


Article

Reinforcing SLA Consensus on Blockchain

Nikolaos Kapsoulis ^{1,*}, Alexandros Psychas ², Antonios Litke ²  and Theodora Varvarigou ²

¹ Innov-Acts Ltd., Nicosia 1101, Cyprus

² Electrical and Computer Engineering School, National Technical University of Athens, 157 80 Athens, Greece; alps@mail.ntua.gr (A.P.); litke@mail.ntua.gr (A.L.); dora@telecom.ntua.gr (T.V.)

* Correspondence: nkapsoulis@innov-acts.com

Abstract: Cloud Infrastructure as a Service (IaaS) Service Level Agreements (SLAs) assessment constitutes the de facto area of interest and applications in the public cloud infrastructure. However, the domination of colossal corporations tends to monopolize the way metrics and Key Performance Indicators (KPIs) are measured and determined, leading to governed environments where the clientele is unable to obtain accurate and unbiased assessment of SLAs. Leaning toward SLA self-assessment, this paper provides a fair SLA consensus approach with innate transparency and privacy by leveraging permissioned blockchains that are equipped with Trusted Execution Environments (TEEs). The SLA assessment intelligence is performed inside enclaved smart contracts isolated from the on-chain entities views. The result constitutes a permissioned blockchain ecosystem where the IaaS and their clientele commonly agree on all the respective SLA monitoring and computation rules beforehand, as defined in any SLA assessment process, while the SLA consensus scheme constantly audits the SLA metrics based on these pre-approved regulations.

Keywords: SLA assessment; public cloud; blockchain; trusted execution environment; chaincode isolation



Citation: Kapsoulis, N.; Psychas, A.; Litke, A.; Varvarigou, T. Reinforcing SLA Consensus on Blockchain. *Computers* **2021**, *10*, 159. <https://doi.org/10.3390/computers10120159>

Academic Editors: Hossain Shahriar and Corrado Aaron Visaggio

Received: 28 September 2021
Accepted: 24 November 2021
Published: 26 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Cloud IaaS SLAs have existed since the beginning of the initial launch of public Cloud infrastructure as a product. The reason of the SLA existence is the fact that a service (e.g., in the case of Cloud computing and infrastructure) cannot be offered without having an agreement that legally binds consumers and providers of the aforementioned service. SLAs are very crucial for the enforcement of trust between Cloud providers and adopters. In private Cloud solutions, the IaaS providers evaluate the SLAs themselves and inform the customers on the outcome. Furthermore, in the public sector, most Cloud providers, such as Amazon EC2, do not monitor the default SLA parameters (availability), which is a logical reality given the amount of users and the reputation of such large corporations. Nonetheless, the failure to compensate unhandled infractions might result in a loss of millions of dollars [1].

There is a plethora of tools, frameworks and generally software components that can be used to monitor, evaluate and take corrective actions as far as the performance and Quality of Service (QoS) assessment of cloud resources is concerned. In fact, many of the public Cloud providers, such as Amazon, Microsoft and Google have produced their own tools and frameworks [2–4] for the evaluation and orchestration of their and other Cloud services.

In principle, the aforementioned tools give access and produce a complete set of metrics and KPIs that aid the cloud adopters to better operate and evaluate their cloud resources with relative ease especially if the tool used is the one created by the Cloud provider. These metrics are well defined and commonly accepted as performance and QoS indicators for virtual and physical resources. CPU/RAM Utilization, network traffic and hard disk IO are a few of the most common performance indicators used and generally

cover the needs for performance and elasticity monitoring for the software that is running on the virtual resources.

Unfortunately, these metrics although being very valuable for the performance assessment of the Cloud infrastructures, cannot be used as metrics for the SLA assessment. The reason is that SLA contracts specify in a mostly verbal way (given that they are contracts) explicit guarantees as far as the service is concerned. For instance, one of the most commonly used across the most popular public IaaS providers is the Availability metric. This metric is very valuable as far as assessing the stability and robustness of a Cloud infrastructure, but also it is one of the easiest guarantees for the IaaS to provide. All the other aforementioned metrics, though being more important for the performance and QoS assessment, are very difficult to be guaranteed since they are affected by various other factors that the IaaS cannot control directly (i.e., quality of the deployed software, internet provider network capabilities and large deviations in workload).

Despite that Availability is considered a pretty straight forward metric as far as the computation is concerned, there are many variables that enroll in the standardization of the process as well as the computation of this metric. As depicted in Figure 1, the process for SLA Monitoring requires specific questions to be answered:

1. Which SLA parameters are contained in the SLA?
2. How are these parameters computed?
3. Is the computation of the parameter in line with the definition of the IaaS?

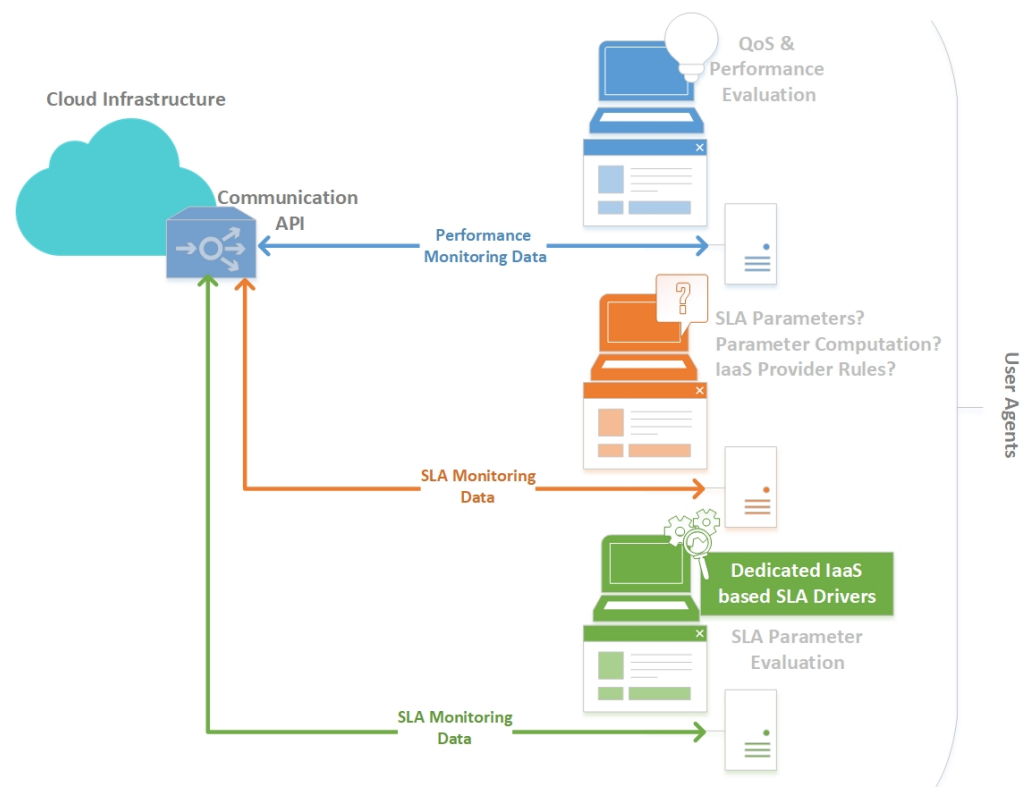


Figure 1. Standard SLA monitoring process.

In case of the availability, the parameter of the SLA is already known, but the definition of the parameters value set that considers a Cloud Service unavailable and how this value set is computed is a crucial step in order to validly monitor an SLA. In addition, important factors such as the sampling rate, the period of evaluation and the formula that calculates the parameters are fundamental in the computation of the metric.

This paper presents a novel way of resolving the aforementioned gray areas of SLA assessment validity. Envisioning the concept of SLA self-assessment, the presented approach

delivers a securely computational privacy environment that lies inside an immutable decentralized ecosystem. In particular, the solution adopts the flourishing and promising permissioned blockchains enabled with isolated execution environments in order to provide a holistic SLA consensus without ruling or biased entities nor intermediaries. Each IaaS and their clientele participate in the solution ecosystem where SLA monitoring and computation unfolds under a common understanding between the two sides.

The IaaS maintains their ability to propose the different parameters that define properties of a specific SLA, while their clientele is able to engage in a secure and confident permissioned ecosystem where their data and services consumption and utilization are characterized by accuracy and integrity. In order to maintain the transparency and privacy of the computational flows within a system that relies on blockchain applications integrity, the proposed SLA consensus design includes Trusted Execution Environment (TEE) capabilities [5]. The specific TEE adopted in this paper offers secure and isolated on-chain calculations of the SLA monitoring and computation by delivering guaranteed privacy of smart contracts data, while this SLA intelligence unfolds within the smart contracts. Both the IaaS and their clientele experience SLA assessment precision and justice through an algorithmic procedure that is pre-approved and fairly agreed by both participant sides. As the corresponding SLA consensus operations trigger the dedicated smart contract workflows, the SLA consensus life-cycle completes while transparency and privacy are realized through the TEE manifestation. In the context of digital trust, the on-chain endorsement of the TEE specification constitutes an important focal point of the implemented secure and isolated computation, and it simultaneously offers significant confidence over the entire permissioned ecosystem. The proposed solution encompasses dissimilar technical concepts and therefore creates advanced dedicated components, while all of them employ certain interoperability rules for the achievement of the automated SLA consensus with important enterprise-scale outcomes.

2. Related Work

With regards to relevant scientific research, Nguyen et al. [6] propose an architecture for evaluating and enforcing tourism SLA agreements based on the distributed ledger software. Their approach relies on keeping the SLA assessment procedure safe and unaltered through the underlying technology's immutable nature. In their work, there exists an automated SLA monitoring and computation process that takes place within the blockchain infrastructure and guarantees the successful completion of the SLA evaluation with status acknowledgment for the end-user. On the other hand, in the presented paper, the SLA consensus solution adopts the concept of SLA self-assessment and provides a technological schema where the SLA intelligence unfolds in a completely decentralized and private from third on-chain parties way.

Moreover, Ranchal and Choudhury [7] present an autonomous and trusted framework for continuous SLA monitoring in a multcloud ecosystem. Their approach is exploiting blockchain and smart contract properties in order to fairly detect SLA violations in a hierarchical system structure. As analyzed in their work, the solution is aiming to address the SLA assessment procedure in a multilevel cloud environment with diverse rules and regulations. Furthermore, Alowayed et al. [8] propose a blockchain-based network provider evaluation system according to the providers' adherence to their SLAs regarding interconnection agreements. In their work, a metric measurement mechanism verifies SLA scores for each provider for the latter's on-chain evaluation. Their approach endorses a privacy-preserving protocol for SLA agreements that aim to objectively define the SLA score of a network provider and privately store it on-chain for the interested end-user to access. Nevertheless, the presented approach in this paper relies on an SLA intelligence mechanism that is declared beforehand between the customer and the provider, includes the Algorithmic Driver for the SLA computation procedure (later in Figure 2) and executes on-chain in the manner agreed by both sides.

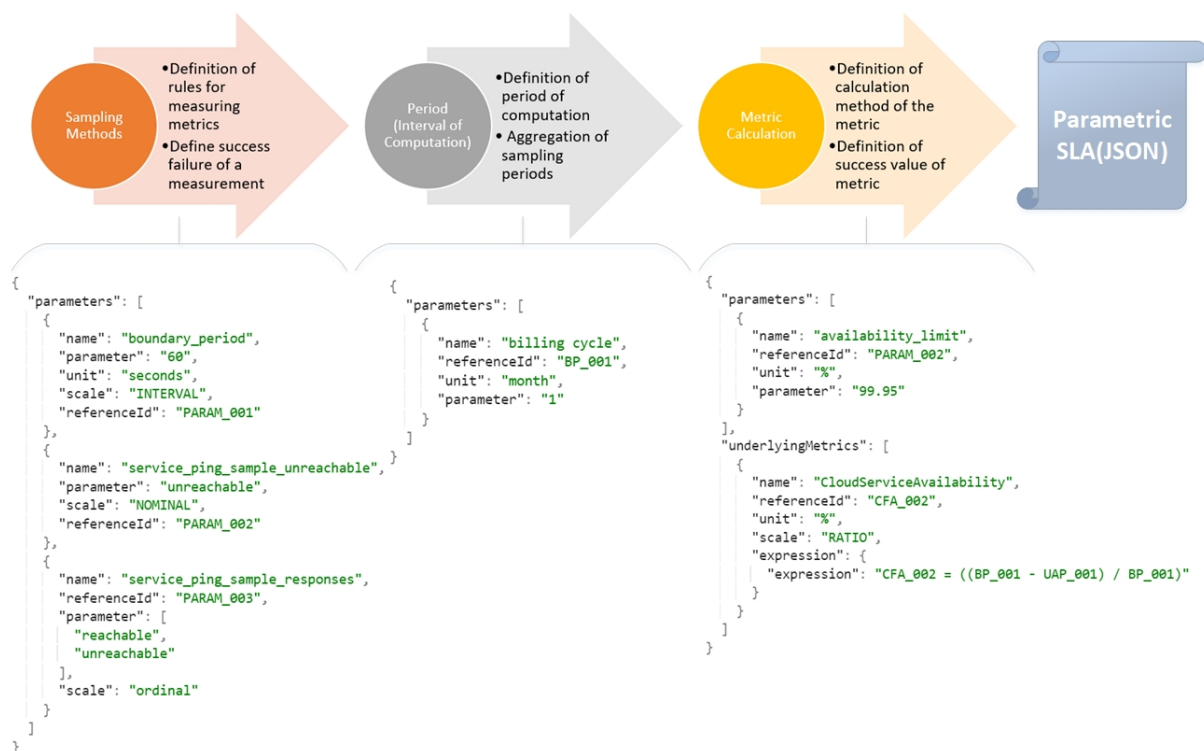


Figure 2. Layered configuration of Algorithmic Driver.

In addition, Uriarte et al. [9] propose an SLA management framework that facilitates the specification and enforcement of dynamic SLAs in order to track and define the service parameter that render the SLAs to change over time. In their work, the presented two-level blockchain-based architecture manages to convert an SLA to its smart contract equivalent that dynamically guides service provisioning within the first level; on the second level of the architecture, their solution aims to generate objective measurements for the SLA assessment through a federation of monitoring entities that scales for multiple nodes. In similar research, Alzubaidi et al. [10] present a blockchain-based approach for assessing SLA compliance and enforcing consequences, which employs a diagnostic accuracy method for dependability validation. Their approach assumes trust in service providers in order to acknowledge SLA breach incidents and consequently execute the relevant compensations. In a previous work of theirs [11], a conceptual blockchain-based framework is proposed to cope with certain limitations associated with traditional SLA management approaches. The main rationale provided in the latter work is that SLA management should be realized in a distributed environment that is not controlled by a single or a few central authorities. In this context, the proposed solution of this paper implements a system based on the latter rationale, while accumulating on-chain private smart contracts structures that protected the SLA intelligence from third blockchain network participants.

Additionally, D'Angelo et al. [12] review the challenges and requirements for enforcing accountability in Cloud infrastructures where violations of SLAs constitute an important and usual phenomenon and argue that smart contracts and blockchain technologies seem to establish a key contribution toward accountable Clouds. Finally, W. Tan et al. [13] present a novel performant and secure SLA model where trust among the IaaS, their clientele and the third-party monitoring entities are addressed through blockchain. The authors argue that there is no effective supervision mechanism for the third-party that is responsible for the monitoring and no efficient compensation mechanism on SLA violation. In their work, the proposed model effectively supervises the service providers on the blockchain with dedicated smart contract mechanisms that execute on the underlying blockchain. In the current paper, the proposed solution adopts the aforementioned concepts and presents an approach where the providers and the customers are only involved in the beginning

and in the end of the workflow with a fair and private SLA monitoring and computation procedure during the entire SLA intelligence process. A comparative assessment table (Table 1) includes the relationships of the state-of-the-art literature presented throughout this section and supplements the analysis of the key differences and additions with regards to the proposed solution.

Table 1. Comparative assessment of the studied literature with the proposed solution.

Authors	Topic	Relationship	Solution
Nguyen et al. [6]	SLA Assessment	SLA Evaluation	Private from Third-Parties
Ranchal et al. [7]	Multicloud SLA	SLA Monitoring	Different Rules in Single Blockchain
Alowayed et al. [8]	Cloud IaaS Evaluation	Privacy-Preserving Protocol	Distributed and Enclaved Operations
Uriarte et al. [9]	Dynamic SLA	SLA Provisioning	Dynamic SLA Self-Assessment
Alzubaidi et al. [10]	SLA Compliance	SLA Dependability Validation	Trustless of Cloud IaaS
Alzubaidi et al. [11]	SLA Management	On-Chain SLA Assessment	Distributed and Enclaved Operations
D'Angelo et al. [12]	Cloud Accountability	Blockchain SLA Monitoring	SLA Self-Assessment
Tan et al. [13]	Performant SLA	Cloud IaaS Supervision	Without On-chain Intermediaries

3. Blockchain SLA Consensus

As blockchains are increasingly adopted by a great deal of industrial enterprises, more trust is needed in transactional activities within the permissioned networks. In the context of SLA self-assessment, the SLA operational intelligence is performed within secure barriers that allow for computational transparency and privacy inside these environments.

In this paper, the SLA consensus's most important quality characteristics are described through the procedure of SLA Trusted Monitoring. The SLA Trusted Monitoring is offering SLA computational justice for the Cloud SLA use case. On one hand, the infrastructure provider engages their clientele in a confident system that allows computational transparency and privacy simultaneously. The former characteristic derives mainly from the common on-chain agreement on the Algorithmic Driver (later in Figure 2) used for the monitoring of the SLA logs, while the computational privacy is established through the TEE [5] properties and capabilities. On the other hand, the customer is benefited by a secure platform where trustworthiness of the SLA consensus is guaranteed. The customer is confident that the provided SLA computations derive from a specific calculation scheme as agreed with the infrastructure provider at the initial phase of their interaction. The entire procedure of the SLA Trusted Monitoring evolves within a permissioned blockchain network that leverages the characteristics of Distributed Ledger Technologies (DLTs) equipped with on-chain TEEs. In particular, the solution is based on the Hyperledger Fabric-distributed ledger software [14] that hosts an on-chain TEE by exploiting the corresponding dedicated enabler, namely the Hyperledger Fabric Private Chaincode (FPC) [15]. The latter provides an on-chain framework extension to develop and deploy smart contracts that are executed in protected isolated environments.

Figure 3 holistically depicts the ecosystem of the solution that describes the SLA assessment consensus under the umbrella of transparency and privacy of operations. The IaaS and their clientele leverage by the ecosystem's standardized workflow that primarily includes the on-chain scheme accompanied with the explicit off-chain interactions. A quick overview of the process and the components interactions follows while the analysis unfolds later in the paper.

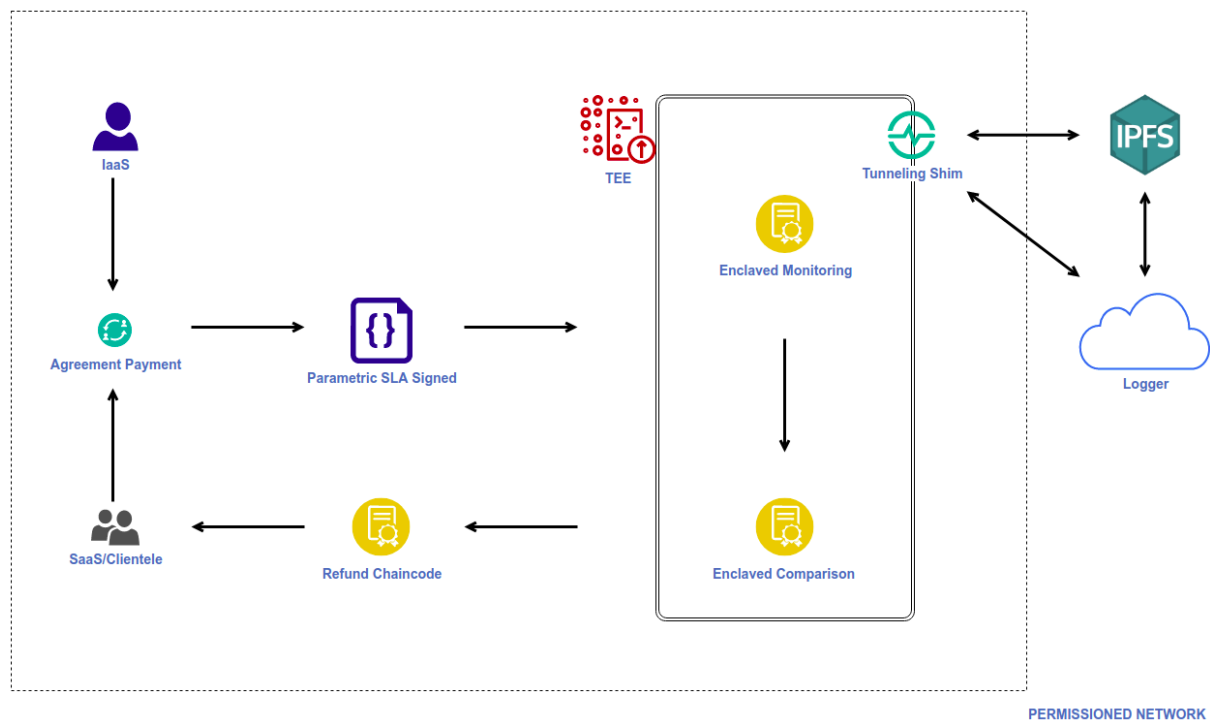


Figure 3. Blockchain SLA consensus architecture.

In principle, the architectural platform is built on a permissioned blockchain network where each of its main two actors is a member of the blockchain. When an SLA product offered by the IaaS is purchased by the SaaS (or other type of IaaS clientele), the Agreement Payment occurs as a transaction between the involved parties. The Agreement Payment constitutes a blockchain transaction that is verified by the signatures of the interested parties and includes the corresponding product acquisition. During this procedure, the agreed Parametric SLA template digital document is signed by both and announced to the on-chain TEE as “Parametric SLA Signed”. The Parametric SLA Signed encloses the necessary SLA data as defined by the Agreement Payment and the validation of the IaaS and SaaS involvement in the form of the respective digital signatures. Sequentially, as soon as the Parametric SLA Signed is prepared, the TEE includes it as a new agreement on its registry of SLA assessments, where it constantly provides the dedicated enclaved operations services. The TEE is backed by the Enclaved Monitoring which derives the SLA metrics as offered from the Tunneling Shim. The latter constitutes a Cloud and InterPlanetary File System (IPFS) integration enabler that manages the dedicated integration points of the blockchain network with the outside world in a secure way. Additionally, the TEE endorses the Enclaved Comparison component which computes the SLA violations on behalf of the TEE and triggers the Refund Chaincode when the corresponding conditions are met. Eventually, the latter is activated in case of violations, while it satisfies and compensates the respective economic or other relations between the IaaS and their clientele as agreed initially in the signed Parametric SLA.

3.1. SLA Standardised Monitoring

In order for the Parametric SLA to work, it must contain all the needed information from the SLA document created by the IaaS provider, in a standardised data schema format. As far as the schema is concerned, Parametric SLA follows the ISO 19086-2 SLA [16] standard, which gives the appropriate guidelines in order to create the fundamental classes of the schema. This process transforms the SLA document into a JSON document, which is then be used as an input to the Algorithmic Drivers for SLA Evaluation. Creating the parametric SLA requires the specification of a number of important information that construct the JSON Shema that represents the SLA. These parameters are the following:

1. **Metrics:** This class represents all the objectives that regard the service for the SLA guarantees. For instance, Availability (as mentioned in Section 1) is one of the most common metrics used for an SLA. In addition, this class contains basic information about the metrics, mainly to make the monitoring and measurement of a given metric feasible.
2. **Parameters:** This class complements each individual metric and contains in detail the parameters that constitute a specific metric. In addition, it contains the type of variables that express a metric and how they can be measured.
3. **Rules:** SLA contracts contain not only the specific metrics that they guarantee but also the rules with which the metrics are measured. To be more exact, the rules constrain the way that the metrics are interpreted, the most common case of a rule is what constitutes for a specific metric failure or success. For instance, for the metric of availability, Amazon AWS SLA [17] consider a service unavailable when it is not accessible in two availability zones (in order to avoid a single point of failure).

Parameters as well as the rules that dictate them not only shape the Parametric SLA schema but also the actual computation of the SLA metrics and, by extension, the algorithmic drivers (Figure 2). In order for algorithmic drivers to evaluate an SLA, they must consider all the information provided by the Parametric SLA. The calculation of the metrics can be affected by the rules so much that different algorithmic drivers must be produced for each individual IaaS provider even if they calculate the same metrics. In the same manner as the Parametric SLA, algorithmic drivers take a standardised approach for their development, by following the SLALOM cloud specification model [18].

The Sampling Methods layer is responsible for creating the methods which are responsible for collecting and evaluating the validity of sample data collected in order to compute a specific metrics. Basically, the constraints that are derived from the rules are expressed in Boolean form, in order to dictate if a specific sample is stored to be used in the computation of the metric or if it is discarded because it is not valid based on the constraints of measurement.

Interval of Computation layer dictate's the sampling rate of the data for computing the metrics and also the interval in which the SLA Metrics are computed and evaluated. One piece of the information an SLA contracts contain is the billing cycle and, by extension, the SLA evaluation cycle. Some services have their SLA parameters calculated once every month and every other year.

The Metric Calculation layer is responsible for the actual computation of the metrics for any given interval of time. Specified functions, which take into consideration the definition as well as the rules that the metrics have, process the sampling data and produce the numbers for the SLA metrics.

After the successful configuration, the Parametric SLA becomes the on-chain proof of the SLA agreement including all the appropriate data, such as the involved sides wallet addresses, SLA metrics details and the Algorithmic Drivers that apply during the SLA enclaved operations. Along with the submission of the Parametric SLA Signed on the ledger, the TEE obtains this new agreement entry and includes it to the dedicated enclaves portfolio where it is tracked and benefited by the SLA Trusted Monitoring.

3.2. SLA Trusted Monitoring

Once the *Parametric SLA Signed* is announced to the on-chain TEE, then the SLA Trusted Monitoring procedure initiates for this new agreement. Since the entire scenario unfolds on-chain, the related and processed data for the SLA agreement is trusted and transparent to the IaaS and their clientele. On top of the aforementioned design properties, the SLA monitoring and computation are kept private from every other blockchain network entity apart from the enclaved environment inside where they occur.

As soon as the Enclaved Monitoring component includes the new agreement in the dedicated enclaves portfolio, the latest SLA logs are retrieved and processed within the TEE. The Enclaved Monitoring exists and executes automatically inside the implemented

FPC as a well-defined smart contract structure [19], while the enveloping FPC guarantees the privacy of the operations as performed by the hosted components, i.e., the smart contract structures of Enclaved Monitoring and Enclaved Comparison. The blockchain activity within the FPC is kept separate and protected from every other blockchain entity that participates in the network and reads the common ledger. The Enclaved Monitoring primarily comprises its own dedicated smart contract functions that acknowledge the newly signed agreement and retrieve the latest agreement logs from the IPFS network with the aid of the Tunneling Shim. The complete workflow is performed within the enclaved chaincode leveraging the aforementioned privacy features.

During the life cycle of the solution workflow, the Logger Cloud component constantly broadcasts new log files of an agreement to the IPFS network. Ultimately, as IPFS constitutes a content-addressed versioned P2P file system [20], the returned Content Identifier (CID) is naturally used by the Tunneling Shim for the fetching of the logs of an agreement. The Tunneling Shim constitutes a specifically developed integration enabler that benefits the Enclaved Monitoring interaction with the IPFS nodes. Through the Tunneling Shim, the retrieval of the latest SLA logs of a specified agreement is achieved, while they are eventually delivered to the Enclaved Monitoring component. Finally, the Enclaved Monitoring circulates the SLA logs to the Enclaved Comparison component.

The Enclaved Comparison also constitutes a specifically defined Smart Contract structure that exists inside the implemented FPC and executes automatically. It is triggered uniquely by the Enclaved Monitoring announcement of the logs. The Enclaved Comparison smart contract structure performs the SLA computation by the calculation of the possible SLA violations. It exploits all the related information data of an agreement, including the SLA agreement metrics, the real metrics (log files) and the agreement algorithmic driver, and determines the status of an SLA violation. As anticipated, this computation is realized inside the private and isolated environment of the FPC leveraging the TEE corresponding features and is completely secured from third entities that participate in the blockchain network. The concluded outcome of the Enclaved Comparison smart contract structure calculations may or may not be in favor of an SLA violation. In the context of the described privacy-preserving FPC environment as the foundation of the solution, the concluded outcome is an unbiased result drawn under the pre-agreed rules of the initial well-defined and acknowledged-by-both-sides SLA agreement.

In the case of an SLA violation, the Enclaved Comparison provokes the execution of the Refund Chaincode. The Refund Chaincode actively receives all the necessary SLA agreement information data in order to complete its part of the workflow. The role of the Refund Chaincode is to satisfy the SLA violation terms as described in the agreement. It constitutes a chaincode realizing a set of functions which primarily execute the violation compensations and possibly other agreed actions, such as modifying the parties' reputation, the offered product score, and others. On the SLA violation, the customers, i.e., the IaaS clientele, receive a refund payment as a compensation of the violation, while the IaaS provider is charged for the violation. The Refund Chaincode is the last component that completes the solution life cycle during the SLA violation workflow.

4. Experimentation and Results

In this paper, the aforementioned concepts and workflows of the presented SLA consensus are implemented and tested in the corresponding setup as depicted in the architecture (Figure 3).

In terms of technological adoption, the underlying blockchain infrastructure relies on a Hyperledger Fabric permissioned network that hosts a TEE and interoperates with it through the dedicated enabler, namely the Hyperledger FPC v1.0.0-rc1. Particularly, the underlying Hyperledger Fabric network adopts a crash fault-tolerant consensus algorithm that enables the network to scale to thousands of transactions per second [21]. Inside this blockchain network, the Hyperledger FPC hosts the dedicated business intelligence in the enclaved smart contracts structures. The corresponding logic that is executed inside these

structures is kept private from the blockchain entities that participate in the network and initiate transactional and contractual activities.

The adopted TEE receives every new agreement in the format of the ISO/IEC 19086-2:2018 specification for a t2.nano Amazon testing instance [22] and processes it through the respective specification-compliant functions of the smart contract structures (see Section 3.2). Additionally, the TEE links and integrates with the off-chain world through the Tunneling Shim component which is a single-purpose dedicated connector that allows the corresponding interoperability with the IPFS network (v0.9.0) and the public Cloud, i.e., the Logger component. Inside the TEE, the private smart contract structures are executed, isolated on the endorsed FPC in the form of a consolidated chaincode package which performs node-level isolation and provides complete privacy against the simply submitted transactions, namely those that are external to the TEE. When an SLA Violation occurs the dedicated smart contract, namely the Refund Chaincode, initiates its execution through the on-chain trigger broadcast that receives from the computation enclaved chaincode. All smart contract structures are written in Golang 1.14.12, while they adopt the standard Hyperledger Fabric Go API [23] for the invocation of the transaction operations and chaincode triggers.

In the context of the SLA violation structural workflow, as soon as the SLA product transaction is completed, the described TEE operations are prompted sequentially, i.e., the Enclaved Monitoring followed by the Enclaved Comparison. Consistently, the TEE triggering initiates from outside the enclaved environment, while it is backed by strong encryption mechanisms for these bilateral interactions [24,25]. Inside the TEE, the SLA log data are safely retrieved through the aid of the Tunneling Shim that interconnects solidly with the IPFS network. This interoperability occurs inside the TEE and is completely protected and isolated from third entities, namely unauthorized chaincodes, blockchain network or consensus nodes. The retrieved SLA logs are processed inside the TEE, i.e., on the corresponding chaincode memory, and the performed SLA intelligence output exits the private isolated environment and simultaneously completes the SLA Trusted Monitoring life cycle. Afterward, if an SLA violation occurs, the Refund Chaincode triggers and executes publicly across the entire permissioned network.

Figure 4 depicts the experimental results of the tested solution by illustrating the time necessary for the system to process and verify whether there exists an SLA violation or not. The proposed approach examines the time performance shift between the circumstances of appearance or not of a violation during the system workflow. The experimental results assert that, inside the proposed framework, the dedicated solution of the SLA consensus detects an SLA violation within an optimal time frame when deployed under the umbrella of isolated and private from on-chain parties SLA monitoring and computation.

In general, the period required for the resolution and submission of the private chaincode transactions inside the enclaved environment varies for both the existence or nonexistence of a violation. However, as depicted in the experimental results, in both types of events, the time performance of the solution ranges within a certain time range that appears to be wider when a violation is present. In general, the absence of a violation displays a regularly more rapid solution output than the case of a violation conclusion. It is true that during the violation case, extra private chaincode calculations are required in order to complete the respective workflow; thus, in a regular execution, the violation sequence should require a longer time period.

In the proposed solution, specified and unaltered smart contract structures are executed for the completion of the workflow of the violation determination; thus, their performance time remains within the measured thresholds for those specific smart contract structures. This is due to the high transaction throughput and the transactional mechanics of the underlying permissioned blockchain that include the rapid transactions per second variant and the sequential transaction flow phases [19]. Thus, the SLA violation transactions can be completed in less time when endorsed inside an executing transaction pipeline. In addition, the specified time ranges of the two cases exist between certain thresholds

and cannot be altered in any significant way toward a positive or negative direction. The enclaved chaincodes utilized in the solution are being submitted on the ledger and invoked from the triggers that initiate from within the blockchain network; therefore, their code structure employs the immutability properties of the chain and important time deviations of the solution for both described cases turn inapplicable due to the smart contract code immutability validating the solution coherence. Ultimately, as far as scalability is concerned, the holistic workflow of the system is able to be scaled in both SLA violation cases by following the corresponding properties of the Hyperledger Fabric network [26]. In this context, the adopted blockchain platform scales in terms of network entities and transactional workflow, delivering a solid framework that supports thousands of transactions per second [27]. In the proposed solution of this paper, the contractual executions of the private smart contract structures follow the scaling properties of the underlying blockchain by exploiting the corresponding performance features.

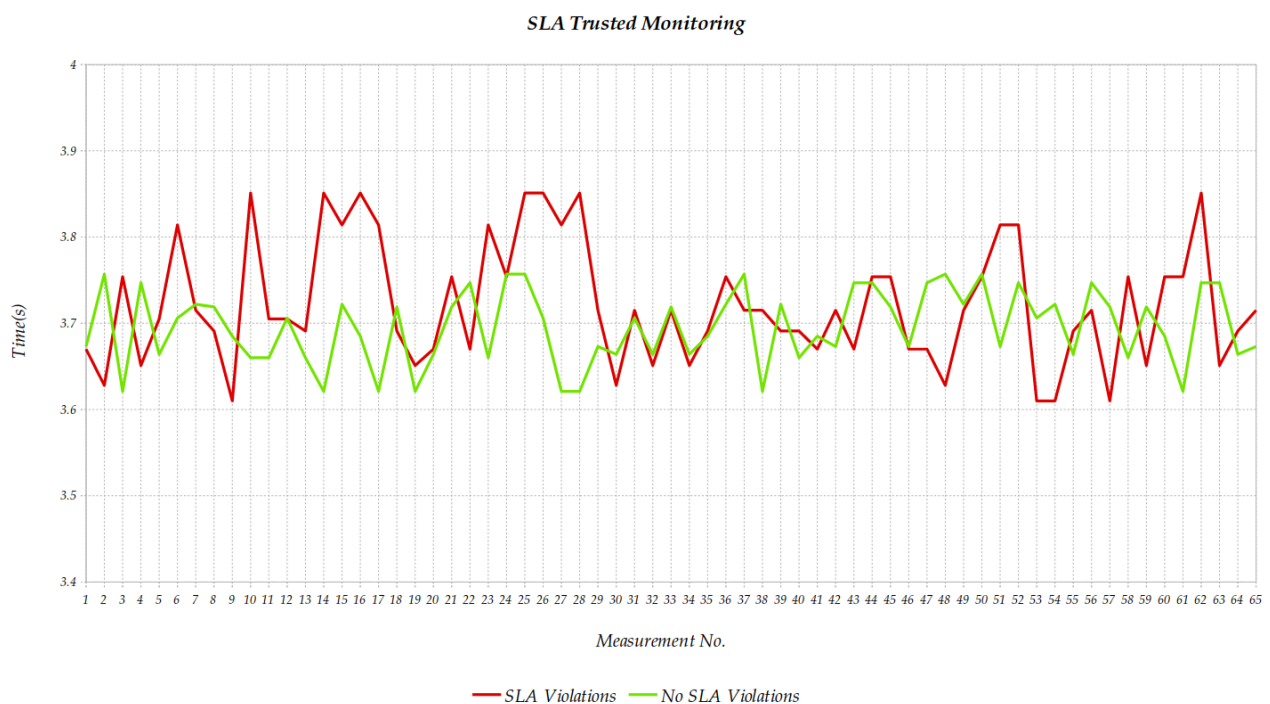


Figure 4. SLA violations time performance shift in SLA Trusted Monitoring.

5. Conclusions and Future Work

The presented SLA assessment system materializes a novel approach of SLA consensus in the public cloud infrastructure realm. The solution concept is defined around the distributed ledger software that hosts TEEs with isolated computation capabilities. The privacy-preserving properties of such a setup benefit the entire SLA assessment life cycle as every SLA agreement is clearly assessed from both the IaaS provider and their customers. All the activity occurs within a permissioned blockchain, while the dedicated SLA intelligence is isolated and executed within the employed TEE. The result is a confident system that both the IaaS provider and their clientele benefit from in terms of precision and computational justice. For the presented experimental outcomes, the proposed solution is able to scale for enterprise use cases with profitable and efficiently applied interest, as per the underlying blockchain scalability features. In terms of future work on the system setup, the current directions focus mainly on supporting more violation-agreedactions, such as adding parties reputation management and SLA product scoring in the workflow of the SLA violations.

Author Contributions: Conceptualization, N.K., A.P., A.L. and T.V.; methodology, N.K., A.P., A.L. and T.V.; software, N.K. and A.P.; validation, N.K., A.P., A.L. and T.V.; formal analysis, N.K., A.P., A.L. and T.V.; investigation, N.K., A.P., A.L. and T.V.; resources, N.K., A.P., A.L. and T.V.; data curation, N.K. and A.P.; writing—original draft preparation, N.K., A.P., A.L. and T.V.; writing—review and editing, N.K., A.P., A.L. and T.V.; visualization, N.K. and A.P.; supervision, N.K., A.P., A.L. and T.V.; project administration, N.K., A.P., A.L. and T.V.; funding acquisition, A.L. and T.V. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results has received funding from the European Commission under the H2020 Programme’s project Pledger (grant agreement No. 871536).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SLA	Service Level Agreement
IaaS	Infrastructure as a Service
TEE	Trusted Execution Environment
DLT	Distributed Ledger Technology
FPC	Fabric Private Chaincode

References

1. Columbus, L. Roundup of Cloud Computing Forecasts And Market Estimates, 2016. Forbes. [Online]. Available online: <https://www.forbes.com/sites/louiscolombus/2016/03/13/roundup-of-cloud-computing-forecasts-and-market-estimates-2016/> (accessed on 11 November 2021).
2. Amazon Cloudwatch. Available online: <https://aws.amazon.com/cloudwatch/> (accessed on 11 November 2021).
3. Microsoft Monitor. Available online: <https://azure.microsoft.com/en-us/services/monitor/> (accessed on 11 November 2021).
4. Google Monitoring. Available online: <https://cloud.google.com/monitoring/> (accessed on 11 November 2021).
5. Sabt, M.; Achemlal, M.; Bouabdallah, A. Trusted Execution Environment: What It is, and What It is Not. In Proceedings of the 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, Finland, 20–22 August 2015; pp. 57–64.
6. Nguyen, T.-V.; Lê, T.-V.; Dao, B.; Nguyen-An, K. Leveraging Blockchain in Monitoring SLA-Oriented Tourism Service Provisioning. In Proceedings of the International Conference on Advanced Computing and Applications (ACOMP), Nha Trang, Vietnam, 26–28 November 2019; pp. 42–50.
7. Ranchal, R.; Choudhury, O. SLAM: A Framework for SLA Management in Multicloud ecosystem using Blockchain. In Proceedings of the IEEE Cloud Summit, Harrisburg, PA, USA, 21–22 October 2020; pp. 33–38.
8. Alowayed, Y.; Canini, M.; Marcos, P.; Chiesa, M.; Barcellos, M. Picking a Partner: A Fair Blockchain Based Scoring Protocol for Autonomous Systems. *Proc. Appl. Netw. Res. Workshop* **2018**, 33–39. [CrossRef]
9. Uriarte, R.B.; Zhou, H.; Kritikos, K.; Shi, Z.; Zhao, Z.; De Nicola, R. Distributed service-level agreement management with smart contracts and blockchain. *Concurr. Comput. Pract. Exper* **2021**, *33*, e5800. [CrossRef]
10. Alzubaidi, A.; Mitra, K.; Patel, P.; Solaiman, E. A Blockchain-based Approach for Assessing Compliance with SLA-guaranteed IoT Services. In Proceedings of the IEEE International Conference on Smart Internet of Things (SmartIoT), Beijing, China, 14–16 August 2020; pp. 213–220.
11. Alzubaidi, A.; Solaiman, E.; Patel, P.; Mitra, K. Blockchain-Based SLA Management in the Context of IoT. *IT Prof.* **2019**, *21*, 33–40. [CrossRef]
12. D’Angelo, G.; Ferretti, S.; Marzolla, M. A Blockchain-based Flight Data Recorder for Cloud Accountability. In Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems, Munich, Germany, 15 June 2018; pp. 93–98.
13. Tan, W.; Zhu, H.; Tan, J.; Zhao, Y.; Xu, L.D.; Guo, K. A novel service level agreement model using blockchain and smart contract for cloud manufacturing in industry 4.0. *Enterp. Inf. Syst.* **2021**, 1–26. [CrossRef]
14. Hyperledger Fabric. Available online: <https://www.hyperledger.org/use/fabric> (accessed on 11 November 2021).
15. Fabric Private Chaincode. Available online: <https://github.com/hyperledger-labs/fabric-private-chaincode> (accessed on 11 November 2021).
16. ISO/IEC 19086-2:2018 Cloud Computing—Service Level Agreement (SLA) Framework—Part 2: Metric Model. Available online: <https://www.iso.org/standard/67546.html> (accessed on 11 November 2021).

17. Amazon Compute Service Level Agreement. Available online: <https://aws.amazon.com/compute/sla/> (accessed on 11 November 2021).
18. SLALOM SLA Specification and Reference Model. Available online: <http://slalom-project.eu/content/final-version-slalom-sla-technical-specification-and-reference-model> (accessed on 11 November 2021).
19. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the thirteenth EuroSys conference (EuroSys'18), Porto, Portugal, 23–26 April 2018; pp. 1–15. [CrossRef]
20. Benet, J. IPFS—Content Addressed, Versioned, P2P File System. 2014. Available online: <https://ipfs.io/> (accessed on 11 November 2021).
21. Diego, O.; John, O. In search of an understandable consensus algorithm. In Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference (USENIX ATC'14), Philadelphia, PA, USA, 19–20 June 2014; pp. 305–320.
22. Amazon AWS General Purpose Instance t2.nano. Available online: <https://aws.amazon.com/ec2/instance-types/> (accessed on 11 November 2021).
23. Go Chaincode Contract API. Available online: <https://github.com/hyperledger/fabric-contract-api-go/tree/main/contractapi> (accessed on 11 November 2021).
24. Fujisaki, E.; Okamoto, T.; Pointcheval, D.; Stern, J. RSA-OAEP Is Secure under the RSA Assumption. *Adv. Cryptol.* **2018**, *2139*, 260–274.
25. Standards for Efficient Cryptography. Available online: <http://www.secg.org/sec2-v2.pdf> (accessed on 11 November 2021).
26. Nguyen, M.; Loghin, D.; Dinh, T.T. Understanding the Scalability of Hyperledger Fabric. *arXiv* **2021**, arXiv:2107.09886.
27. Dreyer, J.; Fischer, M.; Tönjes, R. Performance analysis of hyperledger fabric 2.0 blockchain platform. In *Proceedings of the Workshop on Cloud Continuum Services for Smart IoT Systems (CCIOT'20)*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 32–38. [CrossRef]