseline	1,530,565	61.127	4.131	18.348	1.109	0.500	234.559	7.552	62.053
Intelligent	1,534,531	2.250	1.195	1.266	1.109	0.500	94.439	42.158	2229.162

Table 2: Evaluation of one week simulation of the proposed Fault Tolerance mechanism for QoS deterioration

4.4 Output description

The RPM component will communicate with the monitoring system as described by the Task 3.1 Resource monitoring & characterization and the Task 4.1 Intelligent, adaptive resource orchestration in order to inform it about the potential faults and trigger proactive measures. The RPM input is illustrated below in Table 3 .

```
nodeID: RP1,
network: [data],
CPU: [data],
memory: [data],
readLatemcy: [ {device:sda, data:[data]}, {device:sdb, data:[data]}],
writeLatency: [ {device:sda, data:[data]}, {device:sdb, data:[data]}],
diskTime: [ {device:sda, data:[data]}, {device:sdb, data:[data]}],
....
```

Table 3: RPM input from resource monitoring

Where data is the current value of the processing edge node in the inference stage, while in the training stage data is a time series with sequences of {time:timestamp, value: datavalue}. The output of RPM, as depicted in Table 4, will contain for each nodeID the probability to have QoS deterioration, if it requires an intelligent replication and the predicted resource usage. The RPM component will exchange JSON files with the abovementioned ACCORDION components. It will also keep some CSV and Keras H5 files for internal training. The API communication will be a REST API implemented with Flask.

nodeID: RP1, qosDeterioration: value, CPU: value, memory: value, network: value, ...

Table 4 : RPM output

4.5 Future Work

The Time series Multi-output regression deep learning model follows a supervised ML approach. Supervised ML models can provide accurate intelligent decisions from the beginning of their operation, but they are application specific and require historical training data. Deep Reinforcement Learning (DRL) approaches suffer from the cold start problem but during their interaction with the Edge computing environment, they learn and improve their decisions. The RPM component will include a generic supervised model as described in the previous subsections and a Deep Reinforcement Learning. In the beginning of the RPM operation the supervised ML model will provide the intelligent replication decisions and the DRL will learn from the environment. When the DRL will be able to provide better predictions of the potential QoS deterioration than the generic supervised ML model, it will be selected for the continuation of triggering the proactive measures. The propose, implementation and evaluation of the DRL is a work that will take place in the next months.

Currently we have evaluated the FT mechanism with QoS and ML evaluation metrics. We plan in the next months to provide evaluation with the following metrics:

Appendix A. Mean time to failure: The expected time to occur a failure given that the edge infrastructure has a normal operation.

Appendix B. Mean time to repair: The expected time to repair the edge infrastructure after failure occurrence.

Appendix C. Mean time between failures: The average time to the next failure.

Appendix D. Reliability: The property that a system can run continuously without failure.

Appendix E. Safety: The situation that when the edge infrastructure temporarily fails to operate correctly, nothing catastrophic occurs.

Appendix F. Availability: The probability that the edge infrastructure is operating correctly at any given timestamp and is available to provide services.

The Accordion engineers and researchers will also work in the next months with evaluation of the RPM with the monitoring data of the use cases and the implementation of the specific APIs of the Accordion architecture.

5 Security-enhanced application development & deployment

5.1 Description & Objectives

The main objectives of task 4.4 are:

- Investigate on specific security topics to support other ACCORDION tasks.
- Identify and assess technologies and tools applicable to the Edge/Cloud continuum framework and to be introduced as DevSecOps techniques to improve the security of CI/CD pipelines in a Secure by Design approach.
- Set up a secret management service to safely handle confidential material in ACCORDION environment, such as private keys, configuration passwords etc.

5.2 Requirements

The requirements of this component can be summarized as follows:

- Provide support to other ACCORDION tasks, in the form of tools, models, guidelines or best practices, to increase the level of security of their results.
- Provide security tools and techniques to the ACCORDION development and deployment pipelines especially suitable for applications and containers.
- Provide a secret management service to other ACCORDION components.

5.3 Research Challenges & Advancements Achieved

Concerning the first objective, two fundamental resources suitable as background references are the annual report on the top threats to Cloud environments published by *Cloud Security Alliance (CSA)* [23] and the CIS Controls Implementation Guidance for the Cloud by the *Center for Information Security organization (CIS)* [24].

The CSA annual report provides an overview of the top cloud security threats based on a periodic survey of about 250 top companies operating in the cloud industry. The list of the top 11 general threats found in the latest (2019) report is the following:

1. Data Breaches
2. Misconfiguration and Inadequate Change Control

3. Lack of Cloud Security Architecture and Strategy
4. Insufficient Identity, Credential, Access and Key Management
5. Account Hijacking
6. Insider Threat
7. Insecure Interfaces and APIs
8. Weak Control Plane
9. Metastructure and Applistructure Failures
10. Limited Cloud Usage Visibility
11. Abuse and Nefarious Use of Cloud Services

For each threat, the report provides a high-level description together with the possible business impacts, examples and suggestion for remediation.

On the other hand, the CIS controls guide uses a different perspective, starting from the security controls description and providing rationales, applicability and considerations on the related threats and why implementing that control is critical. The guide includes detailed description of the following 20 fundamental security controls:

- Inventory and Control of Hardware Assets
- Inventory and Control of Software Assets
- Continuous Vulnerability Management
- Controlled Use of Administrative Privileges
- Secure Configuration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers
- Maintenance, Monitoring and Analysis of Audit Logs
- Email and Web Browser Protections
- Malware Defences
- Limitation and Control of Network Ports, Protocols and Services
- Data Recovery Capabilities
- Secure Configuration for Network Devices, such as Firewalls, Routers and Switches
- Boundary Defence
- Data Protection
- Controlled Access Based on the Need to Know

- Wireless Access Control
- Account Monitoring and Control
- Implement a Security Awareness and Training Program
- Application Software Security
- Incident Response and Management
- Penetration Tests and Red Team

Both documents cover all major Service Models (IaaS, PaaS, SaaS and FaaS) and Deployment Models (Private, Public and Hybrid) and then can be used in the context of ACCORDION as a starting point for the analysis and possibly the implementation (procedural or tool-based) of the most relevant security controls, selected and prioritized according to different metrics including for example risk, impact on performance and feasibility in terms of required resources. Examples include Access Control systems, Hardening, etc. and they will be investigated in future ACCORDION development phases based on specific emerging requirements and priorities.

The second objective is related to DevSecOps technologies and tools applicable in ACCORDION development and deployment pipelines. Several security tools and software have been evaluated adopting specific metrics considered relevant for the ACCORDION project, including license type, capabilities, project activity and maturity, documentation and finally the possibility to be extended. For the evaluation, in addition to the official project's sites and available documentation, the online community *Black Duck Open Hub* [25], which collects relevant information and metrics on a great number of open-source projects, has been consulted as well. The evaluation focused on tools related to three areas:

- Static Application Security Testing (SAST).
  The application's code is scanned while the application is at rest to verify general code quality and searching for potential security vulnerabilities. This technique is limited to the application code and cannot find environment or run time related vulnerabilities. Issues are detected at the early stages of the software development life cycles reducing the overall impact and the cost of mitigation.
- Dynamic Application Security Testing (DAST).
  The application is analysed from the outside while it is running, searching for run time vulnerabilities. Issues found at this stage are generally more expensive to fix.
- Static container image security testing.

The application containers images are statically scanned searching for known vulnerabilities, typically by parsing through image's packages or other dependencies.

Automated static and dynamic application testing combined with container testing techniques allow the detection of a broad range of different vulnerabilities.

A key point for the DevSecOps tools evaluation is the fact that the selected platform for ACCORDION development is GitLab, which natively offers SAST, DAST and container scanning tools that can be added to the CI/CD workflows.

The results of the evaluation and produced following results:

- Proposed software for SAST, two alternatives:
  - SonarQube[26]. A tool providing centralized services for automatic scanning of source code to detect bugs and vulnerabilities. The key feature of SonarQube is its capability of scanning more than 25 programming languages source, including plugin for Shell Scripting code analysis, and it has been considered a key factor for ACCORDION because it allows to use a single security service to analyse the ACCORDION source code artifacts which are possibly developed in different programming languages.
  - GitLab SAST[27]. Natively integrated in GitLab, but some functionalities not available in the free version of GitLab software.
- Proposed software for DAST, two alternatives:
  - OWASP Zap[28]. ZAP allows web applications penetration testing acting as a proxy between browsers and applications, performing both passive and active scanning searching for several known security issues.
  - GitLab DAST [29]. Based on ZAP, not available in the free version of GitLab software.
- Proposed software for Container image security, three alternatives:
  - Anchore[30]. Anchore is a standalone tool providing container image vulnerability scanning and integration in existing CI/CD pipelines.
  - Clair [31] + GitLab. Natively integrated in GitLab, not available in the free version of GitLab software.
  - Clair + Harbor[32]. Harbor is a trusted cloud native repository for Kubernetes that offers many interesting features like private docker image registry, integrated control on image signing process and integrated control on image vulnerabilities.

To put these tools in a context of a generic DevOps process, the following figure, shows the phases where the selected tools should be integrated. Additional details on these tools can be found in deliverable D2.3, par. 5.2.3.

The third objective is related to the analysis of a secret management service to protect sensitive data needed in ACCORDION, such as private keys or security passwords to be used to access services. At this phase of the project, only a preliminary analysis has been performed and the selection is oriented towards the open-source platform *Vault*[33], but additional research is needed.

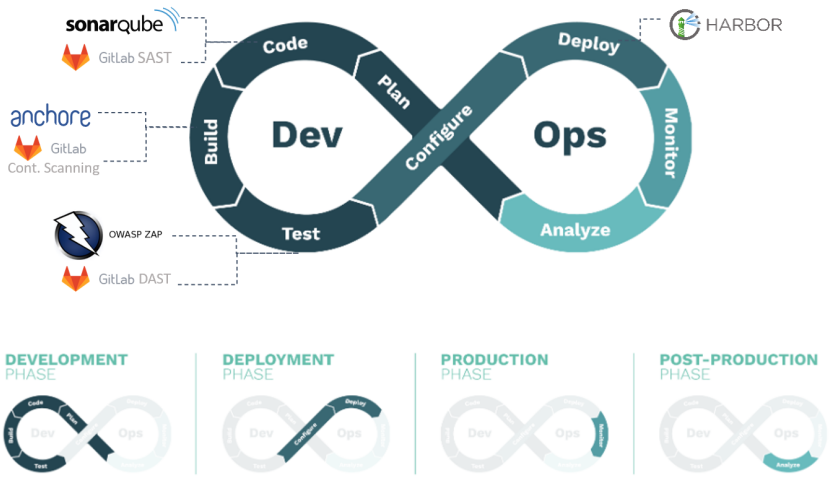

Figure 9: Generic DevOps process and security tools

5.4 Output description

The output of this component is the installation, integration and usage guides of the proposed software and tools, in the form of standalone services or as integrated tools. In this milestone, the SonarQube service for Static Code testing will be provided. SonarQube interfaces and components are described in D2.3, par. 5.2.3.

5.5 Future Work

Future work for next milestones includes:

- Further analysis and finalization on secret management service for ACCORDION
- Installation of DevSecOps selected software described in 8.3
- Support to specific security related aspects possibly impacting other ACCORDION tasks in a case-by-case basis.

6 Privacy-preserving mechanisms

6.1 Description & Objectives

The privacy-preserving component (PPC) is responsible for providing users' privacy at various levels within the Accordion infrastructure. This component will carry out (at least) 3 main functions. First, PPC will provide the mechanisms needed to ensure that containers can be securely and correctly executed on top a network infrastructure with the least number of privileges and without network administrators (e.g., the cloud provider) being able to infer any information about them (e.g., which type of application runs inside them). Second, PPC will enable the generation of machine learning (ML) models that provide high accuracy while at the same time being resistant to attacks that aim to infer private information from them. This task will also involve the design of privacy-preserving mechanisms suitable for federated learning (FL). Finally, PPC will allow the detection of user data leakage to unauthorised third parties by passively or actively monitoring users on devices and user components (e.g., browsers or containers). All these privacy features can either be enabled by default within the Accordion infrastructure or be offered as a service, e.g., to customers who run sensitive workloads.

6.2 Requirements

- Access to a large amount of data (e.g., container images) and ability to monitor components and capture data at various levels within the Accordion infrastructure.
- Understand the overall goals of Accordion, their security and privacy threats and the desired level of security and privacy.

6.3 Research Challenges & Advancements Achieved

Objective 1) Secure and private container execution

1.1 Hardening the security of the host OS kernel

Challenges. Previous work has demonstrated that adversaries who have access to a container can exploit vulnerabilities in the host OS kernel to trigger memory leaks, write arbitrary contents into shared files or gain elevated privileges (among others). One of the reasons why this type of attacks is possible is because containers have too many capabilities enabled by default. Researchers have proposed solutions that rely on identifying the syscalls invoked by containers while they are being executed using either static analysis or

dynamic analysis techniques, then enabling the necessary capabilities in containers for them to be able to invoke these syscalls only. However, these solutions suffer from several limitations. Dynamic code analysis techniques are practical but can miss syscalls triggered in execution paths that are not observed during the profiling phase. Instead, static code analysis techniques are known to produce more accurate results, but they introduce a significant overhead. In addition to this, so far existing works have focused on analysing a few popular types of Docker containers (e.g., web servers) only.

Our work. We will design a tool that accurately and efficiently characterises the syscall patterns of any type of application or container (i.e., not only Docker). So far, we have captured the syscall patterns of many containerised applications and have developed techniques based on the use of static and dynamic analysis to examine them.

1.2 Enhancing privacy within the container ecosystem

Challenges. Currently, there exists a wide range of containerised applications, each with their own internal (i.e., with the host OS kernel) communication patterns. Unfortunately, there is no work that explores how these communication patterns can be exploited by adversaries to compromise privacy. We work under the hypothesis that the containers' syscalls patterns can act as a unique fingerprint of applications running them. If true, this would allow cloud providers to know which type of applications their customers are executing (among others). This is particularly relevant for cloud providers that offer confidential cloud computing services to their customers. With confidential cloud computing, users can deploy such containerised applications inside a hardware-backed Trusted Execution Environment (TEE) without needing to trust the cloud provider or the software stack of the underlying hardware (e.g., the OS and kernel libraries). Therefore, in such a case it should be impossible for cloud providers to infer any information about the applications running inside the containers.

Our work. We will propose a tool that analyses the susceptibility of containerised applications to such privacy attacks and will develop security mechanisms to protect against them. So far, we have captured the syscall patterns of many containerised applications and have started to analyse the uniqueness of their syscall patterns. Different features have been extracted (e.g., the number of different types of syscalls invoked during the container's execution) and a preliminary analysis has been conducted.

Objective 2) Privacy-preserving ML models

Challenges. Recent works have shown that adversaries can execute attacks to retrieve sensitive information from the ML model parameters. Prominent examples of such attacks are data reconstruction and various types of inference attacks. Motivated by these attacks, researchers have recently introduced several countermeasures to prevent them. Existing solutions can be grouped into three main categories depending on whether they rely on: (i) homomorphic encryption, (ii) multi-party computation or (iii) differential privacy. However, these solutions suffer from important privacy or performance limitations, or negatively affect the utility or fairness of the model. While practical, the main problem of homomorphic encryption is that it only supports a limited number of arithmetic operations in the encrypted domain. Alternatively, the use of fully homomorphic encryption has been employed to allow arbitrary operations in the encrypted domain. Yet, this comes with too much computational overhead. Similarly, multi-party computation-based solutions cause significant computational overhead. Also, in some cases, differential privacy can fail to provide sufficient privacy. Furthermore, it can negatively impact the utility and fairness of the model as well as the system performance.

Our work. We plan to build on previous efforts and study optimal ways to build privacy-preserving FL models in a hierarchical fashion. Our goal is to optimize the ML models such that they converge well and offer high ML performance, while preserving the privacy of the data owners. We also plan to investigate the trade-offs between federated and pure peer-to-peer solutions, with respect to convergence, accuracy and privacy, and how much information may be unnecessarily shared during the federated or peer-to-peer learning, with respect to the performance achieved from each method. So far, we have explored the use of hardware-based Trusted Execution Environments (TEEs) to preclude the above-mentioned ML attacks. TEEs allow to securely store data and execute arbitrary code on an untrusted device almost at native speed through secure memory compartments. However, in order to keep the Trusted Computing Base (TCB) as small as possible, current TEEs have limited memory, which makes it impossible to simultaneously place all ML layers inside the TEE while training them. Therefore, new techniques are needed that protect against privacy attacks while considering the current limitations of TEEs. In addition to this, we have proposed Federated Learning as a Service (FLaaS), a system that enables different scenarios of 3rd-party application collaborative model building which addresses the challenges of permission and privacy management, usability, and hierarchical model training.

Objective 3) Detection of user data leakage

Challenges. Over the past few years, we have seen an increasing concern about user data protection with respect to the data of European (and other countries') users. This general concert has led to the General Data Protection Regulation (GDPR) which was adopted in April 2016 and came into force in May 2018, the California Consumer Privacy Act (CCPA) in the USA which was enforced in January 2020 and other similar legislations around the world. Such regulations have helped establish what companies should or not be doing with users' private data, and how they should be penalized when they do not respect users' online (and offline) privacy. Past research work has shown numerous ways that online companies still do not respect users' privacy, and instead, apply predatory methods and strategies to collect, store, process and share personal data. Users have been using tools such as ad-blockers and privacy browsers, in the hope of some peace of mind. However, companies still find ways to track users and profile them for their own gains.

Our work. We plan to build on past research work and develop new techniques for identifying fingerprinting happening on user devices, as well as uniqueness of devices, in order to then offer methods for blocking such identifications and removal of sensitive information leaked to unauthorized 3rd-parties. We will focus on studying how online companies perform tracking of users while they browse websites, if this tracking is differentiated depending on the user profile and the visited websites, if it has any association with political ideologies of users, or their demographics, if retargeting is affected based on user consent selection, etc.

6.4 Output description

- 1) Secure and private container execution
- 2) Creating of privacy-preserving machine learning models
- 3) Detection and prevent of user data leakage

6.5 Future Work

- (Ob1) Secure and private container execution. As future work, we will investigate the feasibility of proposing a middle-ground solution that combines static and dynamic analysis techniques to get accurate results while keeping the overhead incurred by profiling containers as low as possible. In addition to this, we will study the uniqueness of their syscall patterns using various privacy metrics and machine learning techniques.

- (Ob2) Privacy-preserving ML models. In future work, we plan to investigate how to leverage TEEs to protect against active attacks against ML models. We also plan to study how network topologies and hierarchies can be used within Federated Learning Machine Learning modeling, to increase privacy guarantees and/or improve model utility.

- (Ob3) Detection of user data leakage. In our future work, we plan to study how users are being tracked when visiting websites from different demographic and profile variations, political leaning, etc. We also plan to study how groups of websites may be capable of sharing information about unique users.

7 Intelligent edge/cloud continuum orchestrator

7.1 Description & Objectives

The ACCORDION Intelligent Orchestrator (AIO) presented in Section 4 is the main component to manage the "continuum" of resources belonging to Edge Miniclouds (EM) or public Central Clouds (CC). Specifically, the AIO aims to allow the allocation of all types of resources of the ACCORDION Federation to the applications running on the Federation.

The main objectives of AIO are the delivery of allocation plans that matches applications' requirements (e.g., in terms of QoE/QoS, etc.) and the adaption of its decisions according to the resource conditions. These decisions have also to achieve an efficient exploitation of the ACCORDION Federation resources. Second, AIO aims to adapt the network capabilities that ensure reliability and latency requirements. Nonetheless, AIO has also to adapt both allocation and network plans based on fault tolerance and resilience components to ensure the integrity of the applications. Both objectives, allocation and network plans, are particularly complex to achieve because the AIO has to take decisions in a highly dynamic changing environment and proactively take decisions to fulfill all application requirements. Finally, AIO has to ensure the security and privacy of the applications inside the ACCORDION Federation.

AIO achieves its goal by organizing its activity in four major steps: (i) the core component for end-to-end cloud/edge selection, and domain specific allocation for delivering both allocation and network plans, (ii) resilience policies & mechanisms component that provides a proactive fault tolerance model using data features related with the resource usage and the mobility, (iii) security-enhanced application development & deployment that identifies and asses technologies and tools applicable to continuum framework, and (iv) privacy-preserving mechanism to ensure the privacy of the users and applications and any data leakage.

7.2 Requirements

The baseline ACCORDION Intelligent Orchestrator must satisfy a set of requirements in different aspects, to have a good open-source basis to enable further development. It should be easy to extend with provided interfaces and should have an active community working on its evolution. It should be able to manage the ACCORDION Federation of resources and to deliver allocation plans while at the same time ensures reliability, latency and application requirements and ensures an efficient exploitation of the resources. Moreover, AIO is connected to many different modules of the ACCORDION Platform. Such interactions are required to feed the orchestrator with all the information needed to drive the allocation process. Interactions with modules developed by WP3 are needed to receive information about resources. From WP5 modules are derived the data related to the applications. Finally, the interplay with the other modules developed within WP4 are aimed at ensuring resilience and security.

7.3 Research Challenges & Advancements Achieved

As stated in the previous sections, the baseline ACCORDION Intelligent Orchestrator highly relies on the component described in Section 4. All the several research challenges described there can be directly applied here. Therefore, to avoid duplication, the achievement and research challenges are described extensively at Section 4.3.

7.4 Future Work

In the second year we will integrate the other components developed within the WP4, while we will develop many advancement and refinements planned on the ACCORDION Intelligent Orchestrator, both from an implementation and algorithmic viewpoint. We will start integration testing using a sample application, but then the plan is to run experiments using the ACCORDION WP6 use cases, to verify if their requirements are satisfied.

8 Conclusion

This deliverable summarizes the effort of WP4 partners, during the first cycle of the project, toward developing a framework for managing the Edge/Cloud continuum. It provides a first overview of each task of this framework. Furthermore, involved partners have also identified the objectives and the requirements of each task.

This deliverable also delves into the scientific challenges and advancement achieved regarding the orchestration of compute and network resources, resilience policies & mechanisms, security for applications' development & deployment, and privacy-preserving mechanisms. Finally, we provided the output of each task and we discussed the plan for future work in the coming months of the project.

9 References

[1] W. Tärneberg *et al.*, "Dynamic application placement in the Mobile Cloud Network," *Future Gener. Comput. Syst.*, vol. 70, pp. 163–177, May 2017, doi: 10.1016/j.future.2016.06.021.

[2] T. Bahreini and D. Grosu, "Efficient placement of multi-component applications in edge computing systems," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, New York, NY, USA, Oct. 2017, pp. 1–11, doi: 10.1145/3132211.3134454.

[3] G. Castellano, F. Esposito, and F. Risso, "A Distributed Orchestration Algorithm for Edge Computing Resources with Guarantees," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, Apr. 2019, pp. 2548–2556, doi: 10.1109/INFOCOM.2019.8737532.

[4] L. Baresi, D. F. Mendonça, M. Garriga, S. Guinea, and G. Quattrocchi, "A Unified Model for the Mobile-Edge-Cloud Continuum," *ACM Trans. Internet Technol.*, vol. 19, no. 2, p. 29:1–29:21, Apr. 2019, doi: 10.1145/3226644.

[5] X.-Q. Pham, T.-D. Nguyen, V. Nguyen, and E.-N. Huh, "Utility-Centric Service Provisioning in Multi-Access Edge Computing," *Appl. Sci.*, vol. 9, no. 18, Art. no. 18, Jan. 2019, doi: 10.3390/app9183776.

[6] S. Wang, M. Zafer, and K. K. Leung, "Online Placement of Multi-Component Applications in Edge Computing Environments," *IEEE Access*, vol. 5, pp. 2514–2533, 2017, doi: 10.1109/ACCESS.2017.2665971.

[7] H. Li, N. Chen, B. Liang, and C. Liu, "RPBG: Intelligent Orchestration Strategy of Heterogeneous Docker Cluster Based on Graph Theory," in *2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, May 2019, pp. 488–493, doi: 10.1109/CSCWD.2019.8791504.

[8] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 3, pp. 2429–2453, thirdquarter 2018, doi: 10.1109/COMST.2018.2815638.

[9] T. Taleb, I. Afolabi, K. Samdanis, and F. Z. Yousaf, "On Multi-Domain Network Slicing Orchestration Architecture and Federated Resource Control," *IEEE Netw.*, vol. 33, no. 5, pp. 242–252, Sep. 2019, doi: 10.1109/MNET.2018.1800267.

[10] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network Slicing in 5G: Survey and Challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, May 2017, doi: 10.1109/MCOM.2017.1600951.

[11] O. Wahab, A. Mourad, H. Otrok, and T. Taleb, "Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems," *IEEE Commun. Surv. Tutor.*, Feb. 2021, doi: 10.1109/COMST.2021.3058573.

[12] D. Kimovski, H. Ijaz, N. Saurabh, and R. Prodan, "Adaptive Nature-Inspired Fog Architecture," in *2018 IEEE 2nd International Conference on Fog and Edge Computing (ICFEC)*, May 2018, pp. 1–8, doi: 10.1109/CFEC.2018.8358723.

[13] J. Violos *et al.*, "User Behavior and Application Modeling in Decentralized Edge Cloud Infrastructures," in *Economics of Grids, Clouds, Systems, and Services*, Cham, 2017, pp. 193–203, doi: 10.1007/978-3-319-68066-8_15.

[14] J. Violos, S. Pelekis, A. Berdelis, S. Tsanakas, K. Tserpes, and T. Varvarigou, "Predicting Visitor Distribution for Large Events in Smart Cities," in *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Feb. 2019, pp. 1–8, doi: 10.1109/BIGCOMP.2019.8679181.

[15] J. Violos, E. Psomakelis, D. Danopoulos, S. Tsanakas, and T. Varvarigou, "Using LSTM Neural Networks as Resource Utilization Predictors: The Case of Training Deep Learning Models on the Edge," in *Economics of Grids, Clouds, Systems, and Services*, Cham, 2020, pp. 67–74, doi: 10.1007/978-3-030-63058-4_6.

[16] J. Violos, T. Pagoulatou, S. Tsanakas, and K. Tserpes, "Predicting Resource Usage in Edge Computing Infrastructures with CNN and a Hybrid Bayesian Particle Swarm Hyper-parameter Optimization Model," presented at the Computing Conference, London, UK, Jul. 2021.

[17] I. Bouras *et al.*, "Mapping of Quality of Service Requirements to Resource Demands for IaaS," Feb. 2021, pp. 263–270, Accessed: Feb. 21, 2021. [Online]. Available: https://www.scitepress.org/Link.aspx?doi=10.5220/0007676902630270.

[18] J. Violos, S. Tsanakas, M. Androutsopoulou, G. Palaiokrassas, and T. Varvarigou, "Next Position Prediction using LSTM Neural Networks," in *11th Hellenic Conference on Artificial Intelligence*, New York, NY, USA, Sep. 2020, pp. 232–240, doi: 10.1145/3411408.3411426.

[19] Y. He, W. J. Ma, and J. P. Zhang, "The parameters selection of PSO algorithm influencing on performance of fault diagnosis," in *MATEC Web of conferences*, 2016, vol. 63, p. 02019.

[20] E. Brochu, V. M. Cora, and N. de Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning," *ArXiv10122599 Cs*, Dec. 2010, Accessed: Feb. 21, 2021. [Online]. Available: http://arxiv.org/abs/1012.2599.

[21] J. Violos, E. Psomakelis, K. Tserpes, F. Aisopos, and T. Varvarigou, "Leveraging User Mobility and Mobile App Services Behavior for Optimal Edge Resource Utilization," in *Proceedings of the International Conference on Omni-Layer Intelligent Systems*, New York, NY, USA, May 2019, pp. 7–12, doi: 10.1145/3312614.3312620.

[22] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[23] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, Cambridge, MA, USA, Dec. 2015, pp. 2755–2763, Accessed: Feb. 20, 2021. [Online].

[24] "Cloud Security Alliance, top threats report." https://cloudsecurityalliance.org/press-releases/2020/09/23/cloud-security-alliance-releases-top-threats-to-cloud-computing-egregious-11-deep-dive-articulates-cloud-computing-s-most-significant-issues/ (accessed Feb. 21, 2021).

[25] "CIS Controls Implementation Guidance for the Cloud v1 (open access but registration required)." https://workbench.cisecurity.org/benchmarks/721 (accessed Feb. 21, 2021).

[26] "https://www.openhub.net." (accessed Feb. 21, 2021).

[27] “https://www.sonarqube.org.” (accessed Feb. 21, 2021)..

[28] “https://docs.gitlab.com/ee/user/application_security/sast/.” (accessed Feb. 21, 2021)..

[29] “https://owasp.org/www-project-zap/.” (accessed Feb. 21, 2021)..

[30] “https://docs.gitlab.com/ee/user/application_security/dast/.” (accessed Feb. 21, 2021)..

[31] “https://anchore.com/.” (accessed Feb. 21, 2021)..

[32] “https://coreos.com/clair/docs/latest/.” (accessed Feb. 21, 2021)..

[33] “https://goharbor.io.” (accessed Feb. 21, 2021)..

[34] “https://www.vaultproject.io/.” (accessed Feb. 21, 2021)..