Trust Management and Federated Resource Utilization in the Cloud Edge Computing Continuum

Mariem ALJENE*, Sofiane MESSAOUDI*, Adlen KSENTINI*
*Eurecom Institute
*Sophia Antipolis, France
Email: *name.surname@eurecom.fr

***Abstract*—Cloud-Edge Computing Continuum (CECC) systems drive digital transformation by linking cloud services to decentralized edge devices, supporting critical applications such as eHealth. However, trust management in such dynamic and distributed federated systems remains a challenge. This paper proposes the Trust Manager, an advanced framework that integrates the IEEE Federation Hosting Service (FHS) model with blockchain technology. It enhances trust through comprehensive trust profiles for infrastructure providers, incorporating performance metrics, Service Level Agreement (SLA) compliance, and user feedback, thus improving scalability, performance, and reliability in federated systems.**

I. Introduction

In the era of digital transformation, Cloud Edge Computing Continuum (CECC) systems are pivotal to modern applications, bridging centralized cloud services and decentralized capabilities of edge devices. These systems support mission-critical applications such as healthcare, and autonomous vehicles, where reliable and secure communication is indispensable. However, trust management in *distributed* (i.e., multiple resource providers) CECC systems presents a significant challenge due to their dynamic and heterogeneous nature.

Federated systems, including CECC, enable collaboration among diverse stakeholders such as infrastructure providers, application developers, and Cloud Edge Computing Continuum Manager (CECCM) (i.e., service and resource orchestrators). The federation of resource providers allows for seamless, scalable, and efficient resource sharing across different administrative domains, ensuring optimal performance and reliability as defined in Service Level Agreements (SLAs). Trust management plays a crucial role in maintaining consistent Quality of Service (QoS), Quality of Experience (QoE), and SLA adherence while ensuring secure and fair resource allocation. In this context, trust refers to the confidence in stakeholders' ability to reliably and transparently fulfill their obligations.

Managing trust in federated CECC environments is complex due to the dynamic roles and objectives of participating entities. Existing approaches primarily focus on resource optimization [1] or cost efficiency [2] but often neglect dynamic trust evaluation and enforcement. Traditional models struggle with frequent fluctuations in resource availability, leading to uncertainties in infrastructure reliability and consistent SLA fulfillment [3], [4]. Xu and Liu [5] emphasized trust in decentralized systems through reliability/availability models but overlooked blockchain's potential for distributed trust. Similarly, Zhao et al. [6] highlighted reputation-based trust mechanisms, though their reliance on centralized control makes them unsuitable for CECC environments.

To address these challenges, we propose a trust management framework that integrates the Institute of Electrical and Electronics Engineers (IEEE) Standard for Intercloud Interoperability and Federation (SIIF) [7] Federation Hosting Service (FHS) reference model with blockchain technology. The FHS model, which extends the National Institute of Standards and Technology (NIST) Cloud Federation Reference Architecture (CFRA) [8], provides a standardized framework for interoperability and collaboration. Blockchain further enhances transparency and accountability by ensuring a decentralized and tamper-proof trust mechanism [9]. Our approach establishes detailed trust profiles for infrastructure providers, incorporating historical SLA compliance, performance metrics, and user feedback, enabling *informed decision-making* in resource allocation and service deployment.

A key innovation of our approach is the integration of a *Trust Manager* external to the CECC ecosystem, leveraging IEEE FHS standards to interface with application developers, infrastructure providers, and the CECCM via standardized Application Programming Interfaces (APIs). It dynamically evaluates trust profiles using an advanced model that considers infrastructure providers' Key Performance Indicator (KPI) monitoring, SLA compliance, and application developer feedback (i.e., QoE KPIs). Blockchain ensures secure and transparent interactions while enforcing trust across edge devices, cloud services, and users. Trust scores for all infrastructure and resource providers are *continuously* computed and stored in the *Resource Broker*. The CECCM utilizes these scores to make informed decisions on microservice deployment and migration, ensuring SLA adherence while optimizing both QoS and QoE for end users.

The remainder of this paper is organized as follows: Section II presents our proposed solution, including the architecture, mathematical model, and workflow. Section III evaluates the performance of the proposed model, showcasing the results. Finally, Section IV concludes

II. Trust Management in CECC

This section is dedicated to the trust overview, model, and workflow.

A. Trust Architecture

To establish and sustain the reputation of infrastructure providers, we propose a novel trust architecture, illustrated in Figure 1, seamlessly integrated into the FHS framework defined by the IEEE SIIF. This architecture introduces the Trust Manager, an independent component hosted by the Federation Manager, responsible for deriving and managing provider reputations while ensuring interoperability and scalability within the FHS model.

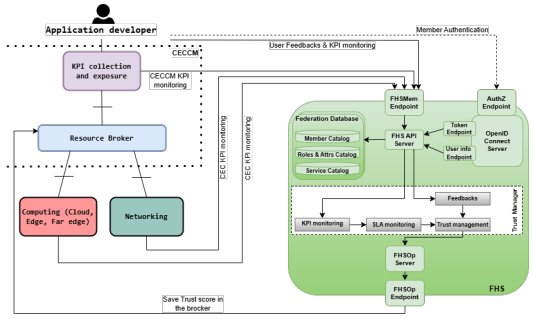


Figure 1: Trust architecture leveraging the IEEE SIIF FHS model

The proposed architecture integrates the following modules within the Trust Manager:

1) **KPI Monitoring Module**: Collects and aggregates monitoring data on KPIs from key stakeholders, including the CECCM, infrastructure providers, and application developers.
2) **SLA Management Module**: Leverages KPI data to detect SLA violations and their sources using Smart Contracts that formalize SLAs and dynamically monitor compliance. Automated SLA enforcement provides real-time violation insights, reinforcing trust.
3) **Feedbacks Module**: Gathers user input (e.g., service load time) periodically or upon application termination to evaluate service quality delivered by infrastructure providers.
4) **Trust Management Module**: Integrates outputs from the SLA Management and Feedback Modules to calculate and update infrastructure providers reputation.

Reputation scores are securely recorded on a blockchain, ensuring transparency and tamper-proof management. These scores are available to the Resource Broker, a component of the CECCM, where they are utilized to make informed decisions on federated resource and application management (including microservice deployment and migration). This ensures SLA adherence while optimizing both QoS and QoE for end users, enabling resource provider selection based on trustworthiness, resource availability, and cost.

To ensure interoperability and scalability among stakeholders (i.e., application developers, CECCM, and infrastructure providers) and the Trust Manager, the latter is integrated into the FHS framework of the IEEE SIIF through the following functionalities:

1) **Authentication and Security**: Member authentication uses the OpenID protocol, with Authorization (AuthZ) Endpoints securing access to FHS and the Trust Manager. Roles and access control are managed via the Roles & Attributes Catalog.
2) **Federation Database**: A central repository (Member/Service Catalog) that stores federation members and services, supporting authentication, authorization, and service details.
3) **Open API-Based User Feedbacks and KPI Monitoring**: The Trust Manager consolidates data from the application developers, CECCM, and infrastructure providers via the FHS API Server. This data is used to detect SLA violations and refines providers trustworthiness.
4) **Open API-Based Trust Score Provision**: The Trust Manager updates trust scores and informs Resource Broker decisions through the FHS Operator (FHSOp) Server API. This ensures trust scores influence resource selection by balancing performance, cost, and reliability.

Operating within the FHS model, the Trust Manager facilitates seamless federation-wide communication and robust trust management, fostering a trustworthy CECC ecosystem. Adherence to the IEEE SIIF standard ensures compatibility and scalability. The FHS-based Open API interfaces enable smooth communication, allowing the Trust Manager to expose reputation scores and retrieve data for trust computation.

B. Trust Computation Mathematical Model

In this section, we present a mathematical model for calculating trust scores in CECC, focusing on the relationship between clients (application developers) T_c and Cloud Edge Continuum (CEC) infrastructure providers T_i. For simplicity, the CECCM component is excluded from the equations, as it follows the same representation as infrastructure KPI monitoring. In practical deployments, CECCM can be incorporated to enhance trust score reliability and replication.

The model evaluates the performance of various KPIs over time. Inspired by the state of the art [10], a simple approach for trust computation T compares actual performance metrics to target values defined in the SLA. The outcome is adjusted based on whether higher or lower values are preferable for trust, as shown in Equation 1. SLAs specify expected service levels, while KPIs measure actual performance against these expectations.

$$\mathrm{T} = \frac{1}{n}\sum_{i=1}^{n} \min\left(1, \left(\frac{\text{actual}}{\text{target}}\right)^{d}\right)$$
$$\text{where} \begin{cases} n: \text{ is the number of KPIs,} \\ d = 1 \text{ in maximization scenarios (e.g., throughput),} \\ d = -1 \text{ in minimization scenarios (e.g., latency).} \end{cases} \quad (1)$$

This method is simple to implement and produces a trust score between 0 and 1. However, it does not consider the duration or magnitude of performance deviations, which could potentially result in misleading trust evaluations.

To address these limitations and ensure robust trust assessment, we introduce an enhanced mathematical model. This model incorporates the final trust calculation, QoE (client feedback) and SLA metrics aggregation, KPI normalization, and trust score adjustments. The final trust score, $T(t)$, is computed by averaging the current trust score with the previous score at $t-1$. This approach integrates past and present performance, stabilizing the trust score. Current performance is represented by the sum of the Capped Adjustment (CA) and Aggregated Score (A), each capped at 1 to maintain the trust score's upper limit. By averaging, the model ensures short-term fluctuations—whether positive or negative—do not disproportionately affect the overall trust score, promoting stability over time (see Equation 2).

$$\begin{aligned} &\mathrm{T}(t) = \frac{\min(1, \mathrm{A} + \mathrm{CA}) + \mathrm{T}(t-1)}{2} \\ &\text{where} \begin{cases} CA: \text{ accounts for deviations in performance,} \\ A: \text{ is Aggregated client/infrastructure trust scores.} \end{cases} \end{aligned} \tag{2}$$

The equation includes complex factors, namely A and CA, which are broken down into simpler components:

Aggregated Client/Infrastructure Trust Score (A): It combines the infrastructure-side and client-side trust scores using weighted contributions. Weights w_c and w_i, as shown in Equation 3, represent the importance of client and infrastructure scores, respectively, in the overall trust computation. Adjusting these weights allows the system to emphasize either side based on specific requirements. This aggregate balances infrastructure metrics, such as server uptime, with client experience metrics, like response time, providing a comprehensive view of system performance.

$$\begin{aligned} &\mathrm{A} = w_{\mathrm{c}} \cdot \mathrm{T_c} + w_{\mathrm{i}} \cdot \mathrm{T_i} \\ &\text{where} \begin{cases} w_{\mathrm{c}} \text{ is the weight for the client trust score,} \\ w_{\mathrm{i}} \text{ is the weight for the infrastructure trust score,} \\ \mathrm{T_c} \text{ is the trust score for the client,} \\ \mathrm{T_i} \text{ is the trust score for the infrastructure.} \end{cases} \end{aligned} \tag{3}$$

The trust scores T_c and T_i can be further refined and normalized:

Normalized Trust: To enable fair comparisons between KPIs, normalized trust computation scales metrics to a common range. Equation 4 evaluates system performance relative to expectations by dividing actual performance by target performance. A ratio greater than 1 indicates performance exceeding expectations, while a ratio less than 1 indicates underperformance. An exponent d adjusts the ratio to account for whether higher values (e.g., throughput) or lower values (e.g., latency) are preferred, ensuring proper interpretation of all metrics. Normalization is crucial for comparing metrics with different units and ranges. To avoid over-rewarding systems for exceeding goals, normalized values are capped at 1. Averaging across all KPIs ensures the trust score reflects multidimensional performance rather than relying on a single metric.

$$\mathrm{T_{c/i}} = \frac{1}{n} \sum_{i=1}^{n} \min\left(1, \left(\frac{\mathrm{actual_{c/i}}}{\mathrm{target_{c/i}}}\right)^{d}\right) \tag{4}$$

This version of the equation evaluates all n KPIs by summing their normalized trust values and dividing by n, ensuring the overall trust score is an average across all metrics.

Capped Adjustment (CA): CA ensures that trust score adjustments based on performance deviations remain within acceptable bounds. The variable r determines the type of adjustment (robustness or penalization) based on the deviation of actual performance from the target. This adjustment is averaged across all KPIs to represent overall system performance.

By capping CA between -1 and 1, large performance variances do not cause drastic trust score fluctuations. This stabilizes the score, preventing any single performance issue from overly affecting it. The capped adjustment ensures that trust is earned or lost gradually, reflecting consistent performance rather than sporadic variations (see Equation 5).

$$\begin{aligned} &\mathrm{CA} = \max\left(-1, \min\left(1, w_{\mathrm{adj}} \cdot \frac{1}{n} \sum_{i=1}^{n} \left(\left|\frac{\mathrm{actual_{c/i}} - \mathrm{target_{c/i}}}{\mathrm{target_{c/i}}}\right| \cdot r\right)\right)\right) \\ &\text{where} \begin{cases} w_{\mathrm{adj}}: \text{ is a weight } | \; w_{\mathrm{adj}} + w_c + w_i = 1, \\ r: \text{ is the adjustment type variable whether is for} \\ \quad \text{penalization } (r = -1) \text{ or robustness } (r = 1). \end{cases} \end{aligned} \tag{5}$$

When performance falls short of the target, a negative adjustment is applied, accounting for the duration and frequency of violations (see Equation 6).

$$\begin{aligned} &\mathrm{CA} = \max\left(-1, \min\left(1, w_{\mathrm{adj}} \cdot \frac{1}{2n} \sum_{i=1}^{n} \left[\left|\frac{\mathrm{actual_{c/i}} - \mathrm{target_{c/i}}}{\mathrm{target_{c/i}}}\right| \cdot r \right.\right.\right. \\ &\qquad \left.\left.\left. - \left(\alpha \frac{\mathrm{duration_{c/i}}}{\mathrm{total\_period}} + \beta \frac{\mathrm{numberViolation_{c/i}}}{\mathrm{total\_transactions}}\right)\right]\right)\right) \\ &\text{where} \begin{cases} \alpha \text{ and } \beta: \text{ are constants such that } \alpha + \beta = 1, \\ \mathrm{duration_{c/i}}: \text{ is the duration of time the actual} \\ \qquad \text{value deviates from the target,} \\ \mathrm{total\_period}: \text{ is the total period over which the} \\ \qquad \text{performance is measured } (t - (t-1)), \\ \mathrm{numberViolation_{c/i}}: \text{ number of violations per KPI,} \\ \mathrm{total\_transactions}: \text{ total number of transactions.} \end{cases} \end{aligned} \tag{6}$$

In summary, we adopted this trust calculation approach for its balanced assessment of system performance, considering both recent and historical behavior. By averaging the trust score at time t with the prior value at $t-1$, the score evolves smoothly, reducing the impact of short-term volatility. Trust adapts to changing workloads and performance fluctuations through capped adjustments, limiting extreme deviations. Normalized KPIs ensure comparability across metrics with different scales, while weighted ratings for client and infrastructure trust prioritize performance aspects based on the environment's needs. Positive and negative adjustments reflect the frequency and duration of performance deviations, ensuring a fair and adaptive trust evaluation.

C. Trust Computation Workflow

The trust score at time t is calculated using Algorithm 1, which evaluates KPIs such as throughput and latency. The algorithm applies weighted aggregations, normalizes metrics, and adjusts for deviations from targets. Stability is maintained by averaging current and past trust scores, ensuring a reliable reflection of system performance and dependability.

Algorithm 1 Calculate trust score

1: **Input:** $T(t-1)$, actual/target KPI values, KPI types (d), durations, violations, w_c, w_i, w_{adj}, α, β
2: **Output:** Trust score $T(t)$
3: **Step 1: Normalize trust for each KPI**
4: Initialize normalized_T as an empty list
5: for each actual/target value and KPI type
6: Set $d = 1$ if "higher_better", else $d = -1$
7: Compare actual to the target value
8: Normalize and append result to normalized_T
9: end
10: **Step 2: Aggregate Trust Score (A)**
11: Compute aggregated trust score by averaging normalized_T for client and infrastructure
12: **Step 3: Calculate Capped Adjustments (CAs)**
13: Initialize capped_adjustments as an empty list
14: for each KPI
15: if "higher_better" and actual $\geq$ target **or** "lower_better" and actual $\leq$ target **then**
16: $r = 1$, compute robustness adjustment (Eq. 5)
17: else
18: $r = -1$, compute penalization adjustment (Eq. 6)
19: Append adjustment to capped_adjustments
20: end
21: **Step 4: Combine Capped Adjustment (CA)**
22: Compute combined capped adjustment by averaging capped_adjustments
23: **Step 5: Compute Final Trust Score**
24: Average combined A, CA, and $T(t-1)$
25: **Return** final trust score $T(t)$

III. Performance Evaluation

This section presents the experimental setup and measurement results used to assess our trust score computation model.

A. Experimental Setup

Figure 2 illustrates the experimental setup, showcasing the architecture for deploying blockchain-enabled applications. This setup simulates a federation of resource providers, demonstrating how blockchain and smart contract technologies are utilized. Kubernetes pods represent the CEC infrastructure, with trust scores computed using our model.

The core infrastructure comprises three Kubernetes-managed servers that orchestrate containerized applications and facilitate seamless communication. These servers interact with a blockchain node that manages transactions and ensures data integrity via a Proof of Work (PoW) consensus mechanism.

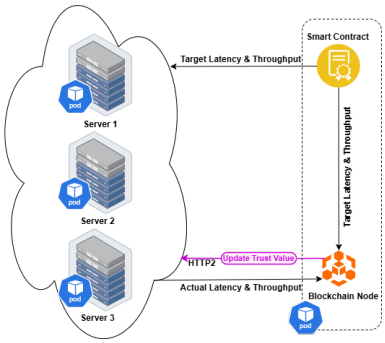


Figure 2: Experimental setup

- **Servers (1, 2, 3):** Manage containerized applications and enable network communication, ensuring high availability and scalability.
- **Smart Contract:** Automates SLAs by adjusting resource allocation or initiating tasks on the servers based on real-time performance metrics like latency and throughput.
- **Blockchain Node:** Secures and validates transactions using HyperText Transfer Protocol (HTTP)/2 for efficient communication.
- **Security:** Transport Layer Security (TLS)/Secure Sockets Layer (SSL) encryption secures communication between servers and the blockchain node.

This architecture, a simplified trust framework, verifies the proposed algorithm for autonomous SLA enforcement and trust management in cloud-edge environments. Blockchain and smart contract integration are key for transaction security, SLA automation, and performance validation.

B. Results

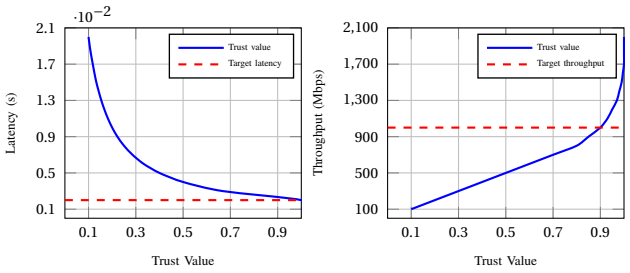
The performance evaluation focuses on monitoring SLAs, emphasizing the relationship between trust, latency, and throughput. This section compares our advanced model (Equations 2, 3, 4, 5, and 6) with the baseline model (Equation 1). The advanced model uses the following weights: $\alpha = 0.9$, $\beta = 0.1$, $w_i = 0.4$, $w_c = 0.4$, and $w_{adj} = 0.2$.

1) KPIs Impacts on the Advanced Trust Model: Figure 3 illustrates the trust value as a function of latency and throughput, with two subplots: 3a and 3b. Each subplot examines how trust correlates with a single KPI under varying network conditions. By isolating the impacts of latency and throughput, we gain a clearer understanding of how these KPIs influence trust. For reference, the target values for latency and throughput are shown in red: 0.2×10^{-2} seconds and 1000 Mbps, respectively.

- **Trust vs. Latency:** Subplot 3a shows an exponential decrease in trust as latency increases, consistent with the defined Algorithm 1 (i.e., line 6 in the case of $d = -1$). At very low latency levels (e.g., 0.1×10^{-2} seconds), trust is near maximum as latency remains close to the target.

However, even small increases in latency result in steep trust declines, highlighting its sensitivity to this metric.

- **Trust vs. Throughput:** Subplot 3b depicts an exponential increase in trust as throughput rises (i.e., $d = 1$). High throughput values (e.g., 900–1000 Mbps) yield trust values close to the maximum, as throughput approaches the target. Conversely, minor drops in throughput cause significant trust reductions, emphasizing its critical role in maintaining trust.



(a) Trust value as a function of latency

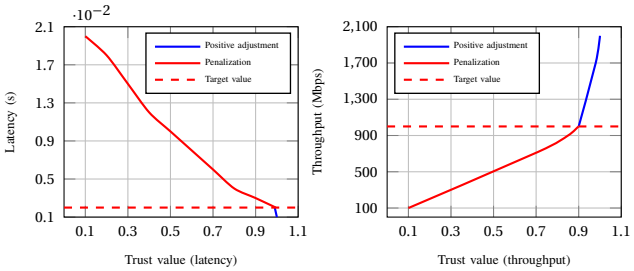
(b) Trust value as a function of throughput

Figure 3: Trust value analysis for latency and throughput metrics

In summary, trust is highly sensitive to changes in latency and throughput. Small increases in latency or decreases in throughput can sharply reduce trust, highlighting the effectiveness of our model.

2) Adjustment Impacts on the Advanced Trust Model: Figure 4 illustrates in depth how changes in latency and throughput during various network events affect trust values. The figure is divided into two subplots, each focused on a single KPI, enabling a detailed analysis of the impact of specific network parameters on overall trust.

- **Trust vs. Latency:** Subfigure 4a shows the relationship between latency and trust. The blue curve indicates an increase in trust as latency decreases (i.e., positive adjustment), while the red dashed line represents the target latency threshold, which should ideally not be exceeded to maintain trust. The penalization curve in red line highlights how exceeding this threshold leads to significant trust loss. Even small increases in latency beyond the target noticeably reduce trust, emphasizing the importance of stringent adjustment mechanism in trust management. The steep slope of the trust curve underscores the critical need to keep latency within acceptable limits.
- **Trust vs. Throughput:** Subfigure 4b examines the relationship between throughput and trust. The blue curve shows that trust increases with higher throughput (i.e., positive adjustment), while the red dashed line represents the target throughput level. When throughput falls below this target, trust declines, as depicted by the penalization curve in red line. This demonstrates the need to maintain adequate throughput to avoid penalization and thereby sustain the providers trust.



(a) Positive adjustment and penalization for latency

(b) Positive adjustment and penalization for throughput

Figure 4: Impact of the adjustment on trust value across different scenarios

In summary, Figure 4 demonstrates that both latency and throughput are vital for sustaining network performance confidence. It showcase the impact of the positive adjustment and penalization on the final trust value.

3) Violation Metrics Impacts on the Advanced Trust Model: To recall, the violation metrics—number and duration—are derived from continuous performance monitoring of each server:

- **Number of Violations:** This metric tracks instances when a server's performance deviates from the required level (e.g., due to increased latency or decreased throughput). Violations are recorded over a specified monitoring period (e.g., hourly or daily) for each server.
- **Duration of Violations:** This represents the total time the system operates in penalization mode, calculated by summing the durations of all individual violations. It reflects how long the server operated below optimal performance during the observation period.

Figure 5 illustrates the impact of violation metrics on trust across the three servers. The orange bars show the normalized duration of violations, while the blue bars indicate the total number of violations. The corresponding trust values for each server are shown as green lines. In the trust calculation model, we use $\alpha = 0.9$ and $\beta = 0.1$ to emphasize that the duration of violations affects trust more than their frequency, although these values can be adjusted as needed.

- **Server 1:** Despite having the highest number of violations, Server 1 maintains a high trust value (0.90) due to the short duration of violations. This demonstrates that trust is more sensitive to the length of disruptions than their frequency. Quick recovery helps preserve user trust.
- **Server 2:** With the fewest violations, Server 2 has the lowest trust value (0.60) due to prolonged violations. This highlights that even infrequent but long-lasting issues can significantly erode trust. Prompt recovery is essential for maintaining user confidence.
- **Server 3:** Server 3 exhibits a moderate level of both violations and their duration, resulting in a moderate trust value (0.80). This shows that both factors must be managed to ensure consistent trust.

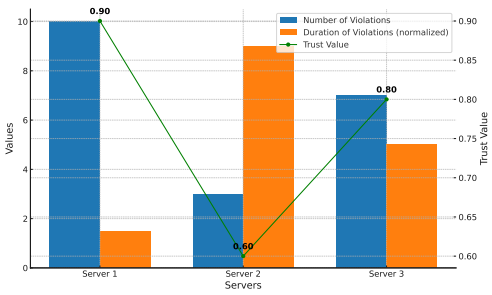
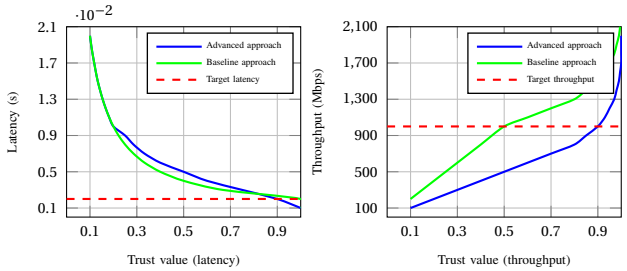


Figure 5: Impact of violation metrics on trust value across servers



(a) Trust value comparison regarding latency

(b) Trust Value Comparison regarding throughput

Figure 6: Trust value analysis: advanced vs baseline for latency and throughput metrics

In conclusion, Figure 5 shows that the duration of violations impacts the trust value more than their frequency, based on the model's weight, which can be adjusted as needed. Frequent issues should be resolved promptly to maintain trust, as prolonged disruptions have a significantly negative effect.

4) Advanced Trust Approach vs. baseline: In this section, we compare the trust metrics from our advanced approach (i.e., Equations 2 to 6) with those from a simple baseline approach (i.e., Equation 1). The analysis focuses on trust variations with respect to latency and throughput under both methodologies, aiming to evaluate how each approach affects the trust value across different network conditions.

Figure 6 illustrates the trust value comparison between the advanced and baseline approaches as functions of latency and throughput. The two subplots (Figure 6a and 6b) directly compare how each methodology influences trust based on these performance metrics. Red dashed target values are plotted for comparison, highlighting how closely each method aligns with the desired trust outcomes.

- **Trust vs. Latency:** In subplot 6a, both approaches show a decline in trust as latency increases. However, our approach exhibits a steeper drop, indicating a more sensitive response to increased latency. The red dashed line marks the target latency, showing how each method deviates from the ideal trust value as latency rises.
- **Trust vs. Throughput:** In subplot 6b, both approaches show an exponential increase in trust with higher throughput. Our approach, however, displays a more dynamic increase, suggesting a more responsive relationship between trust and throughput. The target throughput is marked with a red dashed line, illustrating how each approach aligns with the desired trust value.

Overall, the advanced approach is more sensitive to variations in latency and throughput, providing a more realistic trust computation than the baseline approach. This heightened sensitivity ensures that trust values more accurately reflect actual network performance, which is essential for effective real-time trust management.

IV. Conclusion

This paper proposed a trust management framework for CECC systems by integrating the IEEE SIIF FHS model with blockchain technology to enhance reliability, transparency, and scalability. The framework demonstrated the potential to address dynamic trust evaluation in federated environments by incorporating SLA compliance, performance metrics, and user feedback into comprehensive trust profiles. Future work will focus on developing adaptive methods for dynamically determining the weights of our model, ensuring the trust computation process is better aligned with the unique requirements of diverse application scenarios, thereby further optimizing trust evaluation and resource allocation.

Acknowledgment

This work was partially supported by the European Union's Horizon Research and Innovation program under AC^3 project and grant agreement No 101093129.

References

[1] I. Ullah, H.-K. Lim, Y.-J. Seok, and Y.-H. Han, "Optimizing task offloading and resource allocation in edge-cloud networks: a drl approach," *Journal of Cloud Computing*, vol. 12, no. 1, p. 112, 2023.

[2] L. Cao, T. Huo, S. Li, X. Zhang, Y. Chen, G. Lin, F. Wu, Y. Ling, Y. Zhou, and Q. Xie, "Cost optimization in edge computing: a survey," *Artificial Intelligence Review*, vol. 57, no. 11, p. 312, 2024.

[3] H. Yu, Z. Shen, C. Miao, C. Leung, and D. Niyato, "A survey of trust and reputation management systems in wireless communications," *Proceedings of the IEEE*, vol. 98, no. 10, pp. 1755–1772, 2010.

[4] P. D. Manuel, S. T. Selvi, and M. I. A.-E. Barr, "Trust management system for grid and cloud resources," in *2009 First International Conference on Advanced Computing*, 2009, pp. 176–181.

[5] J. Xu and C. Liu, "A survey on trust management for cloud computing and edge computing," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 8, no. 1, p. 5.

[6] H. Zhao, S. Stuckmann, and K. Kim, "A survey of reputation and trust systems for online service provision," *Computer Science Review*, vol. 1, no. 4, pp. 325–344.

[7] R. B. Bohn and C. Lee, "The ieee 2302-2021 standard on intercloud interoperability and federation," 2022.

[8] C. A. Lee, R. B. Bohn, and M. Michel, "The nist cloud federation reference architecture 5," *NIST Special Publication*, vol. 500, p. 332, 2020.

[9] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, "Bitcoin and cryptocurrency technologies: A comprehensive introduction," *Princeton University Press*.

[10] K. Papadakis-Vlachopapadopoulos, R. S. González, I. Dimolitsas, D. Dechouniotis, A. J. Ferrer, and S. Papavassiliou, "Collaborative sla and reputation-based trust management in cloud federations," *Future Generation Computer Systems*, vol. 100, pp. 498–512, 2019.