Privacy-Aware Cloud Auditing for GDPR Compliance Verification in Online Healthcare

Masoud Barati, Gagangeet Singh Aujla, *Senior Member, IEEE*, Jose Tomas Llanos, Kwabena Adu Duodu, Omer Rana, *Member, IEEE*, Madeline Carr, Rajiv Ranjan, *Senior Member, IEEE*.

***Abstract*—Emerging multi-tenant cloud computing ecosystems allow multiple applications to share virtualised pool of computing and networking resources. As a result such ecosystems are becoming increasingly prone to data privacy concerns (personal data leakages and unauthorised access). While cloud computing providers support robust security and privacy mechanisms (e.g, public key cryptography, firewalls, virtual private networks, among many others), they lack mechanisms and frameworks to monitor, audit and verify these data privacy concerns. The emergence of data protection regulations around the world, such as General Data Protection Regulation (GDPR) in Europe and the Data Protection Act (DPA) in the UK, further emphasise the need to overcome these privacy limitations. A novel technique for monitoring, auditing and verifying the operations carried out on a user's personal data in cloud computing ecosystems is proposed. Our research methodology leverages distributed ledger technologies (e.g., Blockchain, Smart Contracts) for developing an immutable recording technique, which transparently logs, monitors and verifies the operations carried out on user data. Using a healthcare pharmacy scenario and extensive real-world experiments, we validate the feasibility of the proposed technique. The proposed work handles a large pool of requests (> 13K) ensuring minimal latency (≈50-60 ms) and overheads for three different service packages varied with respect to the number of actors and operations).**

***Index Terms*—blockchain, container-based monitoring, data privacy, healthcare, smart contracts**

I. Introduction

Cloud-hosted applications which allow users to run software through web browsers without the need to install any specific software on their own devices are growing in popularity. As a result, user (i.e. "data subjects" in GDPR) data needs to be migrated to the cloud, a trend that is unlikely to be reversed in the foreseeable future [2]. However, storage of user data coupled with the complexity of multi-tenant cloud ecosystems, raises significant governance and compliance concerns under the EU GDPR [3] and the UK DPA, for example. GDPR and DPA compliance concerns serve as a barrier to migration to the cloud, especially for small and medium scale companies, who rely more and more on public cloud data-centres, for application hosting, to benefit from economies of scale and on-demand provisioning of hardware and software services. As a result, they are more likely to "find it difficult to exercise the full control required by data protection legislation on how the cloud data-centre provider hosts the requested services" [1].

M. Barati is with the School of Computing at Edinburgh Napier University, UK. E-mail: m.barati@napier.ac.uk

G. S. Aujla is with the Department of Computer Science, Durham University, UK. E-mail: gagangeet.s.aujla@durham.ac.uk

J. T. Llanos and M. Carr are with the Department of Science, Technology, Engineering and Public Policy, University College London (UCL), London, UK. {j.llanos,m.carr}@ucl.ac.uk

K.A. Duodu, and R. Ranjan are with the School of School of Computing, Newcastle University, Newcastle, UK.
E-mail: kwabenaaduduodu@gmail.com, and rranjans@gmail.com

O. Rana is with the School of Computer Science & Informatics, Cardiff University, Cardiff, UK. E-mail: ranaof@cardiff.ac.uk

Corresponding Author: Gagangeet Singh Aujla

A. Research Problem

Cloud-hosted application solutions are typically "layered", involving a chain of cloud infrastructure (e.g., Amazon Web Services, Microsoft Azure) and software providers (e.g., payment gateways and customer relationship management services). In this architecture, it can be difficult for the application providers, "data controller" (based on GDPR terminology), to verify data handling practices of a cloud providers, and thus to be certain that data is being handled in a lawful manner. Similarly, due to a loss of governance and control, a data controller may be incapable of providing evidence of compliance with *inter alia* the data location and transfer, "privacy-by-design" and accountability requirements laid down in GDPR (Arts. 44–47, 25 and 5(2)) [10].

On the other hand, users (i.e. "data subjects" in GDPR) are seldom aware of the highly intricate and layered architecture of cloud ecosystems [5]. They typically interact only with a Web interface rather than the larger, composite ecosystem of service providers, entrusting their personal data and identity to the consumer-facing component without realising that the cloud-based application may share their data with several additional providers (e.g. online advertising services and payment gateways). In this opaque context, it is hard for data subjects to exert any control over their personal data after the initial disclosure. Also, when multiple infrastructure and software providers are involved in a cloud-hosted application solution, the risk of personal data being processed for additional purposes can be high [1]. Crucially, the lack of transparency that characterises cloud ecosystems impede the data subject's ability to give "informed consent" to the use, sharing and repurposing of their personal data. Users experience a lack of control over their personal data, exacerbated by the growing number of data breaches resulting from cyberattacks on cloud systems. This threatens to undermine consumer trust and hinder the use of infrastructure and software services offered by cloud.

1) Research Questions: To realise the full potential and benefits of cloud computing, the data protection concerns must be effectively addressed. To this end, cloud ecosystems

should be both transparent and auditable. However, to achieve this, we need to answer following research questions:

- how do we develop an approach that ensures that data subjects and data controllers can verify who possess their personal data and with whom it is/was shared?
- how to develop an approach that can transparently record operations on personal data and support policies to enable data subjects and data controllers to identify and verify operations that are permissible on the data?
- how to develop an immutable recording mechanism to log and verify the operations carried out on user data based on the GDPR legislation?

B. Overview of Research Approach and Contributions

We develop a novel privacy-aware auditing approach and related framework (see Fig. 1) consisting of following novel features: As an automated approach is needed for logging data operations in the cloud ecosystems, our approach includes a novel monitoring technique. To simplify the portability of monitoring framework across multiple cloud service providers that employ heterogeneous virtualisation approaches, we leverage existing container-based application-level virtualisation technologies, e.g., Docker, for implementing the monitoring technique (see section IV-A). We call this as user's "data container" in the remainder of this paper. Along with monitoring feature, the user's "data container" also needs to have capability to record the data operation while ensuring immutability. To this end, we present a novel approach (see section IV-B) that applies Distributed Ledger Technologies (DLTs), such as Blockchain and smart contracts, to enable an immutable audit (provenance) trail [11] and verification of GPDR compliance on monitored data operations. This approach takes into account of user preferences for verifying obligations and access control policies. The key contributions of this work include:

- an online pharmacy scenario to motivate how GDPR requirements arise when combining a number of cloud-hosted services;
- a data "container" (built on existing open source approaches) that can be migrated across cloud providers, and used to record and monitor events/operations from a provider (referred to as a "data controller" in GDPR);
- a model and related smart contracts for verifying GDPR obligations taking account of user preferences – i.e. not all user data may need to confirm to GDPR requirements.
- an implementation of smart contracts in a Blockchain network, evaluated using a Blockchain test network.

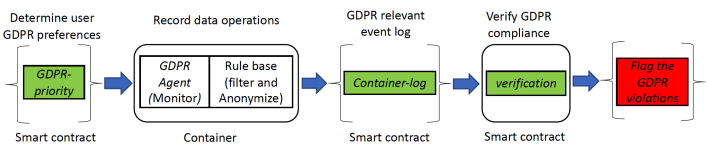


Fig. 1: Overview of the research approach

The remainder of this paper is structured as follows: Section II describes state-of-the-art proposals associated with data privacy & GDPR. Section III presents the online pharmacy scenario used to contextualise this work. Section IV presents design of the container-based architecture making use of smart contracts. Section V describes experimental results of our Blockchain-based technique, and finally Section VI describes conclusions and lessons learned.

TABLE I: Comparison of existing monitoring tools

Monitoring tool	Performance	Compliance
CAdvisor	✓	×
CLAMS	✓	×
Cloud Watch	✓	×
M2CPA	✓	×
M3	✓	×

II. Related work

There are several monitoring tools and architectures (like, CAdvisor[1], CLAMS [27], Cloud Watch[2], M2CPA [29], and M3 [28]) that are deployed in the cloud ecosystem. But, these monitoring tools either do not support compliance monitoring or they are not compatible with container-based virtualisation. A comparative anlaysis of the exisiting monitoring tools is presented in Table I. Hence, a novel monitoring architecture is required that can monitor data operations undertaken in the cloud ecosystem and later verify them for GDPR compliance. In this regard, a novel architecture compliance-aware cloud application engineering was proposed in [30]. This proposal highlighted that blockchain or DLTs can be very useful to design an end-to-end GDPR monitoring, auditing and verification architecture for the privacy-aware cloud services. A procedure for designing privacy-aware cloud-based applications was described in [18] under which applications are published through a GDPR compliant software architecture using containers. The architecture, however, does not use Blockchain technology for verifying GDPR rules in an automatic way. In [17], a container-based architecture was proposed to support real time identity and provenance verification within cloud & edge computing. The architecture uses a permissioned Blockchain to track actions of devices/services on user data. The approach lacked a monitoring strategy for GDPR compliance verification.

A Blockchain-based approach that improved data accountability and provenance tracking was proposed in [20]. The approach used GDPR and proposed two models for running enforced smart contracts: (i) user consent was recorded in a Blockchain; (ii) privacy policies of providers appear in a smart contract and permit users to join or leave the contract. However checking compliance of consent rules was verified manually. A combination of GDPR ontology and Blockchain was proposed in [21] to ease real-time automated data compliance. The approach ensured that data protection is permitted, to ensure that data privacy policies are compliant with GDPR rules. However, use of the approach within cloud-based systems was not considered. A Blockchain-based platform was designed in [22] that uses GDPR principles for data provenance and transparency in a cloud environment. The platform utilises user consent to ensure that only trusted

[1]https://github.com/google/cadvisor.
[2]https://aws.amazon.com/cloud- watch.

cloud providers can use personal data. The activities of such providers are recorded in a Blockchain. But, GDPR compliance verification of cloud providers was not considered.

A forensic (evidence) data system using distributed cloud resources was proposed in [23] for GDPR compliance checking. The approach provided the availability and integrity of the forensic data using Blockchain. The translation of GDPR obligations (e.g. data protection and data transfer) into smart contracts was not presented. Moreover, the approach did not provide automated verification of GDPR obligations over the activities of providers. In [25], the authors analyzed GDPR obligations that are necessary for tracking operations of cloud providers on user data through a Blockchain. However, a solution for checking the compliance of such obligations in an automatic way was not provided. The authors in [26] proposed a formal representation for verification of GDPR obligations. Such obligations are encoded in smart contracts to provide an automatic approach for data accountability. The approach, however, did not consider the preference of users for verifying obligations (an essential requirement to ensure scalability of the approach). Furthermore, it also lacked a suitable container or virtual machine-based environment.

In [13], a number of GDPR rules were translated into smart contracts in order to automatically verify legal compliance for operations executed by providers on data of cloud customers. However, both data minimisation and data protection obligations were not fully studied. In general, although the foregoing approaches make use of GDPR and Blockchain for improving data privacy, these do not consider preferences of cloud customers for verifying GDPR obligations. Moreover, they cannot be directly utilised alongside existing container technologies, e.g. Docker, Kubernetes. A two phased framework supporting heterogeneous privacy for personal data was proposed in [9]. The nodes hosting user data utilised one-shot noise perturbation to reach heterogeneous privacy protection over various data servers. Moreover, if data servers have fixed budgets, an efficient incentive strategy was presented for optimising computation accuracy. However, verifying compliance of the framework with privacy regulation was not studied. Moreover, a practical implementation of the proposed framework using privacy-aware solutions was not described.

III. An Online Pharmacy

A cloud-based pharmacy scenario (inspired by the DermaTran app. by *dinCloud* [12]) is used to motivate the importance of GDPR compliance verification in a multi-cloud system. Consider a user who visits a (cloud-hosted) online pharmacy to place an order, make a payment and receive a prescription to a home address. After order placement, the pharmacy accesses various personal user data (such as name, address, general practitioner (GP) diagnosis, the e-prescription and bank account details), and extends the user Electronic Health Record (EHR). The data is transferred to an IaaS vendor (Cloud4U); the pharmacy uses Cloud4U to host and operate its website and mobile app. The pharmacy's website and mobile app are embedded with "social plugins" (a "Like" button and a "Share" button) from a leading social network (Friendface). The Friendface API is designed to automatically transmit users' personal data (such as IP address and location data) from the online pharmacy to Friendface when a user visits the pharmacy's website or opens the app. Friendface uses this data to enrich a patient's profile that it has built over time (i.e. profiling), which is valuable for its advertising business. Also, the online pharmacy uses a real-time bidding (RTB) system of a prominent online advertiser and intermediary (Froogle) to sell advertising inventory space on its website and mobile app., to derive another revenue stream. This system involves extensive profiling and dissemination of personal data (such as location, device properties, unique tracking ID and browsing habits) to multiple companies (actors) in the ad tech chain. The broadcast of personal data is made by Froogle's "Supply Side Platform" on behalf of the pharmacy, to solicit bids from companies which may want to show an ad. to the pharmacy's user.

The pharmacy makes use of subcontracts for payment and shipping service providers to handle the payment and delivery of prescriptions. The payment service provider receives the patient's name and bank account details from the pharmacy, offering two alternatives, Western Union and Paypal, to manage the payment process. The shipping service provider, in turn, receives the patient's name and address details from the pharmacy, whereupon it packs the order, sends the patient a reference number to track the parcel, and delivers it. Each data flow in this cloud-based ecosystem constitutes "personal data processing", thus warranting an assessment under the GDPR legislation. In this regulatory framework, the online pharmacy is the data controller, whereas all other cloud components act as data processors. Determining the roles is a preliminary step, as controllers and processors are subject to different obligations under GDPR. For example, data protection rights can be enforced against controllers only.

Based on the data flows and roles explained above, we consider the following GDPR requirements:

- Transfer of personal data to a non-EU country. There are two main ways of allowing international transfers of personal data: on the basis of an adequacy decision by the European Commission, or in lieu thereof, where the controller or processor provides appropriate safeguards, including enforceable rights and legal remedies for the data subject [10], Art. 45. These safeguards can be established by a legally binding and enforceable instrument between public authorities (Binding Corporate Rules); standard data protection clauses adopted by the European Commission or a supervisory authority; codes of conduct; certification mechanisms [10], Arts. 46 & 47.
- The principle of data protection requires that appropriate technical or organisational measures are implemented when processing personal data to protect it against accidental, unauthorised or unlawful access, use, modification, disclosure, loss, destruction or damage [10], Arts. 5(1)(f) and 32(1). Depending on the circumstances of each case, these measures may include pseudonymising and encrypting personal data.
- Under the data minimisation principle, the personal

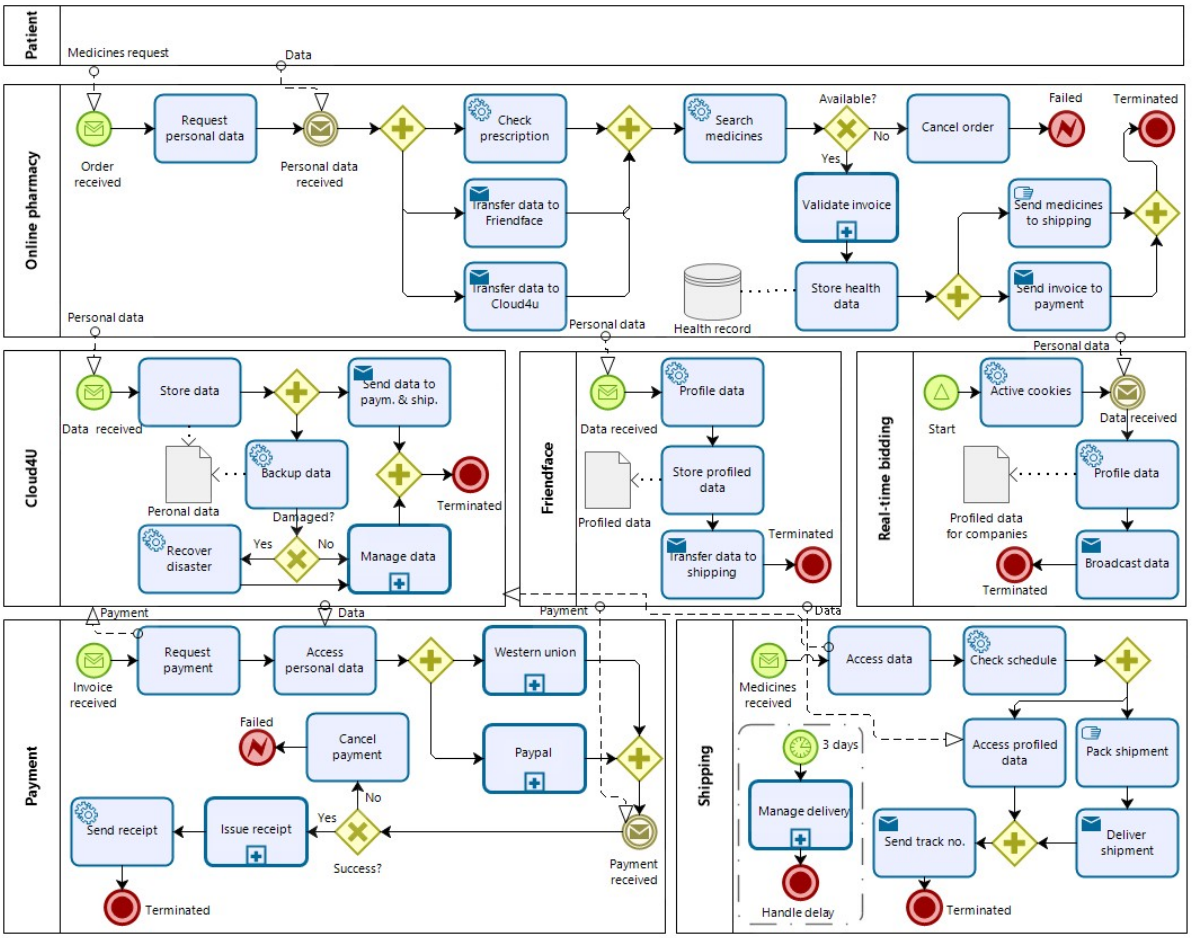


Fig. 2: A business process model for online pharmacy

data being processed must be limited to what is necessary [10], Art. 5(1)(c). Whether the data is necessary will depend on the controller's specified purpose for collecting and using the personal data.

- For data storage obligation, Art. 5(e) of GDPR does not allow actors to keep personal data longer than the time required for data processing. Moreover, the encryption of such data in the local storage of actors should be guaranteed.

The proposed architecture is not limited to a specific case study and can be applied to any cloud-based application.

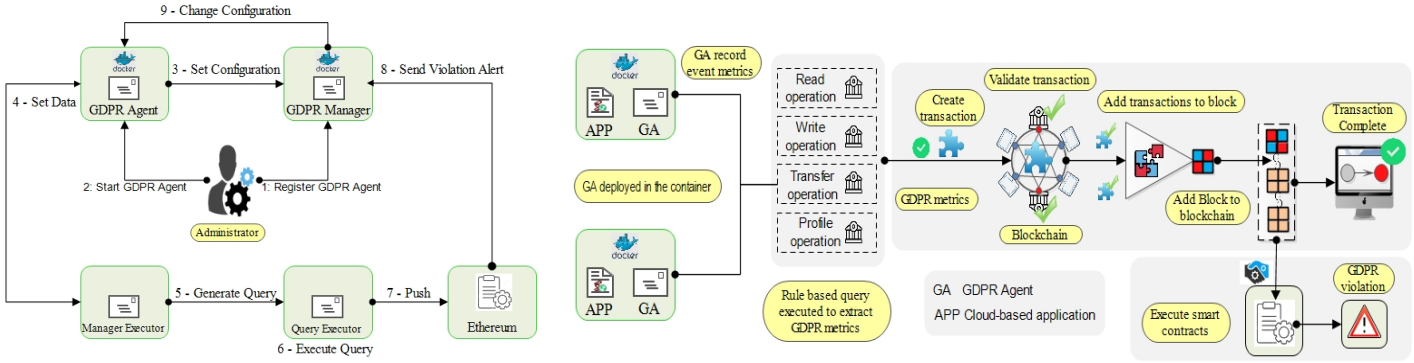
IV. Architecture

We propose a multi-tier container-based monitoring architecture where operations related to the pharmacy scenario are executed inside a container hosted over multiple cloud service providers. Container-based virtualisation is popular for reasons such as efficiency, ease of application deployment and less overhead (compared to use of virtual machine). Also, containers provide portability, where applications running in containers can be migrated easily to different platforms. Additionally, containers support monitoring at activity and infrastructure levels making it more suitable for multi-cloud system. At infrastructure level, containers should be compliant with organisational rules and enable the functionality for logging, checking, and remediation for these rules. At activity level, white-box monitoring and tracking at run-time are performed to trace internal systems vulnerabilities [7].

The proposed architecture consists of three functional levels: 1) service provisioning, 2) monitoring, and 3) blockchain. At the service provisioning level, whenever any customer places an order on the online healthcare pharmacy website/app, the data exchange between the customer and the cloud provider takes place in a containerised environment. The controller (Cloud4U) receives the request and distributes the service compositions to different processor cloud providers (Western Union/Paypal or shipment provider). Once the service is initiated, the monitoring system is activated. The monitoring system contains two main components, 1) GDPR-Agent, and 2) GDPR-Manager, which are responsible for monitoring the data operations taking place on the data in question throughout its lifetime and recording them on the blockchain, to analyse potential GDPR violations using smart contracts.

A. Monitoring System

Cloud services generally use virtualised environments to support multiple co-hosted services over the same cloud infrastructure [8]. In cloud architectures, the monitoring process is used to track any unwarranted or anomalous events and store them for further verification (if required). However, this is generally limited to performance monitor-



(a) Registration and data collection. (b) Architecture.

Fig. 3: Monitoring System

ing, and there are limited automated mechanisms for GDPR compliance verification. To overcome these limitation, we propose a new GDPR monitoring agent. The GDPR-Agent is a monitoring agent which is executed inside a container to track data operations in the container-hosted online services. The GDPR-Agent understands the underlying heterogeneity of the containers deployed on a multi-tier cloud infrastructure. The main task of the GDPR-Agent involves the collection of data operation statistics (e.g. read, write, transfer, profile) sending them to a collection engine (running inside the GDPR Manager) that further sends them to a filtering engine (query executor). The GDPR query is executed in the filtering engine, whereas the GDPR-Manager runs on the controller and administrates all the GDPR-Agents. The monitoring architecture provide a fault tolerant mechanism as it is quick and easy to replicate or clone an existing container in case the manager or agent fails or compromised. The working is described below.

1) GDPR-Agent: GDPR-Agent captures events based on data operations being performed inside a container in a cloud-hosted service. The GDPR-Agent (also known as *SmartAgent*) receives data from various sub-agents (*Read*, *Write*, *Transfer*, and *Profile Agents*).

The GDPR-Agent follows three operations – 1) *Register()*: a registration request is sent by the agent to the manager, 2) *SendData()*: all events related to data operations are sent by the agent to a collection engine periodically, and 3) *SetConfiguration()*: the agent sends its configuration information to the manager, which can update configuration parameters of the agent. Initially, the agent has to register with the GDPR-Manager as per the process shown in Fig. 3a.

2) GDPR-Manager: The GDPR-Manager receives registration requests for all the GDPR-Agents (deployed inside containers). Once the GDPR-Agent is registered with the GDPR-Manager (step 1), the *ADMIN* starts the agent (step 2). The access key and endpoints are sent to the GDPR-Agent by the manager. The GDPR-Agent sends its current configuration to the manager (step 3). Thereafter, the agent sends the monitored events to the collection engine (running inside the GDPR Manager) through Manager Executor (step 4). The GDPR manager acts as a gateway to the other internal systems of the proposed monitoring model. The proposed manager-client architecture is based on the the REST architecture style using the *Restlet* framework, due to its powerful routing and filtering capabilities. The manager is implemented as a restlet server with API functionalities to allow users working as clients to connect with the overall system. Users can only communicate with the manager that controls the internal running processes, working as REST clients to make requests to the manager to either read or update their data stored in proposed ecosystem. The clients are third parties that connect to manager to make request either as individual users taking advantage of the designed functionality for GDPR compliance or service providers.

Data operations (such as read, write, transfer, profile) collected from the container are sent to the GDPR Manager, where they are filtered to extract GDPR-relevant metrics. The data received from the monitoring agent is exposed to rule-based queries activated inside a filtering engine through a query executor (step 5). These queries filter out GDPR-specific metrics from the data collected by GDPR-Agents (step 6). GDPR metrics are then transmitted to the Ethereum Blockchain as *container-logs* using a push-based mechanism (step 7). Precisely, the GDPR metrics submitted to the Blockchain are: the anonymised version of provider address, the operations (e.g., read, write etc.) executed on personal data, the processed personal data items (e.g., name, ID etc.), the collected personal data items from user, the encryption status of operations, the location of provider, and the retention period claimed by provider for storing personal data. Note that the values of personal data items are not sent to the Blockchain to ensure compliance with GDPR. A verification smart contract is then executed over the *container-log*. If there are any GDPR violations, the GDPR-Manager is updated (step 8). The manager updates the configurations of the agents if required (dynamic configuration enables real-time communication) (step 9). To communicate with the Blockchain, the manager has to create an Ethereum account.

3) Monitoring Process: The monitoring process involves different steps and programming operations as shown in Fig. 3b. These steps and related operations are described below.

- All GDPR-Agents are deployed in the container of the user and cloud providers; the data operations are recorded and sent to the collection engine.
- These events are added into different queues based on their type or characteristic (e.g. read operations added to a Read queue) via REST messages.
- A rule-based query engine filters out data containing

GDPR metrics. This ensures that events that are not needed to validate GDPR compliance do not get added to the Blockchain, improving the efficiency of the verification process and reducing performance overhead.
- Finally, this *container-log* is added to the Blockchain as a transaction. For this reason, the manager (which already has an Ethereum account) creates a genesis (i.e. first) transaction to which data, the timestamp, the signature, and the public key of the controller cloud provider are appended.
- This is followed by the creation of the genesis block (B) with a unique ID; B: $[B_{ID}]$. The genesis transaction (T1) is added to the block (B) after validation by the trusted nodes. The user creates new transactions in the same manner until the block achieves the maximum block size (fixed) or threshold (variable).
- Finally, the block is added to the blockchain where the smart contracts are deployed to verify the GDPR obligations over the *container-log*.

B. GDPR-supported smart contracts

Smart contracts such as *GDPR-priority* and *container-log* are proposed to record the information required for checking GDPR compliance. The *verification* smart contract is deployed to verify compliance with the aforementioned obligations (data protection, data minimisation, data transfer and data storage) by cloud providers. The activators of *GDPR-priority*, *container-log* and *verification* are the data subject (user), the smart manager, and the verifier. Note that *verifier* is a trusted third party connected to the Blockchain in charge of flagging GDPR violations.

C. Determining GDPR Compliance Preference

Verifying compliance with GDPR obligations (i.e. data protection, data minimisation and data transfer) is a costly process. Therefore, a data subject may not have the computational resources to verify GDPR compliance. This phase enables data subjects to activate the *GDPR-priority* contract in order to specify their preferences for verifying compliance with obligations. The smart contract allows the data subject to give a compliance score, identifying a "preference" value for each obligation.

Definition 1. Let Σ be a set of GDPR obligations. A compliance score is a function $\mathcal{S} : \Sigma \mapsto [0, 1]$. A compliance score $\mathcal{S}(\sigma) = 1$ indicates a requirement for full verification of an obligation, where $\sigma \in \Sigma$; a score $0 < \mathcal{S}(\sigma) < 1$ represents the partial verification of σ, and $\mathcal{S}(\sigma) = 0$ shows that no verification of σ is needed.

The contract stores such scores into a Blockchain to inform the verifier about the data subject's preferences concerning the verification of GDPR obligations. For example, *data protection* is likely to be the greatest concern of the customers of the online pharmacy depicted in Fig. 2, since their personal data involve sensitive healthcare information.

D. Recording Data Processing Operations

This phase collects and sends a number of useful information for verifying GDPR compliance by cloud providers. The GDPR-manager of the container deploys the *container-log* smart contract to send such information to a Blockchain. The information includes (i) the anonymised provider's address (p), (ii) processed operations (A_p) on user data, (iii) processed personal data items (D_p), (iv) collected personal data items from user (D_{c_p}), (v) encryption status of operations ($\mathcal{E}_{a_p}$), (vi) the physical location of the provider (loc_p), and (vii) the period of time claimed by provider for storing/ processing (t_p) user data.

A permutation technique is used to anonymise information kept in the Blockchain, such as provider address (IP). As there is no specific GDPR compliance requirement for other information, e.g. operation, encryption status etc. these be deployed on any Blockchain environment (i.e., permissioned or permissionless).

Thereafter, data operations are recorded by the GDPR-Agent; the filtering engine is triggered to extract only events that meet GDPR obligations defined by Σ.

Definition 2. Let $\mathcal{E}$ be a set of data operations and multi-level statistics recorded by the GDPR-Agent. Data operations ($\mathcal{E}_\Sigma$) based on GDPR obligations Σ are filtered using the function $\mathcal{F} : \mathcal{E} \mapsto \{0, 1\}$, so that

$$\mathcal{E}_\Sigma = \begin{cases} 1: & \text{If } \mathcal{E} \in \{\text{read, write, transfer}\} \\ 0: & \text{Otherwise} \end{cases}$$

E. Verifying GDPR Compliance

This phase verifies GDPR compliance of operations carried out by cloud providers (data controllers/processors) on personal data.

Definition 3. A preference for checking obligations is a strict partial order relation, denoted by $\mathcal{P} = (\Sigma, \mathcal{R}_\mathcal{P})$, where $\mathcal{R}_\mathcal{P} \subseteq \Sigma \times \Sigma$. If $\sigma, \sigma' \in \Sigma$ are two different GDPR obligations, then $\sigma \; \mathcal{R}_\mathcal{P} \; \sigma'$ is expressed as "σ' is preferred rather than σ".

Given a set of obligations selected by the data subject and their verification time, the following definition gives the verifier a priority for detecting violations in accordance with the preferences of the data subject.

Definition 4. Let $\sigma, \sigma' \in \Sigma$ be two GDPR obligations, $\mathcal{S}(\sigma)$ and $\mathcal{S}(\sigma')$ be the compliance scores determined by the data subject. Checking the obligation σ' is preferred over checking σ (i.e. $\sigma \; \mathcal{R}_\mathcal{P} \; \sigma'$) *iff* $\mathcal{S}(\sigma) < \mathcal{S}(\sigma')$.

Algorithm 1 Checking GDPR compliance

$V \leftarrow \emptyset$
case *data protection*
  $V \leftarrow V \cup \{p \in P \mid \exists\ a_p \in A_p \;\; and \;\; \mathcal{E}_{a_p} = \perp\}$
case *data minimisation*
  $V \leftarrow V \cup \{p \in P \mid D_{c_p} \not\subseteq D_p\}$
case *data transfer*
  $V \leftarrow V \cup \{p \in P \mid \exists\ p' \in P \; and \; (\mathcal{E}_{a_p} = \perp \; or \; loc_{p'} \notin BCR)\}$
case *data storage*
  $V \leftarrow V \cup \{p \in P \mid \mathcal{E}_{a_p} = \perp \; or \; t_s > t_p\}$
return V

The verifier deploys the *verification* contract to report providers breaching the aforementioned obligations (i.e. data protection, data minimisation and data transfer). The contract involves a function presented in Alg. 1 to flag GDPR violators that are extracted from a set V.

Data protection case: A provider p from the set of cloud providers P, executing a set of operations A_p on user data, is a violator if it has an operation a_p that does not encrypt personal data.

Data minimisation case: A provider p is reported as a violator if the data D_p processed by p is a subset of the data set D_{c_p} requested by p. In other words, a provider collecting data that is not used for processing is classified as a violator.

Data transfer case: A provider p is reported as a violator if personal data is transferred to a provider p' such that the country of p' (denoted by $loc_{p'}$) is not included in the *BCR* set. This set includes the names of non-EU countries having an *adequacy decision* with the European Commission, and the names of enterprises which adhere to duly approved data protection policies to move personal data internationally over different jurisdictions [6]. The encryption of the data transfer operation is also checked through the contract.

Data storage case: A provider p is a violator if the retention period (t_s) for keeping data is longer than the one (t_p) claimed for data processing by a provider (Art. 5(e)). Also, any data storage is subject to encryption of personal data required by GDPR. The contract checks both conditions $\mathcal{E}_{a_p} = \top \ \& \ t_s \leq t_p$, where a_p is *write* operation. The verification contract checks obligations in a decreasing order based on compliance scores provided by a data subject.

V. Experimental Setup and Validation Results

In [4], we formally verified a cloud-based order system via Uppaal in accordance with a GDPR obligations under which processing activities violating the obligations were detected at design time. However, the implementation of such verification through practical technologies/ tools (e.g., Blockchain and container) was not discussed. In this work evaluation is separated into two sections: section V-A evaluates the monitoring system and section V-B assesses transaction costs associated with the blockchain model.

A. Validation of Monitoring System

The monitoring system is implemented in Java using a container-based environment running on a host operating system. The host device is an M1 MacBook Air, 2020 running MacOS BigSur v11.3.1 OS with 8 core CPU (4 performance and 4 efficiency cores) with 8GB memory. For test purposes, a private blockchain network is set up using Ganache as: 1) Ganache is able to quickly start a local blockchain network for testing. 2) Ganache provides a number of test blockchain accounts (default of 10), preconfigured with 100 *ethers* each. This aligns with requirements of our test scenarios which involve a high number of requests and hence more ether consumption. GDPR agents are implemented using the SIGAR and RESTLet libraries to ensure compatibility with multiple cloud providers. The monitoring system uses SIGAR to collect various system overheads, namely CPU usage, memory usage, latency, and throughput. The RESTLet Framework includes a unified client and server API that enables developers to build secure and scalable RESTful services. The manager is implemented as a RESTLet server, and users implemented as REST clients.

The overheads are measured to validate the efficacy of the proposed monitoring system with respect to downtime. The metrics (delay, CPU usage, and memory usage) are collected using a Docker-based container environment that starts running when the manager process is started. This helps to separate the metrics collection from other running manager processes, thereby preventing these process from influencing the response times for client requests and other processes. Metrics are collected every second and the information is stored in the *sys_overheads* table. Given the online pharmacy from Fig. 2, three different workload scenarios or Service Packages (SP) are used, as described below. These workloads are representative of actual services that can be deployed to verify GDPR compliance.

- Service Package 1 (SP1) contains two actors (Blockchain peer nodes) with 9 operations executed on user personal data.
- Service Package 2 (SP2) involves four actors with 16 operations carried out on personal data.
- Service Package 3 (SP3) has six actors with 23 operations executed on user data.

The aim of such packages is showing the scalability of our approach. Overhead information is collected for each of the SPs, as indicated in the following subsections.

1) CPU usage: The variation in CPU usage for recording logs for data operations is depicted in Fig. 4 – with events recorded every second. For SP1, CPU usage lies between 5%–20%, increasing with 2 instances to 24.5%. For SP2, the variation of CPU usage can be inferred in 3 parts, a) for first 60 instances it ranges between 8%–17%, b) for instances between 60–105, it ranges between 18%–40%, and c) for all instances from 105 onward till 210, the ranges is again similar to part 1, i.e., 8%–17%. We observe that CPU usage for some instances can increase by up to 30%. While analysing SP3, it is observed that CPU usage remains between 8%–17% throughout with two exceptions, a) from instance 90–130 the range lies between 15%–35% with one instance increasing by up to 40%, b) the last 10 instances also shows an increase in CPU between 10%–22%. It can be inferred from the above results that variation in CPU usage is constrained to a limited range, and only in restricted cases it shows growth due to larger size of recorded data operations. The average CPU usage for these cases, a) SP1 = 11.96%, b) 11.67%, and c) 11.53%. Although the average CPU usage is comparable, fewer operations are logged for SP1 compared to the other SPs. Hence, CPU usage is comparatively higher than the other two cases if we consider the number of operations logged for each SP.

2) Memory usage: The variation in memory usage during the monitoring process is illustrated in Fig. 5. In SP1, the highest memory usage reaches just above 3420MB, whereas for SP2 and SP3 it is 3500MB and 3550MB respectively. It can be inferred that peak memory usage for all three SPs are not very different (between 3420MB–3350MB), primarily

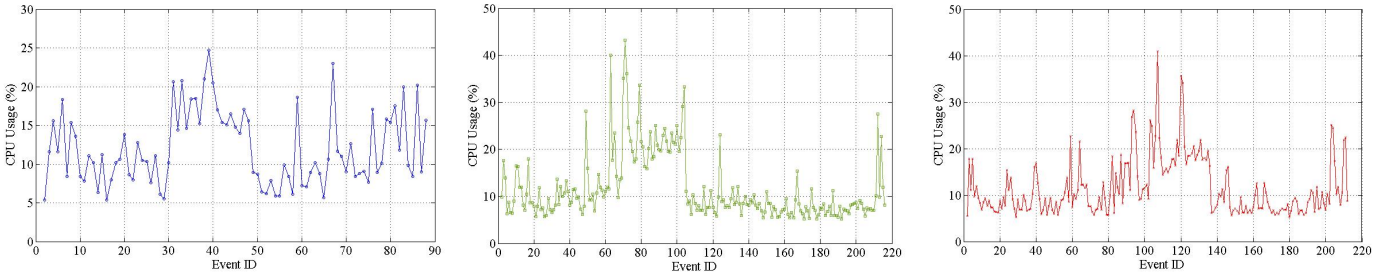


Fig. 4: Variations in the CPU usage for a) SP1, b) SP2, and c) SP3

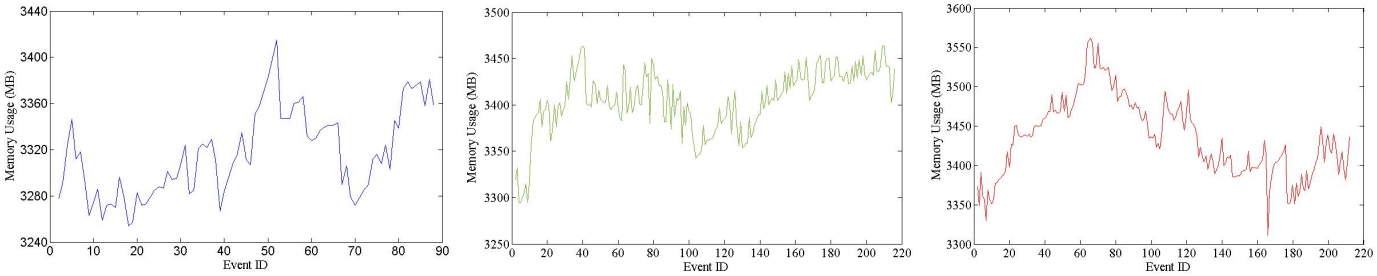


Fig. 5: Variations in memory usage for a) SP1, b) SP2, and c) SP3

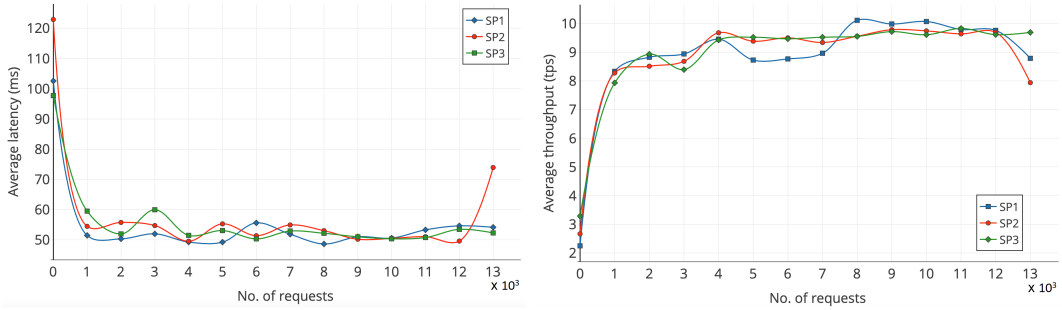


Fig. 6: Variations in a) Latency, and b) Throughput

affected by the considered data operations. It is further supported by the average memory usage. The average memory usage for these cases, a) SP1=3316MB, b) SP2=3404MB, and c) SP3=3434MB. Memory usage for SP1 is lower than the other two cases because the number of operations logged are less in contrast to the other SPs.

3) Average Latency: The variation in latency during the monitoring process is illustrated in Fig. 6a. Latency is measured as the time difference (in ms) between submission and completion of a blockchain logging transaction. We have evaluated the latency for all three service packages (increasing number of actors and operations) with respect to an increase in the number of requests. For simplicity, we have plotted the average latency at each instance with respect to an increase in the number of requests. For a test set of 13K requests, the latency for log submission for all three service packages was found to be ≈58ms, ≈63ms and ≈52ms, respectively. Initially, the average latency is higher as it takes some time to adjust to the system setting. But, as it reaches 450 requests, the average latency stabilizes and remains within the range of ≈50-60 ms. But, after 12K requests, there is an abrupt change in latency for SP2 but for all other service packages it remained stable.

4) Average Throughput: It is calculated by finding the start and end time for the processing of a set of requests sent to the monitoring framework and then dividing the number of requests by the time interval. We have evaluated the throughput for all three service packages with an increase in the number of requests. Here also, for simplicity, we have plotted the average average at each instance with respect to an increase in the number of requests. For 13K requests, the average throughput for all three service packages was found to be 8.7tps, 7.7tps and 9.6tps, respectively. After 12K requests, there was an abrupt change in throughput for SP1 and SP2 but it was stable for SP3.

5) Scalability: The scalability of the monitoring tool was assessed by increasing the number of actors and operations (different for each service package) as well as the number of requests (increased to 13K requests). The results consider how throughput and latency are affected – illustrated in Fig. 6. The results are system dependant and the specifications for the host device used to carry out benchmarking are as given at the start of this section.

In Fig. 6a, the measured average latency for all three service packages is maintained at ≈50-60ms, indicating that increasing the number of monitoring processes (including actors and operations) does not affect the speed at which

records are logged in the blockchain. The reason for this behaviour is due to an absence of concurrency in blockchain transactions (for a given account due to the nonce), as such, transactions are processed sequentially. Multiple logs can be generated via separate accounts simultaneously. However, we witness a sudden change in latency for SP2 after 12K requests in contrast to the others.

The variation in throughput for the various test scenarios is illustrated in Fig. 6b. The throughput increases as number of requests increase for all three service packages. The rate of increase starts to taper off as the saturation point of this tool is being reached. When the number of requests reach 450, the average throughput reaches a value of ≈8.3 for SP1, ≈8.2 for SP2, and ≈8 for SP3. These test results show that this monitoring tool handles an increasing number of requests without degradation in the quality of service for each service package. After this, the throughput becomes stable (with no much variation), as it was experimented till the number of request reach 12K. After this, there is an observable degradation in the value of throughput.

The lines in the plot overlap at some instances due to the average and randomness of the experiments.

B. Validation of Smart Contracts

After the data (or event) operations are collected and logged in the Blockchain using the proposed monitoring system, the smart contracts are executed in the Blockchain to verify the GDPR compliance. For this reason. the experiments provided in this section investigates the transactions' costs required to deploy and run our proposed smart contracts in Blockchain networks. A prototype has been developed via Ropsten [15], which is a public Blockchain test network. The prototype provides the number of operations needed when executing transactions, represented as the amount of *consumed gas* when executing transactions and the time the mining process takes. Gas is a unit in *wei* that measures the amount of computational effort needed for executing operations in smart contracts. In other words, it is an internal currency in Ethereum to pay for transaction fees. Gas is paid in *ether* – a cryptocurrency in Ethereum permitting smart contracts to be executed. Though the amount of gas consumed for running a transaction is high, its translation to ether unit is low. For example, if the consumed gas of a transaction is 10000 *wei*, the transaction fee is around 0.0002 *ether*.

In this experiment, our proposed smart contracts were implemented on Ethereum [14] using Solidity. Our approach creates smart contracts with a minimum gas consumption for each function. The contracts were compiled by Remix, a browser-based compiler for Solidity using an Ethereum Virtual Machine. The smart contracts *GDPR-priority*, *container-log* and *verification* were executed in the Ropsten network. The amount of gas consumed for deploying a contract was 371045 *wei* for *GDPR-priority*, 1146256 *wei* for *container-log* and 1642780 *wei* for *Verification*.

1) Transaction costs: This evaluation calculates the amount of gas consumed in the execution of our proposed smart contracts. It assumes that we have three SPs for the pharmacy scenario represented in Sect. III. The proposed smart contracts were deployed in the Ropsten test network and were executed five times to calculate the average results. We investigated the average cost for recording information via the container-log contract and the average costs for verifying compliance with the data protection, data minimisation, data transfer and data storage obligations. Table II shows the experimental results, indicating that when the number of operations/ actors increases, the amount of gas consumption increases sharply. In this experiment, the amount of gas used for recording user preference via the GDPR-priority contract was 83791 *wei*.

TABLE II: The transaction costs of service packages

	SP1	SP2	SP3
Number of actors	2	4	6
Number of operations	9	16	23
Total container-log (wei)	1562478	2782774	3882652
Data protection (wei)	297628	743436	1401864
Data minimisation (wei)	905648	1582621	2305178
Data transfer (wei)	323501	1112821	1803427
Data storage (wei)	304562	762341	1522370

TABLE III: Verification costs and average mining time

	SP1	SP2	SP3
Gas price (gwei)	45	57	70
Data protection ($)	3.12	9.70	23.19
Data minimisation ($)	9.49	20.65	38.13
Data transfer ($)	3.40	14.52	29.83
Data storage ($)	3.31	10.15	24.02
Mining time (seconds)	257	115	26

2) Verification cost vs mining time: This evaluation measures compliance verification for the data protection, data minimization, data transfer and data storage obligations under different gas prices. It assumes that we have three SPs, being the same as those described in Sect. V-A. The gas price units requested by user are 45, 57 and 70 *gwei* for running transactions of SP1, SP2 and SP3 respectively. Our proposed smart contracts were executed five times in Ropsten to calculate average results. As seen in Table III, the verification costs are expressed in US dollars (USD), and the average time taken for mining such transactions is calculated in seconds. Given a SP, *data protection* has the minimum verification cost, as it only considers the encryption status of operations and hence its complexity is lesser than verification of other obligations. In contrast, the most expensive verification cost is allocated to *data minimisation*, since it requires checking operations and also examines the types of personal data processed by these operations. The evaluation also shows that when the rate of gas price unit increases, there is a considerable decrease in the average mining time.

3) Evaluation of violation detection rate: This evaluation investigates the relationship between the amount of money paid by a user for verifying GDPR obligations and the average rate of successful violation detection.[3] It assumes that the aforementioned SPs are offered to the user (patient ordering medication in the pharmacy scenario). We used

[3]A verification fee shows the affordable cost that can be paid by user for the obligations' verification.

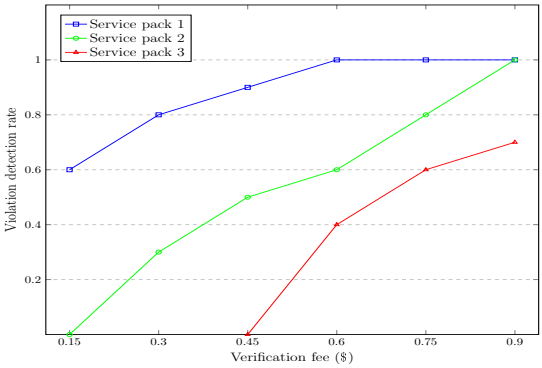


Fig. 7: Transaction fee vs Violation detection rate

Ropsten for executing the *verification* contract and the gas price is 1 *gwei*. There is a GDPR violation in a SP selected by the user for each operation. The violation is randomly selected from amongst the aforementioned obligations. The contract was executed ten times to calculate the average detection rate. Figure 7 illustrates the results of the experiment, where the x-axis shows the fee paid by the user for verifying compliance with obligations, and the y-axis shows the number of successfully detected violations. As can be seen, for a given price, SP3 involving the highest number of operations, has the lowest likelihood of violation detection. For instance, GDPR compliance cannot be detected when a user selects SP3 and their budget is limited to USD $0.3.

VI. Conclusion

This paper presented the design of a container-based monitoring and auditing architecture for improving data privacy in cloud ecosystems. The architecture makes use of a Blockchain network and supports GDPR obligations in the tracking of activities executed by cloud providers on user data. By proposing the *GDPR-priority*, *container-log* and *verification* smart contracts, the system described in this work can be used to verify compliance with three GDPR obligations by cloud providers, namely data protection, data minimisation, data transfer and data storage. The *GDPR-priority* contract allows a user to provide a preference for verifying compliance with an obligation. The *container-log* contract allows the GDPR-manager to send information relevant to the aforementioned obligations to a Blockchain. The *verification* contract enables a verifier to check compliance with obligations based on preferences determined by the user. These smart contracts were deployed in the Ropsten test network, and evaluation results show that as the number of operations in a composite service grows, the verification cost increases significantly. The implementation of our proposed smart contracts in Blockchain test network enabled us to estimate the potential cost of verification for GDPR compliance. The evaluation of transaction costs in the online pharmacy service packages, with different scales of processing activities, enabled a user to understand how changes in operations/ actors affect the cost of GDPR verification. Although a single scenario has been used in this work, our container-based architecture and proposed smart contracts can be generalised to other scenarios.

Acknowledgement

This work is supported by the Engineering and Physical Sciences Research Council (EPSRC) funded project PACE: Privacy-Aware Cloud Ecosystems (EP/R033293/1, EP/R033439/1). The work is also partially supported by the Startup Fund from Durham university, UK (090614).

References

[1] Article 29 Data Protection Working Party Opinion 5/2012 on Cloud Computing 01037/12/EN WP 196, 2012.
[2] K. Bila, O. Khali, A. Erbad, and S. U. Kha, Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers, *Computer Networks*, vol. 130, pp. 94–120, 2018.
[3] B. Russo, L. Valle, G. Bonzagni, D. Locatello, M. Pancaldi, and D. Tosi, Cloud Computing and the New EU General Data Protection Regulation, *IEEE Cloud Computing*, vol. 5, no. 6, pp. 58–68, 2018.
[4] Barati, M., Theodorakopoulos, G., Rana, O.: Automating GDPR compliance verification for cloud-hosted services. In: IEEE International Symposium on Networks, Computers and Communications, Montreal, Canada, (2020)
[5] M. Virvou and E. Mougiakou, Based on GDPR privacy in UML: case of e-learning program, *in Proc. of the 8th International Conference on Information, Intelligence, Systems & Applications*, Larnaca, Cyprus, 2017.
[6] M. Corrales, P. Jurcys and G. Kousiouris, "Smart contracts and smart disclosure: Coding a GDPR compliance framework," SSRN Electronic Journal,2018.
[7] A. Khan, "Key Characteristics of a Container Orchestration Platform to Enable a Modern Application," in IEEE Cloud Computing, vol. 4, no. 5, pp. 42-48, September/October 2017
[8] N. Kumar, G. S. Aujla, S. Garg, K. Kaur, R. Ranjan and S. K. Garg, "Renewable Energy-Based Multi-Indexed Job Classification and Container Management Scheme for Sustainability of Cloud Data Centers," in IEEE Transactions on Industrial Informatics, vol. 15, no. 5, pp. 2947–2957, 2019.
[9] X. Wang, J. He, P. Cheng and J. Chen, "Privacy Preserving Collaborative Computing: Heterogeneous Privacy Guarantee and Efficient Incentive Mechanism," in IEEE Transactions On Signal Processing, vol. 67, pp. 221–233, 2019.
[10] Regulation (EU) 2016/679 of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation, 'GDPR') OJ L119/1 2016.
[11] Y. Zhang, S. Wu, B. Jin, and J. Du, A Blockchain-based Process Provenance for Cloud Forensics, *in Proc. of the 3rd IEEE International Conference on Computer and Communications*, Chengdu, China, 2017, pp. 2470–2473.
[12] "dincloud." https://www.dincloud.com/cloud-for-pharmacy, March 2021. [Online].
[13] M. Barati and O. Rana, Tracking GDPR Compliance in Cloud-based Service Delivery, *IEEE Transactions on Services Computing*, 2020. DOI: 10.1109/TSC.2020.2999559
[14] G. Wood, Ethereum: A secure decentralised generalised transaction ledger, *Ethereum Project Yellow Paper*, 2014.
[15] "Ropsten testnet pow chain." https://github.com/ethereum/ropsten, 2020. [Online].
[16] W. Y. Tsai, T. C. Chou, J. L. Chen, Y. W. Ma, and C. J. Huang, Blockchain as a platform for secure cloud computing services, *in Proc. of the 22nd International Conference on Advanced Communication Technology*, Phoenix Park, South Korea, 2020, pp. 155–158.
[17] N. El Ioini and C. Pahl, Trustworthy orchestration of container based edge computing using permissioned blockchain, *in Proc. of the 5th International Conference on Internet of Things: Systems, Management and Security*, Valencia, Spain, 2018, pp. 147–154.
[18] T. Kittmann, J. Lambrecht, and C. Horn, A privacy-aware distributed software architecture for automation services in compliance with GDPR, *in Proc. of the 23rd International Conference on Emerging Technologies and Factory Automation*, Turin, Italy, 2018, pp. 1067–1070.
[19] A. A. Omar, M. Z. A. Bhuiyan, A. Basuc, S. Kiyomoto, and M. S. Rahman, Privacy-friendly platform for healthcare data in cloud based on Blockchain environment, *Future Generation Computer System*, vol. 95, pp. 511–521, 2019.

[20] R. Neisse, G. Steri, and I. Nai-Fovino, A Blockchain-based approach for data accountability and provenance tracking, *in Proc. of the 12th International Conference on Availability, Reliability and Security*, Reggio Calabria, Italy, 2017.

[21] A. Mahindrakar and K. P. Joshi, Automating GDPR compliance using policy integrated blockchain, *in proc. of the 6th IEEE International Conference on Big Data Security on Cloud*, Baltimore, MD, 2020.

[22] N. B. Truong, K. Sun, G. M. Lee, and Y. Guo, GDPR-compliant personal data management: A blockchain-based solution, *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1746–1761, 2020.

[23] M. Westerlund, M. Neovius, and G. Pulkkis, Providing tamper-resistant audit trails with distributed ledger based solutions for forensics of IoT systems using cloud resources, *International Journal on Advances in Security*, vol. 11, no. 3, pp. 288–300, 2018.

[24] B. Faber, G. Michelet, N. Weidmann, R. R. Mukkamala, and R. Vatrapu, BPDIMS: A Blockchain-based personal data and identity management system, *in Proc. of the 52nd Hawaii International Conference on System Sciences*, Hawaii, USA, 2019, pp. 6855–6864.

[25] F. Zemler and M. Westner, Blockchain and GDPR: Application scenarios and compliance requirements, *in Proc. of the Portland International Conference on Management of Engineering and Technology*, Portland, OR, 2019.

[26] M. Barati, O. Rana, I. Petri, and G. Theodorakopoulos, GDPR compliance verification in Internet of things, *IEEE Access*, vol. 8, pp. 119697–119709, 2020.

[27] K. Alhamazani, R. Ranjan, K. Mitra, PP. Jayaraman, Z Huang, L Wang, Clams: Cross-layer multi-cloud application monitoring-as-a-service framework, IEEE international conference on Services computing (SCC), 2014. p. 283–90.

[28] A. Noor, D. N. Jha, K. Mitra, P. P. Jayaraman, A. Souza, R. Ranjan and S. Dustdar, "A Framework for Monitoring Microservice-Oriented Cloud Applications in Heterogeneous Virtualization Environments," 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), Milan, Italy, 2019, pp. 156-163.

[29] A. Noor, K. Mitra, E. Solaiman, A. Souza, D. N. Jha, U. Demirbaga, P. P. Jayaraman, N. Cacho, and R. Ranjan. "Cyber-physical application monitoring across multiple clouds." Computers Electrical Engineering, vol 77, pp 314-324, 2019.

[30] GS. Aujla, M. Barati, O. Rana, S. Dustdar, A. Noor, J. T. Llanos, M. Carr, D. Marikyan, S. Papagiannidis, and R. Ranjan. "COMPACE: Compliance-Aware Cloud Application Engineering Using Blockchain." IEEE Internet Computing, vol 24, no. 5, pp 45-53, 2020.