MEDINA

An architecture proposal for the MEDINA framework

(Whitepaper)

Editor(s):	Iñaki Etxaniz
Responsible Partner:	TECNALIA
Status-Version:	Final - v1.0
Date:	30.04.2022
Distribution level:	Public

Project Number:	952633
Project Title:	MEDINA

Editor(s):	Iñaki Etxaniz, TECNALIA
Contributor(s):	Mika Leskinen, NIXU Cristina Regueiro, TECNALIA Immanuel Kunz, Florian Wendland, FhG Franz Berger, Fabasoft Michela Fazzolari, CNR Daniele Garbagnati, HPE
Reviewer(s):	Jesus Luna Garcia, Bosch Cristina Martínez, TECNALIA
Approved by:	All Partners
Recommended readers:	Cloud Service Providers, Auditors, IT Security-related

Keyword List:	Software Architecture, Components, API, Metrics, Automated Monitoring, Continuous Certification
Licensing information:	This work is licensed under Creative Commons Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0) http://creativecommons.org/licenses/by-sa/3.0/
Disclaimer	This document reflects only the author's views and neither Agency nor the Commission are responsible for any use that may be made of the information contained therein.

Document Description

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
v0.1	25.04.2022	First draft version	TECNALIA
v0.2	29.04.2022	Revised draft	All contributors
v1.0	30.04.2022	Final revised version	TECNALIA, Bosch

Table of contents

List of Tables

List of Figures

Terms and abbreviations

API	Application Programming Interface
AWS	Amazon Web Services
CAB	Conformance Assessment Body
CISO	Chief Information Security Officer
EUCS	European Cybersecurity Certification Scheme for Cloud Services
CI/CD	Continuous Integration / Continuous Delivery
CNL	Controlled Natural Language
CSA or EU CSA	EU Cybersecurity Act
CSP	Cloud Service Provider
CSC	Cloud Service Customer
CSPM	Cloud Security Posture Management
DSL	Domain Specific Language
DLT	Distributed Ledger Technology
EC	European Commission
EUCS	European Cybersecurity Certification Scheme for Cloud Services
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICO	Internal Control Owner
IoT	Internet of Things
JWT	Json Web Token
KPI	Key Performance Indicator
NCB	National Certification Body
NLP	Natural Language Processing
OPA	Open Policy Agent
PaaS	Platform as a Service
PII	Personally Identifiable Information
RKE	Rancher Kubernetes Engine
SaaS	Software as a Service
SATRA	Self-Assessment Tool for Risk Analysis
SSO	Single Sign-On
ToC	Target of Certification
TOM	Technical and Organizational Measure
UI	User Interface
VM	Virtual Machine

Executive Summary

This whitepaper focuses on the description of the software and hardware architecture of the MEDINA framework, which has been designed and implemented during the first 18 months of the EU MEDINA project.

First, the report briefly introduces the problem of trustworthiness in cloud services in the EU, the continuous monitoring of cloud services and the certification issues. The MEDINA project aims to provide CSPs with a tool that allows them to audit and certify of Cloud Services in an automated and near real-time manner. The continuous certification proposed in MEDINA is aligned to the EUCS framework. The paper describes the principal concepts and ideas leading to the MEDINA framework, namely: the definition of a metrics catalogue based in EUCS; the implementation of a machine-readable language to allow the interpretation of natural-language specifications; the automated and continuous evidence collection and assessment; and the managing of digital evidences for accountability.

Then, an overview of MEDINA is presented, which includes the roles, workflows and tasks related to the framework, as they have been defined during the first half of the project. Three are the main roles taking part in the basic workflow: the Compliance Manager, the Control Owner and the External Auditor. Seven workflows are defined, that compose the whole tasks that the users of MEDINA have to undertake to achieve their goals. The workflows comprise the preparation of the components, the definition of the target of certification (ToC), the deployment of the framework on the ToC, the self-assessment, the compliance assessment, and the maintenance and report of the certificate. This part is completed with a diagram of the system architecture, that decomposes the framework in components that are dedicated to specific tasks, and that collaborate to provide the explained functionality.

Finally, the components of the MEDINA framework are described. The components in which we have divided MEDINA are the Catalogue of Controls & Security Schemes; the components that deal with the NLP (Natural Language Processing) techniques; the Risk Assessment and Optimisation Framework; the Continuous Evaluation; The Orchestrator and the Trustworthiness System; the Evidenced Collection and Security Assessment; the Certificate management System; and the Integrated User Interface that works as a wrapper of the rest of components for the final user. The functionality of each component is briefly presented, showing the interactions with the rest of the framework. It is not the objective of this white paper to present a deep view into the technical issues tackled by each of the components, but a birds-eye view that allows the reader to grasp a general vision of what MEDINA tries to offer. In this sense, some screenshots of the prototypes developed in the project are also included throughout the text, which can help the reader to get an idea of the current state of development of the tools.

1 Introduction

The adoption of Cloud Computing in the EU is still limited. Among others, one of the reasons for this is lack of security and transparency that customers perceive in this technology. Cloud Service Providers (CSPs) often rely on security certifications as a mean to improve this transparency and trustworthiness, but this is not easy to do in Europe nowadays, mainly due to the fragmentation of the certification market.

In this context, the EU Cybersecurity Act (EU CSA) proposes improving customer trust in the European ICT market through a set of EU-wide certification schemes. One of them is the European Cybersecurity Certification Scheme for Cloud Service (EUCS[1]) being developed by the European Union Agency for Cybersecurity (ENISA). The draft EUCS introduces novel concepts such as:

- Three different levels of assurance: Basic, Substantial and High
- Composition of certifications for the cloud supply chain
- Automated/continuous monitoring for high assurance certification

Figure 1. EUCS levels of assurance at a glance (adapted from ENISA)

All this implies new technological challenges for cloud service providers, which need to be solved in order to fully achieve the expected benefits. In this context, the MEDINA project[2] aims to provide CSPs with a tool that allows them to audit and certify of Cloud Services in an automated and near real-time manner. Among other technology partners, the MEDINA project involves public CSPs (like Bosch and Fabasoft) and Assessment Bodies (NIXU) [3].

*The main objective of the MEDINA European research project is to **provide a holistic framework that enhances cloud customers' control and trust in consumed cloud services**, by supporting CSPs (IaaS, PaaS and SaaS providers) towards the successful achievement of a continuous certification aligned to the EUCS. The proposed framework will be leveraged into a cloud supply chain, is comprised of tools, techniques, and processes that will support the continuous assessing of the efficiency and efficacy of security measures to ultimately achieve and maintain a certification of cloud services.*

[1] Draft version available at https://www.enisa.europa.eu/publications/eucs-cloud-service-scheme
[2] Please refer to https://medina-project.eu/
[3] Bosch is also member of ENISA AdHoc WG for EUCS. More information about Bosch, Fabasoft and NIXU can be found here https://medina-project.eu/partners

MEDINA is based on some main ideas or pillars, that give the project a special relevance regarding the notion of continuous and automated monitoring of EUCS. They can be summarised up as follows:

METRICS CATALOGUE

The EUCS draft provides a set of security requirements, mostly based on international standards, which shall we leveraged to certify cloud services. At the time of writing, EUCS does not define the concrete guidelines or "compliance metrics" to be used to assess the requirements. The lack of standard EUCS-metrics can become a problem for CSP and CABs, which might need to leverage their own custom metrics for implementing/assessing EUCS requirements in an automated manner.

MEDINA is defining a catalogue of metrics associated to technical and organizational measures (TOMs) in EUCS. The metrics repository in MEDINA covers topics such as system security and integrity, operational security, business continuity and incident management.

RISK-BASED APPROACH FOR SECURITY CONTROLS

MEDINA proposes a risk-based, tool-supported methodology for the selection of EUCS-complementary controls and associated TOMs based on the CSP's risk appetite. Such controls and requirements shall address the concrete needs of a CSP, by also taking into consideration the targeted EUCS assurance level.

CERTIFICATION LANGUAGE

The security control frameworks -and EUCS is not an exception- are in practice defined in natural language. At some point it needs to be "translated" into a machine-readable representation, which facilitates the elicitation of metrics and controls.

MEDINA proposes to transform the natural-language specification of control frameworks like EUCS into a machine-readable expression, by using NLP (Natural Language Processing) techniques. The expected outcome should comprise aspects like scope of the certification, assurance level and conformity assessment method.

EVIDENCE COLLECTION AND CONTINUOUS AUDIT

Essential for achieving continuous audit-based certification is the collection of actual, technical evidence related to the automated monitoring. From a technical point of view, one could distinguish between tools and methodologies to address this at code level and at service level. The topic of managing digital evidence related to EUCS will become critical once CSPs start applying for a high-assurance certificate.

MEDINA aims to develop a framework for managing digital evidence related to EUCS that, as well as risks, are continuously monitored and evaluated. Collected evidence in MEDINA will use DLT / Blockchain techniques for implementing accountable tracking.

In this whitepaper we present the work done on the architecture and integration tasks in the EU MEDINA project, aimed at the implementation of (automated) continuous monitoring as defined in the draft EUCS[4]. It describes the architecture of the software framework behind the tool, the

[4] EUCS version of December 22nd, 2020, available online at https://www.enisa.europa.eu/publications/eucs-cloud-service-scheme

parts into which the MEDINA framework is divided, and how they are organized, and gives some details of the current prototype implementation.

The rest of this document is organized as follows: Section 2 provides a high-level overview of the MEDINA architecture, the context of the solution, the key expected outcomes, the task workflows involved and how it is all being integrated. Section 3 describes the individual software components of the architecture presented before, explaining their function, features, and some implementation details.

2 MEDINA Architecture and Overview

The definition of the MEDINA framework has been approached from different points of view. Starting with the use cases, the users of the system and the workflows they are involved in have been defined. From the functionality point of view, the whole framework has been divided into several components or sub-tools dedicated to specific tasks. The following is a general view of MEDINA from different angles, including integration and deployment.

2.1 Initial approach, roles and workflows

The core of the MEDINA project is a modular framework that enables CSPs to implement and manage continuous certification of their cloud supply chain according to the EU CSA requirements. Due to the myriad of different cloud service delivery models, as well as the different provisions of the EU CSA certification scheme, the framework itself needs to be agnostic with respect to the actual cloud service and offer abstractions through different components. While MEDINA aims to provide a reference implementation of those components, it is important to note that the framework will also enable and guide a CSP to achieve continuous certification through its already established toolchain (or a combination of both).

The main roles involved in the continuous certification of cloud services have been identified and are shown in the table below:

Table 1. Generic MEDINA roles

Role	Description
Compliance Manager	Responsible for the implementation of the Security Control Frameworks
	Responsible for the correct implementation of the internal controls
	Responsible for incident management
Control Owner	Responsible for creating the internal controls
	Executing the internal controls
External Auditor	Performs all actions required to audit a Company
	Responsible for performing internal audits

The basic workflow of MEDINA is illustrated in Figure 2. The users of the solution have their own swimming line in the figure and time flows from left to right. This diagram only serves to illustrate the basic concept.

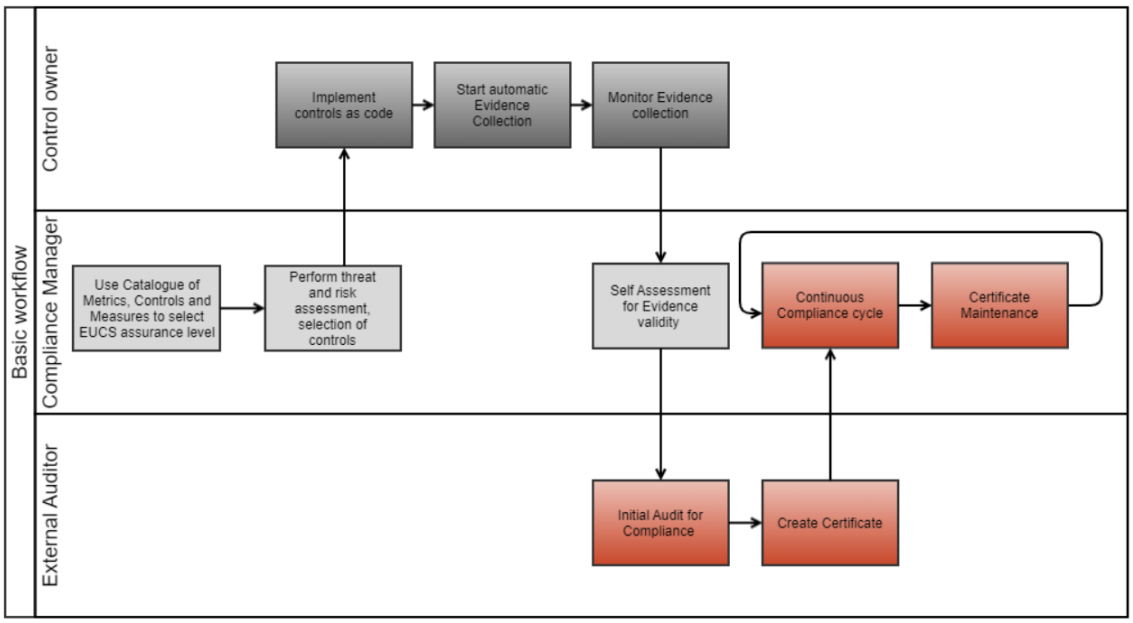

Figure 2. Basic workflow of MEDINA

The basic workflow shown above can be broken down in several parts, which have to be performed sequentially by MEDINA users to achieve the goals defined in different MEDINA use cases. These workflows, which make use of the components of the MEDINA framework, are grouped into seven scenarios/interactions as shown in Table 2.

Table 2. MEDINA generic workflows

WF1: Preparation of Target of Certification (ToC)
Objective: Setup, configure and deploy the cloud service to certify (ToC) on top of the chosen hyperscaler(s). This process includes configuring the underlying PaaS/IaaS. **Other**: Prerequisite, Mandatory workflow; CSP Responsibility. **Dependencies**: None
WF2: Preparation of MEDINA components
Objective: Setup, configure and deploy MEDINA components. Only related to those components under the responsibility of the CSP. **Other**: Prerequisite, Mandatory workflow; CSP Responsibility. **Dependencies**: WF1
WF3: EUCS - Deployment on ToC
Objective: Setup, configure and deploy the corresponding EUCS framework (for the chosen assurance level basic/substantial/high) on the ToC. **Other:** Prerequisite, Mandatory workflow; CSP Responsibility. **Dependencies**: WF1, WF2

WF4: EUCS - Preparedness – ToC Self-Assessment
Objective: Self-assess preparedness for EUCS certification based on the chosen assurance level. This is a risk-based approach. **Other:** Optional workflow; CSP Responsibility. **Dependencies:** WF1, WF2, WF3
WF5: EUCS – Compliance assessment
Objective: Perform a point-in-time (discrete) EUCS compliance assessment for the ToC. When such discrete assessment is periodically executed, we achieve the MEDINA notion of "continuous". **Other:** Mandatory workflow; CAB Responsibility. **Dependencies**: WF1, WF2, WF3
WF6: EUCS – Maintenance of ToC certificate
Objective: Start certificate maintenance life-cycle for the ToC. Based on current EUCS, the maintenance process comprises the following stages: (issuance[5]), renewal, continuation, update, re-issuance (new certificate), withdrawal, and suspension. **Other:** Mandatory workflow; CAB & CSP Responsibility. **Dependencies**: WF1, WF2, WF3, WF5
WF7: EUCS – Report on ToC certificate
Objective: Report on EUCS certificate status for a ToC. The report can be obtained by the CAB and the CSP, in which case the level of provided details might vary. **Other:** Optional workflow; CAB & CSP Responsibility. **Dependencies**: WF1, WF2, WF3, WF5

From the seven workflows presented in this section, more complex workflows can be built. Creating and instantiating real-world scenarios based on the generic workflows are goals to be achieved in the following months.

2.2 General view of the architecture

The architectural framework proposed by MEDINA can be abstracted into the components shown in Figure 3. Each block corresponds to a distinct functionality of the tool, that will be developed and refined in the coming months, and finally instantiated and validated in the two use cases of the MEDINA project.

The main building blocks of the MEDINA framework are described below.

[5] Despite the initial issuance of certificate is not mentioned in the maintenance process defined in the EUCS core document, for MEDINA purposes this discussion is part of the life-cycle manager.

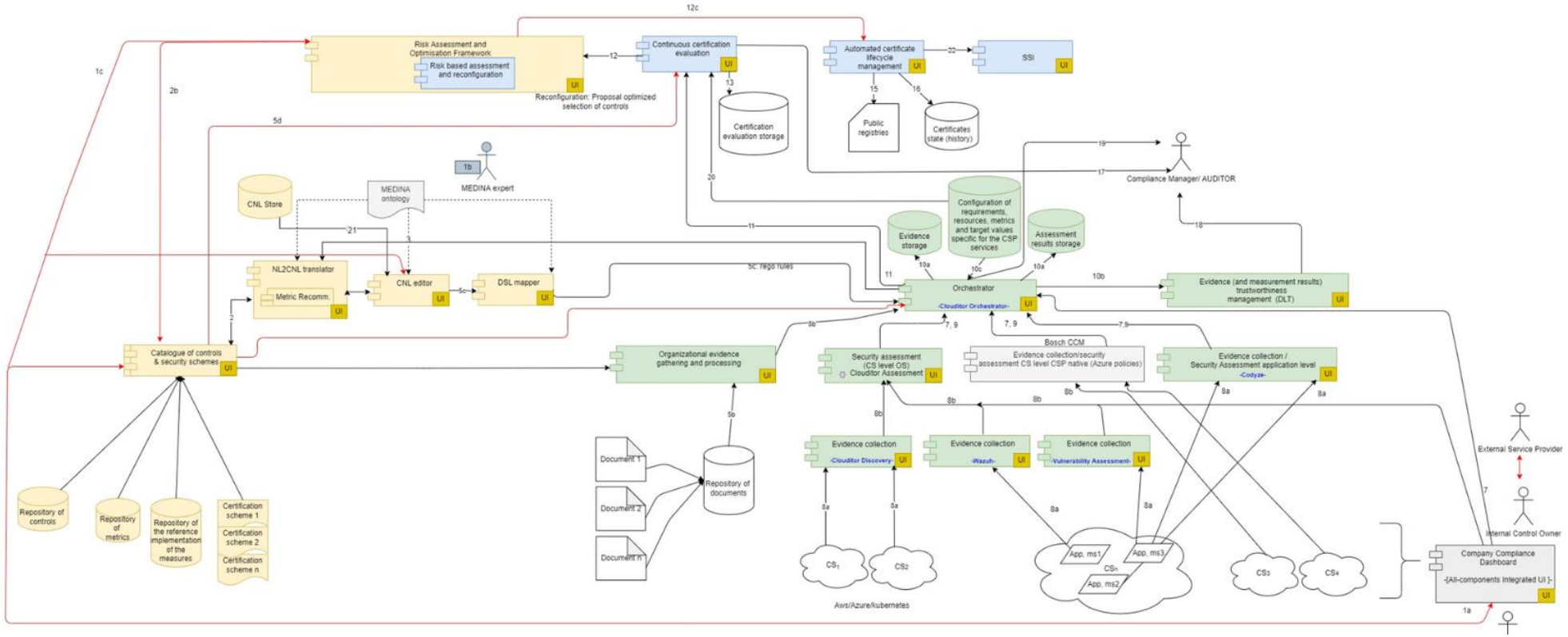

Figure 3. Architecture diagram of the MEDINA framework

FREE OPEN SOURCE FRAMEWORK FOR EVIDENCE COLLECTION AND ASSESSMENT

The core of the MEDINA solution is the continuous process of evidence collection and assessment. This functionality is implemented by **Clouditor** and the **Orchestrator** (KR4), an open source tool that enables "continuous cloud assurance".

This means that Clouditor will continuously (i.e. automatically and repeatedly) test if a cloud-based application complies with the critical requirements arising from the compliance framework. As Clouditor natively collects evidence only from the cloud infrastructure, the Orchestrator complements this tool with evidence collected for Malware intrusion, threat resistance and vulnerability resistance in Cloud Services and platforms. Also, organisational evidence - typically documents- can be collected and scanned for non-conformities.

WIDE RANGE OF EVIDENCES COLLECTED

The MEDINA framework collects evidence relevant to the selected compliance framework. For this purpose, complementary tools are used in partnership:

Clouditor evidence discovery can discover resources on different cloud platforms (Microsoft Azure, Amazon Web Services and OpenStack) and can discover different services on these systems, including computing, storage, and networking services. After discovery the tool can start evidence collection.

Wazuh is an open-source security monitoring tool for threat detection, integrity monitoring, incident response and basic compliance monitoring. In MEDINA, Wazuh is used to satisfy and verify security controls related to malware protection, logging, threat analytics and automatic monitoring.

Vulnerability Assessment Tool (VAT) is a vulnerability scanning and detection framework that includes several vulnerability scanning tools. It can be configured to periodically scan the CSP's infrastructure and detect vulnerabilities and potential threats. VAT can be used to satisfy or gather evidence for controls related to vulnerability detection, use of encrypted communication, detection of new devices, etc.

Codyze is an evidence gathering tool specialized in detecting software vulnerabilities through static analysis of source code and software dependencies.

AMOE (Assessment and Management of Organisational Evidences) is a tool that has been developed for organisational evidence gathering (cloud security schemes include some controls and requirements that are organisational in nature, which means that they are not suitable to be automatically monitored like technical requirements). The same approach followed for technical requirements can be adapted, resulting in the usage of so-called organisational metrics.

For this purpose, MEDINA uses **Natural Language Processing** (NLP) techniques. The core of this block is the **Certification Language** (KR3) based in a novel **ontology** that ensures that requirements written in natural language are automatically associated to specific CSP resources. Our goal is to automate the current (manual) process performed by compliance managers and auditors to interpret, map and assess organizational requirements, saving time and effort in the auditing process.

CONTINUOUS CERTIFICATION IS THE GOAL

To provide this core functionality, the MEDINA framework relies on additional tools which are seamlessly integrated.

The **Catalogue of Controls and Metrics** (KR1) provides the information of the certification scheme -in this case EUCS- with the controls, requirements, metrics and instructions on how to assess the target cloud services. It may also include the equivalencies of controls among different security schemes.

The **Metric Recommender** -also based on NLP- identifies the most appropriate metrics to fulfil a requirement described in natural language. The Certification Language also provides the translation of natural language (e.g. a pdf document) into Domain Specific Language (Rego) to be consumed by the Orchestrator.

The **Trustworthiness System** provides a secure mechanism for MEDINA to maintain an audit trail of evidence and assessment results. It is based on a blockchain network where information is stored to ensure the integrity of the information.

Continuous certification is a unique feature of MEDINA. The **Certificate Lifecycle Manager** (KR6) uses the assessment data to continuously evaluate the certificate, makes a risk aware recommendation for major non-conformities, and finally updates the certificate status to a public registry accordingly. The certificate status can be continued, suspended or withdrawn. National Certification Bodies (NCBs), or even pan-European entities such as ENISA[6], can benefit by becoming public registers of issued certificates.

MEDINA will also provide a user interface to facilitate human interaction with the other components. The **Compliance dashboard** provides situational awareness. For example, a corporate compliance manager visualizing the near real-time status of issued EUCS certificates, or an external assessment body reviewing the evidence used for a certification process.

2.3 Integration and test environment

During development, the MEDINA framework is available in a testbed environment that serves to test and verify all the functionalities. The testbed environment is setup with a three-node Kubernetes[7] cluster with two different, independent and isolated virtual environments: development and test. All the micro-services are containerised and communicate with each other with RESTful APIs over HTTPS secure protocol. The micro-services cluster are packaged in Docker[8] images and stored in a private Docker registry running on Artifactory[9].

[6] https://www.enisa.europa.eu/
[7] https://kubernetes.io/docs/home/
[8] https://docs.docker.com/engine/reference/commandline/images/
[9] https://jfrog.com/artifactory/

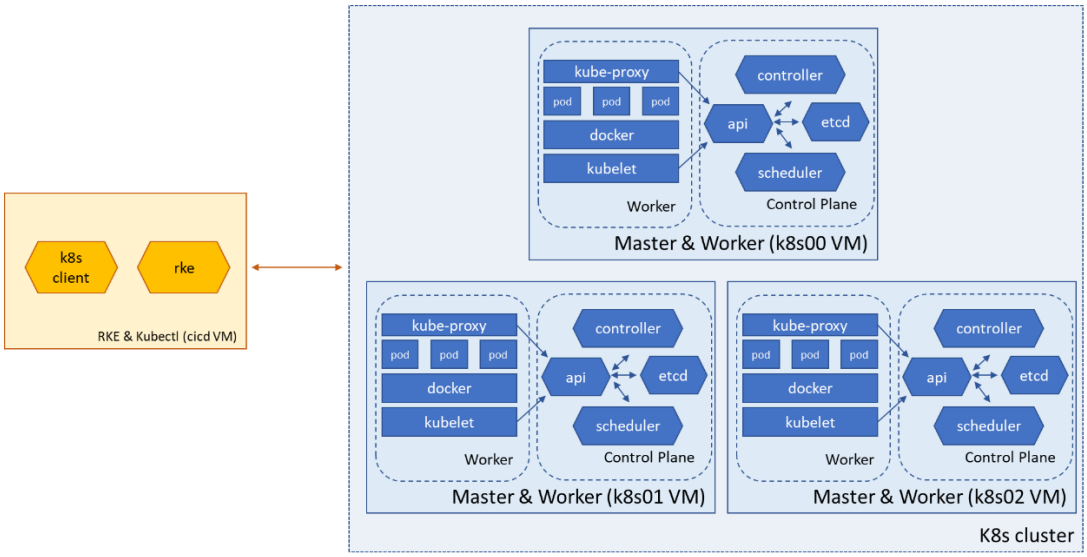

Figure 4. Kubernetes cluster installation with RKE

The final deployment model will depend both on the business model that is finally defined and on the requirements of the certification stakeholders. From a technical point of view, most MEDINA components can generally be offered as Web Tools (SaaS) accessible through a browser, or as containerised tools, but in some cases (e.g. the Evidence Management Tool) it is mandatory to install the application locally.

3 Components description

In the following paragraphs we present the main components into which the MEDINA framework has been divided, describing the functionality they provide and the main properties.

3.1 Catalogue of Controls & Metrics

The catalogue provides the following functionalities:

- Endorsement of Security Control Frameworks and related attributes: security requirements, categories, controls, reference TOMs, metrics, evidence types and assurance levels.
- Provision of guidance for the (self-)assessment of the requirements.
- Filtering of the information based on some values for the attributes (e.g. selection of requirements of a certain assurance level; metrics related to a specific reference TOM).
- Homogenization of the certification schemes: provision of information about related requirements from different frameworks especially referenced to the EUCS.

The catalogue is implemented as a JHipster component that is composed by several parts (see Figure 5): (i) **Registry**: that stores the available list of Frameworks and the related info for a specific CSP. This component will also include the corresponding (MySQL) database for persistence; (ii) **Back-end**: is the core sub-component of the Catalogue. It will perform the actual discovery of the requirements, evidence, etc. from the registry, using the set of filters established by the user through the UI/ API; (iii) **Frontend**: is the graphical user interface of the Catalogue, that allows the user to filter and select the information related to the security frameworks (i.e. requirements of a certain assurance level, metrics related to reference TOM, etc.) .

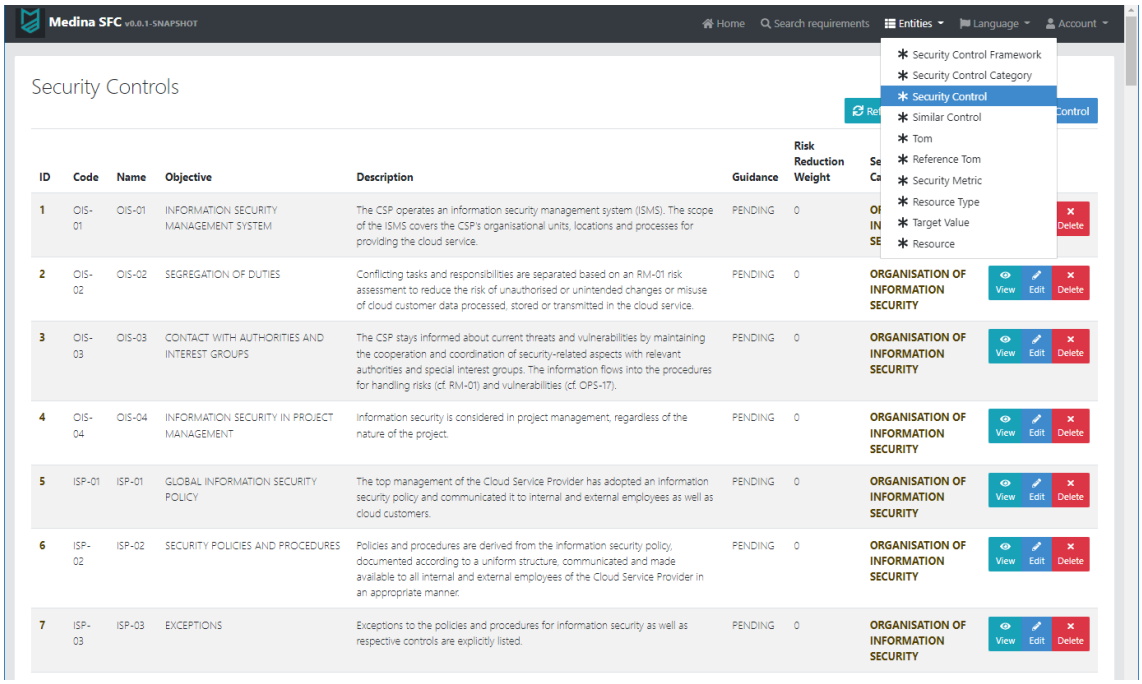

Figure 5. List of Security Controls in the Catalogue GUI

Although the main intended use of the catalogue is through its RESTful API, which allows other components to access and modify database elements, the catalogue provides a basic CRUD GUI (Create/Retrieve/Update/Delete) to access and manipulate database entities. Figure 5 presents a screenshot representing the Controls entities.

3.2 NLP Techniques

This section describes the components involved in the translation of the requirements from Natural Language (NL) to a machine-readable language, also called Controlled Natural Language (CNL). The EUCS requirements, in fact, can be associated to metrics, which can be seen as obligations, i.e. rules that the Cloud Provider has to fulfil in order to obtain the certification. The expression of these rules in the machine-readable language, so called MEDINA Domain Specific Language (DSL), will allow the MEDINA assessment tools to automatically process them. The following components are involved in the process:

- NL2CNL Translator
- CNL Editor
- DSL Mapper

Some of these components interface with the catalogue described in section 3.1. For the representation of the requirements and metrics, these components rely on the MEDINA ontology.

NL2CNL TRANSLATOR

The NL2CNL Translator is implemented as a set of RESTful APIs. The translation process is triggered by the Orchestrator, which allows the user to select a set of EUCS requirements and to send them to the NL2CNL Translator. Then, given this set of requirements in NL, the NL2CNL Translator associates to each requirement a set of metrics, by relying on an internal Metric Recommender. Then, the NL2CNL Translator transforms each metric into an obligation, expressed by a Controlled Natural Language (CNL). The output of the translation is stored in a set of objects called REOs, which are the input of the CNL Editor.

The Metric Recommender has been introduced as a research project to solve the problem of associating metrics to "new" requirements, which is normally performed manually and represents a time-consuming task. To this aim, the Metric Recommender exploits NLP techniques. In particular, the requirements and metrics are described as Word Vectors (also known as Word Embeddings), which are a way to map words or phrases to a corresponding vector of real numbers.

CNL EDITOR

The CNL Editor has a friendly user interface which allows the user to inspect the generated obligations for each requirement, to edit or delete them.

DSL MAPPER

The DSL mapper is responsible for mapping the CNL representation of the obligations to a DSL representation, whose statements are a machine-readable version that can be given as an input to the assessment tools. The implementation of this component is realized through a set of RESTful APIs. The DSL chosen as output of the DSL Mapper is the Rego language, which was inspired by Datalog, an old and widespread query language. Rego queries can be used to define policies that are easy to read and write. The output of the DSL Mapper, i.e. the Rego policies (also called Rego rules) are pushed to the assessment tools.

3.3 Risk Assessment and Optimisation Framework

The Risk Assessment and Optimisation Framework serves for the analysis of requirements demanded by a certification scheme, ensuring that fulfilment of these requirements is indeed relevant for the cloud provider (CSP). Non-conformities should be evaluated, and we use the risk assessment to say whether the detected non-conformities are major ones -and the certificate should be revoked- or the deviation is minor -and the certificate should be maintained-. The tool/service is called **Self-Assessment Tool for Risk Analysis (SATRA).**

RISK ASSESSMENT MODEL

Our risk assessment model is based on analysis of *cyber* security risk, i.e., potential events aiming to compromise *cyber* assets. Like usual, the model includes the following key components: (i) **Asset** is a valuable object -including digital objects like a container, a virtual machine or a network interface- which can be compromised by a threat; (ii) **Threat** is a potential cause of an unwanted incident, which may result in harm to a system or organization[10] e.g. account hijacking, ransomware or a hardware loss; (iii) **Vulnerability** is a weakness of an asset that can be exploited by one or more threats.

Our model aims to quantitatively estimate cyber security risks. In order to conduct such a quantitative analysis, the model estimates the possible ***impact*** of the successful compromising of an asset, the expected ***frequency*** of threats to arrive, and the ***probability*** of the threats to successfully exploit the existing vulnerabilities in the system. The model also defines relations between the outlined concepts to form a mathematical tool for computing risk values.

We consider these three typical vectors of security which could be compromised: Confidentiality, Integrity, and Availability. After a series of calculations on frequency, survival

[10] ISO/IEC 27000:2016 Information technology — Security techniques — Information security management systems — Overview and vocabulary, at https://www.iso.org/standard/66435.html

probability and expected total loss, the risk per threat vector is obtained and, finally, the total risk (a scalar real value) as a summation of risks per threat:

$$Risk = \sum_{\forall i} r_i.$$

GRAPHICAL INTERFACE AND API

The current version of SATRA covers EUCS -based risk assessment (in the future, the possibility to select a certification scheme from the *Catalogue of Controls & Security Schemes* is contemplated). The tool provides both GUI and API for interaction.

The GUI is created for direct interaction with the tool by a human operator (e.g. a compliance manager). The operator is asked to provide (i) general information, like the service market type, the selected certification scheme, and the assurance level; (b) a list of assets of each defined type, number of them and expected loss if Confidentiality, Integrity or Availability of these assets is compromised; (c) the information about implemented requirements of the selected certification scheme, e.g. EUCS (see Figure 6).

Once the inputs are provided by a CSP, the tool will calculate the risk level according to the model (see Figure 7). The CSP may see the computed risk level (a value from 0 to 100), the specific risk level for every threat and non-conformity gaps (none/minor/major non-conformity). The CSP may perform several rounds of the analysis to determine the less risky configuration of its security if full conformity with the selected certification scheme is impossible, or not required.

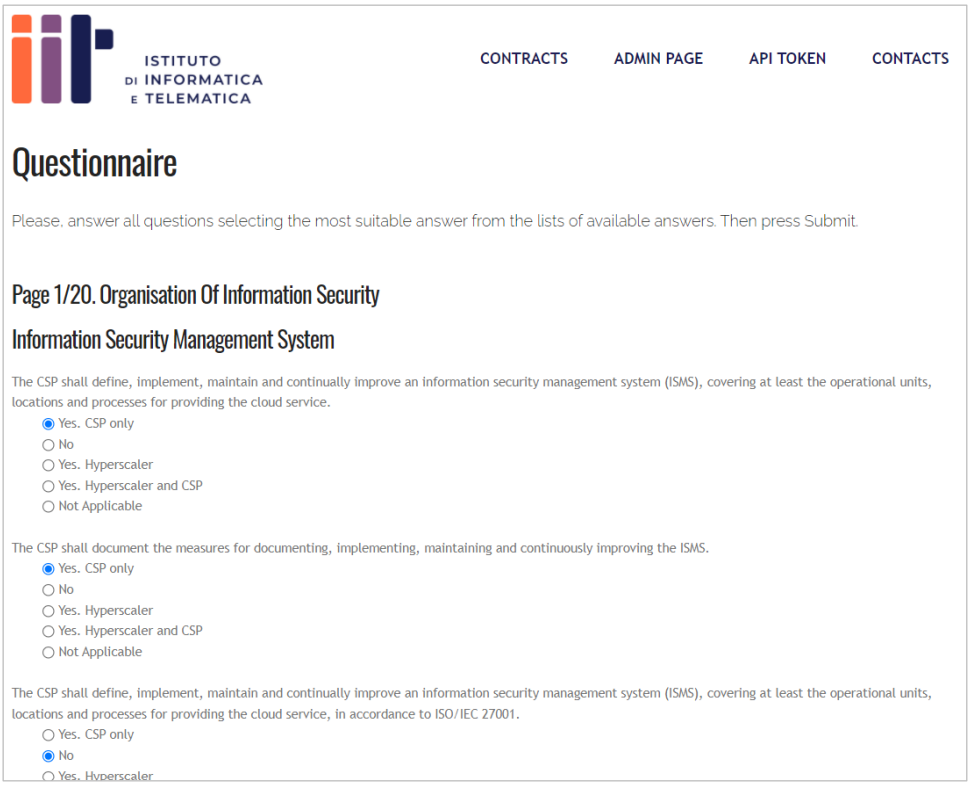

Figure 6. SATRA Questionnaire

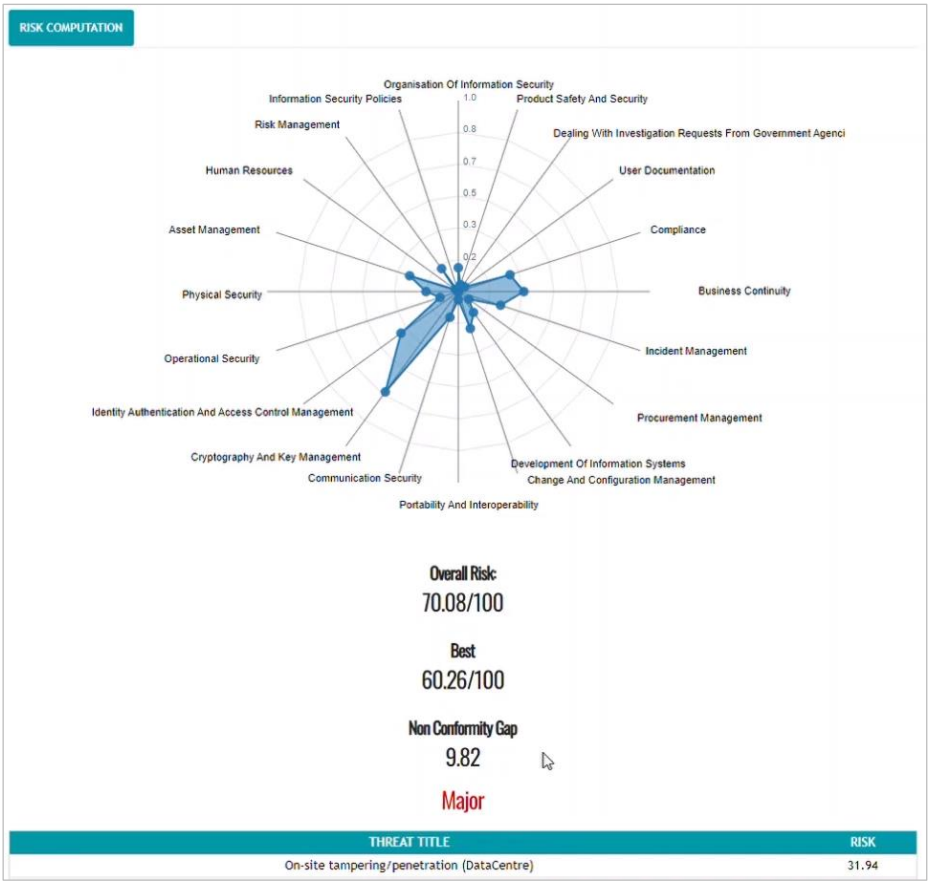

Figure 7. Risk Assessment result page

3.4 Continuous Evaluation

LIFE-CYCLE MANAGER

The Life-Cycle Manager is the final component of the evidence and assessment result data flow: it takes inputs from the Risk Assessment and Optimisation Framework, which indicate if a minor or major deviation has been detected, and translates this information into a certificate status. The certificate status is based on the status defined in the EUCS, such as *new*, *suspended*, or *renewed*. When, for example, a major deviation has been identified in the cloud system, i.e. a significant non-conformity of the security requirements has been detected, the Life-Cycle Manager uses this information to temporarily suspend the respective certificate –until remedial action has been performed.

One purpose of this component is therefore to (semi-)automatically set the certificate status, since a manual handling of the continuously generated assessment results would be impossible. A second purpose is to give internal and external auditors an overview of a set of certificates and their statuses, and to link a certificate's status back to the problematic resources, to facilitate the remediation.

The Life-Cycle Manager is implemented as a standalone Go component and it forwards decisions on certificates to the Orchestrator.

3.5 Orchestrator and Trustworthiness System

ORCHESTRATOR

The Orchestrator is a core component in the MEDINA framework since it manages the flow of evidences and assessment results and provides many APIs that are used by other components. It provides the following high-level functionalities:

- Accept assessment results from Security Assessment components and forward them to the Continuous Certification Evaluation.
- Accept assessment results from Security Assessment components, refactor them into the expected data model by the Trustworthiness System, and forward them.
- Receive metrics and provide them to the Security Assessment.
- Provide further organisational APIs, e.g. for storing or listing evidence.

The Orchestrator is implemented in Go and also provides a graphical user interface (see Figure 8).

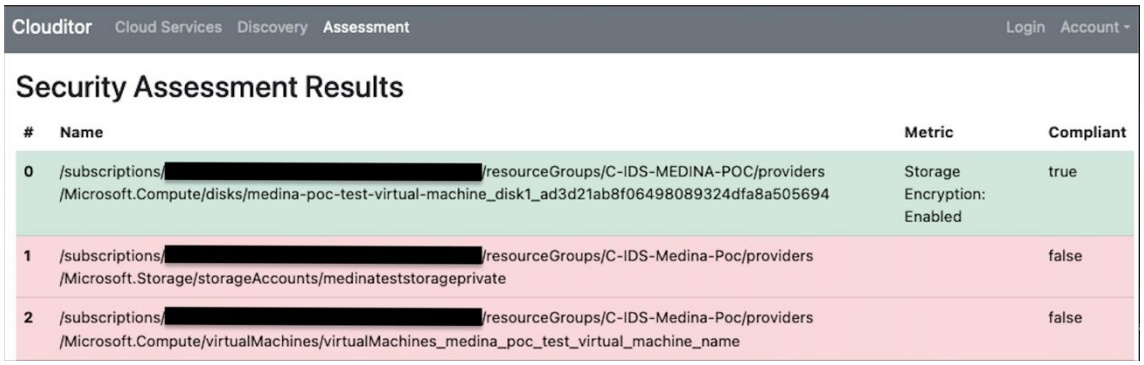

Figure 8. Excerpt from an early prototype of the Orchestrator UI showing assessment results

TRUSTWORTHINESS SYSTEM

MEDINA's Blockchain-based Evidence Trustworthiness Management System provides a secure mechanism for MEDINA to maintain an audit trail of evidence and assessment results. In MEDINA, the Trustworthy management system is implemented on **Smart Contracts** backboned by a common **Blockchain** network for all instances of the MEDINA framework, providing the following functionalities:

- Secure long-term recording of information, thanks to the inherent advantages of the Blockchain (transparency, integrity, decentralisation, authenticity…).
- Logic for all the MEDINA orchestrators instances to provide the information required to be audited (on evidence and assessment results) through a REST API.
- Logic for external users to access the audited MEDINA information (on evidence and assessment results) in a graphical and user-friendly way, making the Blockchain technology transparent to "ordinary" users, as shown in Figure 9.

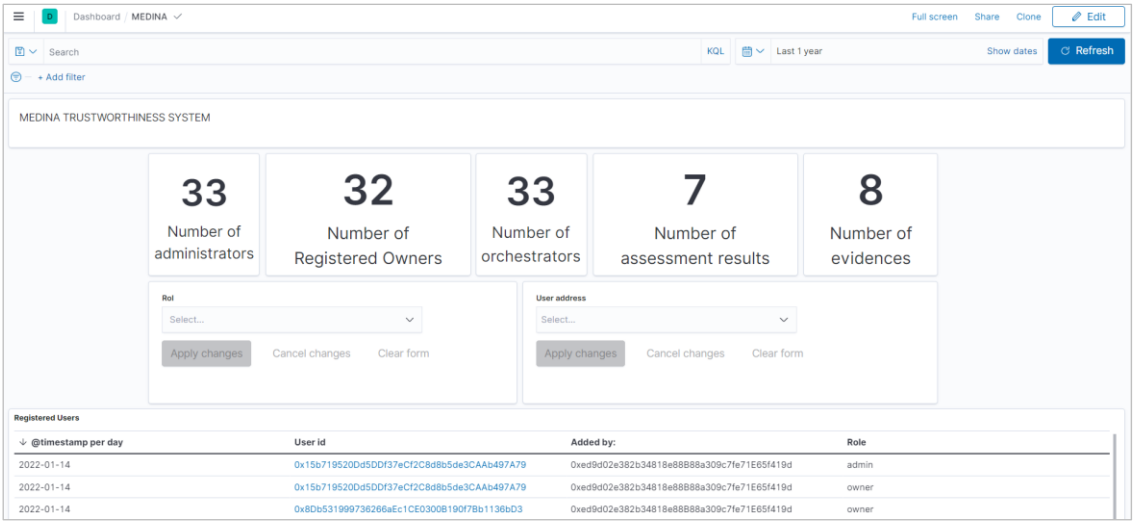

Figure 9. Trustworthiness System General Dashboard

3.6 Evidence Collection and Security Assessment

CLOUDITOR EVIDENCE COLLECTION

The Clouditor Evidence Collection is a component for collecting security evidence from cloud infrastructure. It must be configured with the respective access rights and then queries a cloud system regarding its existing resources and their configurations. Its functionalities are as follows:

- Collect cloud infrastructure data from AWS and Microsoft Azure cloud systems.
- Transform the data received into the MEDINA evidence format, including the relationship of the resources to the MEDINA ontology.

WAZUH EVIDENCE COLLECTION

Wazuh[11] is an open-source security monitoring tool for threat detection, integrity monitoring, incident response and basic compliance monitoring. In MEDINA, Wazuh is used to satisfy and verify security controls related to malware protection, logging, threat analytics and automatic monitoring.

Vulnerability Assessment Tools (VAT) is a framework that includes several vulnerability scanning tools. It can be configured to periodically scan the CSP's infrastructure and detect vulnerabilities and potential threats. VAT can be used to satisfy or gather evidence for controls related to vulnerability detection, use of encrypted communication, detection of new devices, etc.

Both Wazuh and Vulnerability Assessment Tools are integrated with other MEDINA components through a common component that collects evidence from both tools: *Wazuh & VAT Evidence Collector*, which forwards the gathered evidence to the *Security Assessment* component of Clouditor. The Evidence Collector currently implements gathering evidence from Wazuh for a limited number of metrics.

The basic integration with *Clouditor Security Assessment* is implemented, sending evidence through the gRPC protocol. Other types of communication between the *Wazuh & VAT Evidence*

[11] https://wazuh.com

collector and *Clouditor* include component registration and configuration instructions and are currently being developed and tested.

CLOUDITOR SECURITY ASSESSMENT

The Security Assessment receives evidence from one or more evidence collection tools and assesses them according to a set of selected metrics. As such, it provides the following functionalities:

- Receive evidence from different evidence collectors, e.g. Wazuh and Clouditor.
- Assess evidence using the policy engine OPA and the selected metrics which are received from the Orchestrator.
- Create an assessment result summarising the assessment and send it to the Orchestrator.

CODYZE

Codyze is a component, which combines evidence collection and security assessment. The modular MEDINA architecture facilitates this combination within components. Rather than inspecting infrastructure information, Codyze analyses source code. It examines the source code of an application and identifies security relevant code sections. Using predefined rules Codyze evaluates these code sections as either compliant or non-compliant with respect to the requirements specified in these rules. Evaluation results from rules act as evidence for concrete weaknesses in the source code of applications. Afterwards Codyze assesses all evidence according to pre-defined metrics. Finally, it forwards the resulting assessment results to the Orchestrator. As part of a CI/CD pipeline, Codyze can act as a quality and compliance assurance tool and prevent the automatic deployment of non-compliant applications. Its functionalities (see Figure 10) are as follows:

- Analyse source code of different programming languages regarding pre-defined security properties, e.g. the usage of cryptographic libraries.
- Evaluate compliance based on rules specifying secure coding practices.
- Assess evaluation results according to metrics to derive assessment results.
- Forward assessment results to Orchestrator.
- Act as compliance and quality gate in CI/CD pipelines.

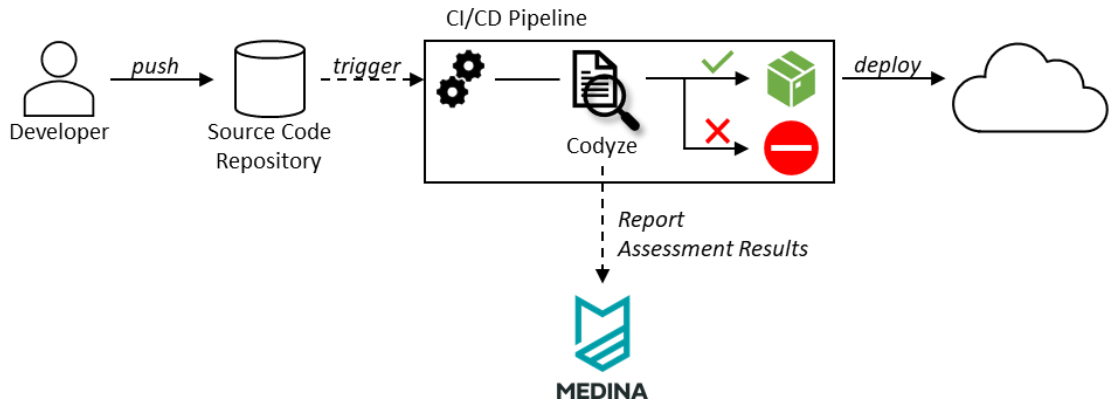

Figure 10. Codyze as compliance and quality gate in CI/CD pipelines

AMOE – ASSESSMENT AND MANAGEMENT OF ORGANIZATIONAL EVIDENCES

The AMOE component uses predefined organizational metrics to aid in assessment of organizational measures. This enables assessment hints for textual documents like internal policies. The UI could assist compliance managers or auditors to speed up monotonous tasks such as checking parts of the policies against target values and provide the necessary information to be able to compare the policy parts to their technical implementations. Once assessment hints are confirmed, the assessment results can be transmitted to the Orchestrator. The basic functionalities are as follows:

- Upload an organizational document to be processed (e.g. policy).
- Process the document and retrieve the evidence information for every possible organizational metric.
- Calculate assessment hints and retrieved evidences to user.
- Forward assessment results to Orchestrator.
- Assess results regarding a set of MEDINA metrics and create assessment results accordingly that are forwarded to the Orchestrator.

3.7 Certificate Management System

The SSI-based certificate management system provides the necessary tools for digitalizing the conformity assessment results report based on the information gathered by the MEDINA framework. It is an additional component to the general MEDINA framework.

On this sense, it is formed by the required Blockchain-based and the necessary services for the CAB, CSP and a potential CSP client, as shown in Figure 11.

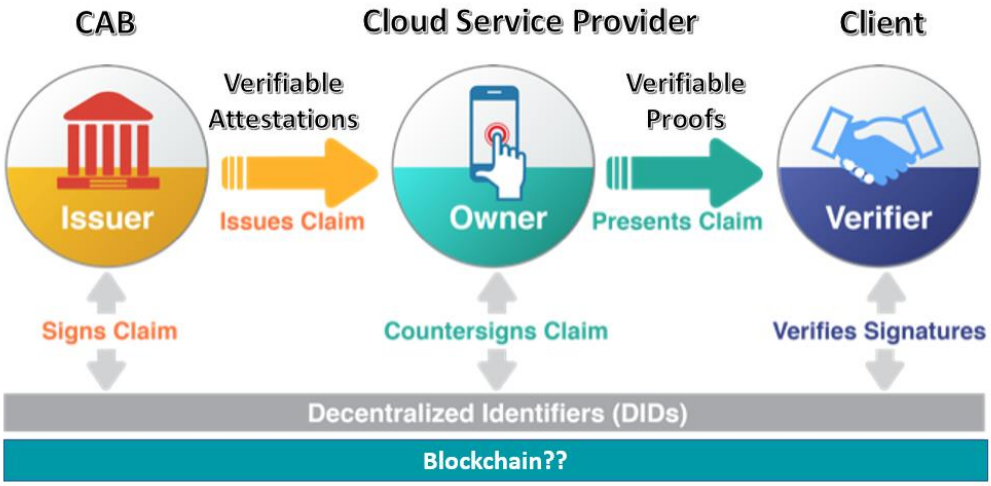

Figure 11. SSI applied to the MEDINA context

The CAB needs the following services:

- A service to listen for MEDINA framework even and know when to issue/update/revoke a certificate.
- A service to issue, update and/or revoke public and private attestations (verifiable credentials) on the CSP based on the MEDINA framework inputs.
- A service to automatically store public attestations in a public registry.

The CSP needs the following services:

- A service to receive public and private attestations (verifiable credentials) from the CAB and store them locally.
- A service to generate verifiable proofs to share with its clients based on verifiable credentials from the CAB.

The CSP clients need the following services:

- A service to request specific proofs (both associated to private or public attestations) from the CSP.
- A service to verify signatures from verifiable proofs.

3.8 Integrated User Interface

Each MEDINA component develops its own GUI, but end-users need a common thread to facilitate navigation through the content. The *Integrated User Interface* is this component that provides a main access point for the MEDINA Framework. It integrates with existing authentication, and guides different users, depending on their role and level of authorization, to specific component UIs.

In order to facilitate the development of frontend functionalities in independent teams, the architecture chosen is that of "micro-frontends"[12]. This type of architecture allows us to embed in a main frontend component (the *Integrated UI*) any UI of the framework, independently of the underlying technology. The following diagram describes a simplified architecture from Integrated UI perspective.

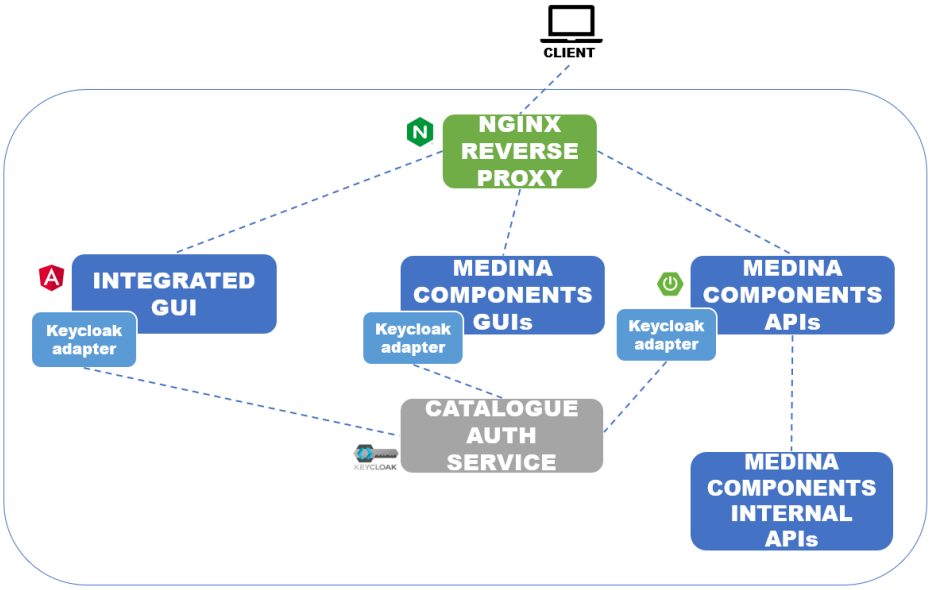

Figure 12. MEDINA UI Architecture

The prototype is developed using Angular 12[13], a modern typescript framework that allows us to build high-performance, scalable, component-based single page web applications. It is

[12] For more information, read "Motivations, benefits, and issues for adopting Micro-Frontends: A Multivocal Literature Review," from M. D. T. Severi Peltonen,. https://www.sciencedirect.com/science/article/pii/S0950584921000549

[13] https://angular.io/

enriched with Angular Material 2 library, a set of high quality animated responsive components that follow Material Design UI specifications[14]. The application runs on a Nginx web server.

Integration of micro-frontends is obtained through iframes and REST API. Ngx-charts[15] is used to render animated graphical content (e.g. Histograms).

AUTHENTICATION AND AUTHORIZATION

Authentication is managed by Keycloak[16], which is a standalone component based on an open source solution. It provides advanced authentication and authorization capabilities, including SSO, Identity Brokerage and role mapping. Every component implements a "Keycloak adapter" which acts as an HTTP interceptor and checks on resources requests whether:

- The client requesting user authentication is a registered client.
- The user is authenticated (if not, it redirects to login page) and his role is authorized for the requested resource (if not, it redirects to an error page).

Once a user is authenticated, a Jason Web Token (JWT) is provided, which contains user and role information. It allows to implement conditional formatting and routing based on the user's role. For example, a CSP wouldn't see what concerns to an Auditor accessing the same panel.

[14] https://material.io/design, https://material.angular.io/
[15] https://swimlane.github.io/ngx-charts
[16] https://www.keycloak.org/