IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 November 2023, accepted 28 November 2023, date of publication 4 December 2023,
date of current version 18 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3339378

RESEARCH ARTICLE

Secured Dynamic Request Scheduling and Optimal CSP Selection for Analyzing Cloud Service Performance Using Intelligent Approaches

MUZAMMIL AHMAD KHAN[1,2], SHARIQ MAHMOOD KHAN[2], AND SIVA KUMAR SUBRAMANIAM[3]

[1]Department of Computer Engineering, Sir Syed University of Engineering and Technology, Karachi 75300, Pakistan
[2]Department of Computer Science and Information Technology, NED University of Engineering and Technology, Karachi 75270, Pakistan
[3]Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka 76100, Malaysia

Corresponding author: Muzammil Ahmad Khan (muzammilahmad.khan@gmail.com)

ABSTRACT Due to its heterogeneity, cloud services providers' (CSPs) rapid expansion presents several challenges, such as optimal service selection and privacy preservation. Multiple users using the cloud service at once increases the delay for service selection and request. Service interruptions result from centralized provisioning and insecurity. Existing work constraints include access control loss, service disruptions, security issues, trust management issues, and delays. Blockchain-based request scheduling and optimal CSP selection in edge-assisted clouds were presented in this research. Five phases—Data User (DS) authentication, sensitivity-aware request scheduling, policy verification, trust management, and optimal CSP selection—are proposed. In the first phase, DU authentication detects and eliminates authorized users. We suggested a chaotic map-based camellia encryption algorithm (CMCE) to boost security. The gateway schedules service requests using Johnson's rule-based Stochastic Gradient Descent method, considering delay, throughput, and priority, in the second phase. This schedules the request into sensitive and non-sensitive services. Policy verification is done in the third phase utilizing Dynamic Policy-based Access control, which allows only sensitive requests. In phase four, we calculate the CSP trust value to boost security. Based on CSP behavior, we introduced the Multi Behavior Analysis-based Nomadic People Optimizer method. Every CSP's trust value is modified based on user feedback over time. Finally, the best CSP is chosen for data user service, and suggested Dynamic and non-cooperative Game Theory is to choose the best CSP from a list. CloudSim is used to simulate and assess.

INDEX TERMS Blockchain, cloud computing, edge, scheduling.

I. INTRODUCTION

As the Internet of Things (IoT) becomes more common, numerous mobile intelligent devices gradually affect our daily lives. The typical centralization architecture design is not compatible with the mobility, flexibility, and quick reaction requirements of these intelligent devices, particularly for the cloud. The distribution of computing resources via the Internet, such as processing infrastructure, processing power, software, and data storage, is known as cloud computing [1].

The associate editor coordinating the review of this manuscript and approving it for publication was Rahim Rahmani.

Cloud computing offers several features, including cost-effective information interchange and convenient storage for consumers [2]. A convenient service, cloud storage has an efficient approach to managing massive amounts of data. It is used by a variety of end users, including businesses, organizations, and private individuals, for the goal of keeping both their commercial and personal data in the cloud [3]. The massive amount of information that connected devices broadcast to the cloud through the Internet is generating congestion, particularly around bottlenecks. The speed of the network and bandwidth will be impacted as well. Because one of the hallmarks of applications for

the IIoT is delay sensitivity, transmission delays result in lower quality of service (QoS), which hurts the experience of an end user [4]. To address growing concerns about cloud security and privacy, the security-on-demand service mode dynamically provides cloud users with trusted computing environments in response to their specific security requests. Designing suitable authentication and authorization solutions is one of the most important components of confidentiality and safety concerns in limited resources IoT gadgets [5]. An efficient method for predicting privacy breaches in cloud environments can be found in the design of cloud forensics based on blockchains. To improve privacy and security, this architecture blends the immutability and transparency of blockchain technology with the capabilities of cloud forensics [6]. Experts are extending cloud resources to the network's edge due to these issues [7]. Because of the indefinite extension of resource sharing and improved satisfaction with users, cloud computing has become known as one of the more important IT areas of study in the last few years [8]. Increased use of cloud computing services depends in large part on security [9]. Watermarking can significantly increase the security of data stored in cloud centers, but this does not guarantee that the CSP will win consumers' trust. To encourage the CSP to offer genuine services, it is vital to address the issues of fraud and user and CSP noncooperation [10]. A cloud service's trustworthiness can be defined as its capacity to deliver a dependable, secure, and robust service in line with customer expectations and assure that the operating behavior and execution outcomes satisfy the user's requirements [11]. The demand for Cloud services is rising along with the considerable advancements in network and Cloud technology over the past few years. The market is seeing an increase in Cloud service providers (CSPs). Each CSP promotes itself based on the characteristics that give its competitive advantage in the Cloud Computing Services Multi-Level Ranking Based on User Preference [12]. Because there are so many different cloud service providers on the market with varying levels of service quality, cost, and reputation, selecting one that will satisfy a given set of application needs is typically challenging and time-consuming [13]. Therefore, reducing energy usage in cloud data centers is necessary to increase profit for the cloud service provider (CSP), lower costs for consumers, and have less negative impact on the environment [14], [15]. As technology develops, new privacy issues arise, and the demand for privacy-driven strategies that foster trust between customers and cloud service providers rises [16], [17]. To achieve a variety of scheduling objectives, cloud computing necessitates a scheduler (broker) to work out how to most effectively distribute a limited amount of assets to the arrival activities and applications [18], [19]. We concentrate on offering comprehensive security to the cloud-IoT ecosystem so that it can fend off several internal and external attacks. By authenticating users and choosing a reliable CSP by meeting the SLA and QoS requirements, total security is attained.

A. MOTIVATION AND OBJECTIVES

From the DUs, a significant volume of service requests are received, and processing them all takes some time. Even though many research projects develop trust-based service discovery models; attackers can readily access and alter the trust values of CSPs. However, the following issues in secure service discovery remain unresolved.

- **Loss of Access Control**: Given that sensitive requests must be handled by the appropriate data owners, access control must be confirmed before services can be provided to the given requests. On the other hand, since they may be easily tracked and altered by malicious attackers, present access policy-based systems fail to give the best rule sets during verification.
- **Lack of Mixed Types of Services:** The DU's devices produce a significant number of service requests, which are simultaneously communicated to the cloud. SLA and QoS standards vary depending on the request. As a result, past studies did not focus on managing all sorts of services without prior knowledge of SLA and QoS.
- **Centralized Security:** The discussion of existing works includes security measures based on cryptography. These researchers used a centralized security paradigm that relies on keys that are kept on a device or server, which significantly compromises DUs' privacy. A decentralized security system is therefore required to simulate the provision of services.
- **Poor Trust Management:** In a cloud context, each CSP's and service's trustworthiness is calculated to provide genuine services in response to user requests. Attackers in this situation can easily change the trust values to target CSPs and hack any or all sensitive sorts of services. Therefore, it must be checked before serving the DUs. Resources, execution time, and SLA history are the only few aspects that are taken into account for trust management.
- **Large Delay:** When the cloud server receives requests for all kinds of services. To reduce the delay and meet the SLA and QoS requirements of the received request, it causes enormous delays in handling all types of requests.

The primary objective of this study is to reduce response times in request computation by achieving secure computation of user requests using the suitable CSP. The following sub-objectives are accomplished to reach this main goal:

- To increase user confidentiality by using multi-factor authentication, enabling only authorized users. We advocated blockchain-based authentication to increase security.
- By dynamically arranging incoming requests in a cloud environment, it is possible to increase user request success rates and lower overall latency.
- To provide access control for the authorized users and sensitive requests during request scheduling and the best service selection, thereby reducing security threats

and computational complexity imposed on by internal attackers.
- To choose the CSP most effectively based on computed trust values of available CSPs to increase computation reliability.

B. RESEARCH CONTRIBUTIONS

The main objective of this study is to protect the cloud-IoT environment from a variety of internal and external assaults. The key benefits of this strategy are listed below,

- When choosing a service for a given service request in a cloud environment, the process of optimal service selection minimizes the search latency as opposed to selecting a random service.
- CSP trust calculation improves CSP security in a cloud context, increases CSP selection efficiency, and decreases service threats and provisioning latency.
- Scheduling enables request priority, which lowers latency when sensitive requests are sent to the service.
- Blockchain offers the CSP more safety and helps prevent fraud and illegal access to the service. Hash-formatted data makes it impossible for hackers to view the data or compromise the service.

C. PAPER ORGANIZATION

The subsequent portions of the paper are structured as follows: Existing research gaps are assessed in the literature reported in Section II. Section III focuses on an overall problem statement. The planned work is extensively outlined in Section IV along with the necessary pseudo code and visuals. The experimental setup, in-depth descriptions of the simulation setup, a comparison analysis, and a study summary are all found in Section V. The conclusion of the planned task is thoroughly covered in Section VI.

II. LITERATURE SURVEY

The research gaps in existing papers are listed below. A graph-based network is proposed in [20], Degree and betweenness centrality are calculated for each network. SLA and security capabilities, SLA history, and cloud service performance are used to calculate cloud service provider trustworthiness using the Analytic Hierarchy Process (AHP). The function association between cloud service provider and cloud service variables is investigated using regression. The SLA depends on cloud service availability. Based on SLA history, CSP trustworthiness is estimated well, but SLA & QoS vary for sensitive and non-sensitive services, which is not focused in this research, reducing trust value and network performance.

The author in [21] proposes a secure service delivery approach for adjacent mobile devices. This model is used in mobile ad hoc clouds. High-service requests overload cloud servers. Thus, service handling computations are offloaded from the lower edge node load to the higher load. Resource-aware service offloading is used in mobile ad hoc edge clouds. Cloud servers are divided into Master, Serving, and Requester Nodes. Requester nodes send offloading requests to the edge, which distributes them to serving nodes. Each node resource level and estimated execution time are calculated using the Genetic Programming Model to pick serving nodes. Here Offloading is incomplete due to static edge node deployment. If there are fewer serving and requester nodes, tasks are not completed within SLA and QoS standards.

The author [22] developed a technique to safeguard cloud users' data and identities by maximizing privacy. The suggested solution allows cloud users to use cloud service providers' services anonymously without their identity being revealed. Their technique outperforms anonymous authentication alternatives in computation and communication costs. A lightweight authentication methodology reduces the computational burden of large authentication methods. Elliptical curve cryptography generated keys. Elliptical curve encryption was used for authentication, but its implementation was difficult, causing scalability issues.

In cloud environments, the author [23] proposed an optimal cloud service selection technique. This paper proposes two multi-criteria decision-making (MCDM) algorithms for selecting the best CSPs: technique for order of preference by similarity to ideal solution (TOPSIS) and Best-Worst Method (BWM), which rank CSPs by reputation and reliability in serving user service requests. Factors such as scalability, sustainability, usability, interoperability, security management, cost, maintainability, service response time, and reliability contribute to the overall weight of each cloud service provider (CSP). The best CSP is a node with a high cumulative weight.

The author [24] suggested the best cloud service option. The Multi-objective Optimal Service Selection (MOSS) algorithm has five steps: prologue, ranking, assessment, integration, and selection. The technique uses decision-makers to choose the best cloud servers based on QoS and Quality of Experience (QoE). The first stage identifies qualified and unqualified services. The Decision Makers (DM) then aggregates similar services. The author estimates QoE and QoS parameter weights in the second stage. The BWM algorithm was proposed in this research. In the fourth level, service rank determines optimal services. Final performance evaluations include QoS and QoE. Compared to others, the proposed technique performs well. Here the suggested BWM algorithm uses MCDCM, which computes rank value and best service selection, increasing delay Service provisioning and request scheduling for edge-enabled blockchains were proposed.

The author in [25] uses two-stage optimization for request scheduling. A highly effective service provisioning decision-making method is proposed. The suggested optimization technique for request scheduling has low convergence, resulting in excessive delay. Request scheduling was suggested for cloud-fog computing to reduce latency.

This study proposes a genetic method for request scheduling to reduce latency in [26]. It also fixes cloud-fog scheduling. The proposed method is compared against

waited-fair queuing (WFQ), priority-strict queuing (PSQ), and round robin. The suggested work outperforms other state-of-the-art techniques in latency. A genetic algorithm is proposed for request scheduling, but it takes a long time in large-scale environments, increasing delay.

The author [27] presented a weighted approach brokering model for choosing cloud services. Four steps, including service selection, user feedback, feedback aggregation service, and dataset modules, are part of the planned effort. Users' requests and comments for cloud services are gathered in the initial phase. The initial phase is in charge of gathering and handling the feedback for the second module. To get customer feedback, fuzzy logic is applied. The proposed work is evaluated using the benchmark module for availability, reaction time, and reliability.

The novelist recommended [28] a way to choose the superlative cloud provider by factoring in QoS considerations. The value of the QoS characteristics is clustered using the K-means clustering technique, and similarity is determined by the distance between the clusters. The weight values of the QoS qualities are calculated using precision rough set theory, and this work selects the best services based on the weight values. The outcome demonstrates that the suggested strategy outperforms other ways in terms of service supply. Here, the best service is chosen based on the QoS weight values, which are insufficient for the best CSP choice and result in low accuracy.

To assess the security of cloud service providers, the author [29] suggested a fuzzy inference method. Calculating the trust value of the CSP involves fuzzy reasoning. The three components of the proposed work are defuzzification, inference engine, and fuzzification. The input values are translated to a fuzzy set with membership functions in the first block. Each Cloud Service User (CSU) uses these circumstances to determine the trust value of CSP. The fuzzy set comprises damaging, average, dangerous, and exceptional conditions. Fuzzy rules are applied to the input in the second block to provide a fuzzy output that the CSU can compute. It is in charge of classifying the output in the third block to determine whether the CSP is secure or not. The outcome demonstrates that the suggested strategy outperforms others in terms of efficiency.

The author in [30] suggested a Gaussian -TOPSIS algorithm-based method for choosing a cloud service to address the rank reversal problem. Numerous elements, including a cloud broker, a CSP, and a cloud service repository, are included in the proposed system design. This research suggested the G-TOPSIS algorithm, a decision-making mechanism, for choosing the best service. It is used to rank the services by the quality of service that the CSP offers. In a cloud environment, the best cloud services are chosen based on QoS values. All of the information about cloud services and QoS is kept in the cloud service repository. Here, the CSP's data is all publicly stored in a cloud service repository, which lowers the CSP's security because attackers may simply breach it.

The author [31] developed multi-dimensional CSP trust calculation in a cloud environment. The proposed work consists of several parts, including a service level agreement (SLA) agent, cloud broker, and cloud auditor. Here, trust scores for CSP and cloud users are computed. The TOPSIS algorithm is used to calculate QoS during the trust calculation. Two methods—direct trust and indirect trust—are used to calculate the CSP's ultimate trust value. According to the simulation results, the suggested model outperforms the current system in terms of trust calculation. Here, CSP trustworthiness is assessed based on QoS, which is insufficient for trust computation, degrading security performance, and this research does not meet SLA requirements Based on QoS and user input in the cloud environment, the author [32] presented a tiered trust management system for CSP. Numerous elements, including cloud users, cloud service providers, trustworthy third parties, and cloud service registries, are part of the proposed architecture. Local subjective trust evaluation and global subjective trust evaluation are the two different types of trust evaluation methods that are suggested. The simulation outcomes validate that the suggested model outperforms earlier research in terms of performance.

The author presented [33] a new architecture based on Deep Reinforcement Learning (DRL) for outsourcing computation-intensive PoW activities to edge servers in a blockchain-based Mobile Crowdsensing (MCS) system. The proposed approach can be used to identify the appropriate offloading strategy for Proof of Work (PoW) tasks in the complex and dynamic MCS environment. Simulation findings suggest that our solution can achieve a lower weighted cost of latency and energy consumption when compared to benchmark alternatives. here Blockchain systems, particularly those that use PoW, commonly run into scalability problems as the number of transactions and users increases.

A blockchain-based cooperative unrestricted inspecting system for active records was suggested by the author [34]. The CSP is created so that it can produce a challenge established using the most recent block hash. In the challenge phase, it is not required to communicate with the blockchain, significantly lowering communication overhead. They also give users the option to look for partners to lower audit expenses while taking economic aspects into account. The likelihood of malicious users engaging is effectively reduced by the Eigen Trust model, which assesses the reputation of each user's audit behavior. Different users and systems may employ various blockchain platforms or technologies in this multi-user scenario. It is important to ensure data consistency and interoperability between various blockchains or blockchain networks.

The author in [35] proposed Blockchain-based Intercloud Resource Discovery (BIRD), a system for intercloud resource discovery based on blockchain, It entails participating CSPs linked in a peer-to-peer network that uses blockchain to handle resource data as well as retain transactions. The BIRD architecture eliminates the need for a dependable

third party to find and manage resources. The BIRD framework's primary components include QTR (Quality-of-Service, Trust, and Reputation) indices, a fine-grained control mechanism, and latency optimization. Faster resource discovery is achieved by latency optimization, intercloud resource discovery is controlled with finer granularity, and QTR stands for quality CSP or source collection. Blockchain is used by BIRD to securely maintain transactions between CSPs. The use of blockchain to record transactions and interactions raises data privacy concerns, particularly when sensitive data is stored on the blockchain.

Using an energy-efficient dynamic decision-based strategy, the author of [36] proposed a unique task scheduling algorithm that addresses energy consumption and time execution. The proposed method accommodates handheld time and energy-intensive computations in addition to cloud computing workloads with ease. In addition, we introduce a unique task scheduling server that offloads computing on the cloud, improving the handheld capability for creating decisions and its computational performance. In the task scheduling process, the proposed empirical algorithm is applied. The study's conclusions enable effective employment scheduling, which cuts energy usage and job scheduling significantly.

To address and resolve several security and privacy-related concerns, the author [37] proposed Consumers and CSPs can both benefit from blockchain-enabled solutions for virtualized cloud services. These technologies will aid the data center sector in enhancing its infrastructure for resource provisioning and virtualized cloud services. Finally, we talked about the difficulties associated with using blockchain in cloud infrastructures. Governments have adopted cloud services, but the report only notes this; it doesn't go into detail on the difficulties and risks associated with moving sensitive government information and services to the cloud, including issues with regulatory compliance and data sovereignty.

Based on the blockchain concept, the author suggested [38] a growing modern algorithm-automated forensic platform. This suggests peer-to-peer Designing forensic structures, acquiring proof, and storing it on a blockchain are all part of the process. Unauthorized users will be secured via the Secure Block Verification Mechanism (SBVM). The cuckoo search optimization technique produces secret keys as efficiently as possible. At the cloud authentication server, all data are saved and encrypted for privacy. Confidentiality-based algebraically the learning of cryptosystems is provided homomorphism, a new encryption technique. In the SDN controller, each piece of data is given a block, and the history is preserved as metadata. This article [39] suggests using symmetrical encryption, smart contracts, and blockchain networks to track the reliability of files stored in the cloud. The proposed method includes a protocol that provides privacy, decentralized management, auditing accessibility, and safe exchange of data integrity monitoring findings without charging the participating services, as well as entire reference implementations that were used to validate the proposal. The validation tests revealed that the solution is reliable and error-free in recognizing files that have been damaged. These studies also show that by exchanging the results of consistency monitoring and applying computational confidence mechanisms, the effectiveness of the proposed strategy was considerably increased. The difficulties of scaling blockchain technology in a large-scale, practical cloud storage environment are not covered in this study. Blockchain networks can experience problems as the volume of transactions and users grows.

The author suggested [40] that a deep reinforcement learning-enhanced two-stage scheduling (DRL-TSS) model is suggested to handle the NP-hard issue of execution difficulty in end-edge-cloud Iot of Things systems. This model may distribute computing resources inside an edge-enabled infrastructure to guarantee that computing jobs are executed at the lowest possible cost. A presorting technique based on Johnson's rule is devised and utilized to pre-process the two-stage tasks on numerous executors. The entire makespan is then minimized using a newly developed instantaneous reward that incorporates the maximum usage of each executor in edge-enabled two-stage scheduling. Their approach's effectiveness is evaluated and compared to three different scheduling strategies. Here, the suggested algorithm needs to be evaluated for generalizability across a range of IoT application domains. Scheduling efficiency may be impacted by the particular requirements and limitations of various IoT use cases.

The author in [41] suggested a framework for cloud services that considers user favorites and selects the best cloud service based on those choices and the user's QoS limitations. They suggest a "principle component analysis (PCA) and best-worst method (BWM)-based approach" for choosing cloud services that remove correlations between QoS and offer users the best cloud services with the best QoS values. Finally, a numerical example is provided to demonstrate the viability and efficacy of the suggested methodology. When working with numerous cloud service providers or a sizable dataset, PCA and BWM may be constrained by their processing complexity. It can take longer and require more resources to select.

However, subsequent studies have addressed several significant challenges were represented in Table 1.

III. PROBLEM STATEMENT

The primary problems identified in this research include the loss of access control, the absence of mixed types of services, centralized security, poor trust management, and significant delay. As explained below, this section covers particular research-related difficulties.

The author of [20] suggested analyzing the dependability of cloud service providers and forecasting their performance in terms of SLAs. The suggested network is built using a graph-based architecture. Degree centrality and betweenness

TABLE 1. Research gap in literature survey.

References	Objectives	Methods or Algorithms Used	Limitations
[20]	To assess the reliability of CSPs and forecast their performance in terms of SLAs.	AHP	Multi-Service Interference and Performance Degradation
[21]	To protect the service delivery mechanism used to provide services to adjacent mobile devices.	Heuristic algorithm	Delay-sensitive Offloading Optimization
[22]	For secure cloud computing services, a simple anonymous authentication method.	Elliptical curve cryptography	Scalability issues.
[23]	Select the best CSP based on a variety of factors to maximize the effectiveness and QoS requests in a cloud-based environment.	MCDM	Complex Consistency
[24]	To create the best cloud services that give equal weight to QoS and QoE	MOSS	high latency
[25]	Provide a way for efficiently gathering and scheduling user service requests, offloading duties, and choosing the best service for edge-enabled blockchain systems.	Two-stage optimization scheme	Increased latency
[26]	To provide request scheduling issues and proves superior performance in cloud-fog computing settings.	Genetic algorithm	High latency
[27]	To improve service ranking and selection by taking into account a diversity of quality issues	Fuzzy logic	poor security
[28]	To present a technique for picking the best cloud services based on QoS criteria.	K-means clustering algorithm	poor accuracy
[29]	To provide a fuzzy inference system to assess the level of safety offered by CSP	Fuzzy logic	Inaccurate Trustworthiness
[30]	To provide a method for choosing the best cloud services in a cloud computing environment based on their QoS characteristics	G-TOPSIS	lack of Security in Cloud Service
[31]	To offer an MDTES that makes it easier for CCs to assess the credibility of CSPs from numerous angles.	Multi-dimensional Dynamic Trust Evaluation Scheme (MDTES)	Lack of Comprehensive Malicious Behavior
[32]	To increase the reliability of CSP trust evaluations by resolving the issues with low data quality in feedback ratings for cloud service providers CSP	multi-level trust management framework (MLTM)	Misclassification of Malicious Feedback
[33]	To reduce latency and power consumption in a blockchain-based MCS system by optimizing the offloading policy for computation-intensive PoW workloads.	DRL	Integration Challenges of Blockchain in MCS Systems.
[34]	To assure the integrity of cloud data while addressing the problems with data leakage and single points of failure related to third-party auditors (TPA)	RSA and AES algorithms	Efficiency-Security Trade-off Challenge
[35]	To provide a safe, decentralized, and effective system that does not require a trusted third party to improve source detection in the non-federated intercloud.	BIRD	Scalability issue
[36]	Introduce a cutting-edge task scheduling system that seeks to improve time and energy use on mobile devices.	energy-efficient dynamic decision-based method	Network overhead
[37]	Enhance the file's safety, privacy, and availability for both end-users and cloud service providers.	consensus algorithm	insufficient specificity
[38]	To propose a blockchain-based automated forensic platform to improve the security and effectiveness of digital forensic procedures while preventing cybercriminal activity.	CB-EL GAMAL Algorithm, Cuckoo Algorithm, SHA-3-512	Lack of Real-World Deployment
[39]	To present a comprehensive approach for ensuring privacy, decentralization, check accessibility, and safe allocation of monitoring outcomes for files stored in the cloud.	Practical Byzantine Fault Tolerance (PBFT) algorithm	Cryptocurrency Integration Complexity
[40]	To create a DRL-TSS for end-edge-cloud IoT systems and effectively distribute calculating incomes within an edge-enabled structure.	Johnson's Rule-based Genetic Algorithm (JRGA)	Scalability issue
[41]	To design a system that takes into account user preferences and QoS limitations while doing away with interdependencies across QoS parameters.	PCA and BWM	Dimensionality Reduction Challenge

centrality are determined for each graph. The SLA and security capabilities, history of the SLA, and performance of the cloud service are used in conjunction with the AHP technique to determine the trustworthiness of the CSP. The function relationship between the CSP and the cloud service variables is investigated using the regression approach, which is proposed. The cloud services' availability is the foundation of the SLA. The following is a list of this strategy's primary drawbacks.

- While SLA and QoS are varied for sensitive and non-sensitive services that are not focused in this study, which lowers trust value and network performance, CSP

trustworthiness is estimated here based on the history of SLA.

- The suggested system manages a variety of services without scheduling, increasing latency while accessing cloud service providers for sensitive services, lengthening high scheduling times, and lowering QoS.
- Because it is not explored in this paper, the accuracy and trust in CSP are decreased because the trust value of CSP needs to be updated (feedback) at every level to improve trust.
- All users, whether authorized or unauthorized, have access to cloud services, which increases complexity and data security risks while lowering CSP security.

A novel security framework for cloud services with customizable policies was suggested by the author in [3]. The framework consists of two main parts, such as the attribute-based signature (ABS)-based remote attestation approach. Six elements, including a cloud customer, registry, cloud manager, AA, blockchain auditor, and server cluster, are part of the suggested design. Attribute information is used to validate policies. Additionally, policy validation is utilized to choose the best physical and virtual machines for a given customer request. For evaluating the reliability of VM migration, which is carried out by the cloud manager, a policy-customized migration protocol is provided. For increased security, all communications between customers and cloud managers are recorded in the blockchain auditor. The following is a list of this strategy's primary drawbacks.

- In this case, all users are permitted to access the service, which increases process complexity and latency when many users utilize the service at once. This compromises security and degrades performance.
- The calculated VM migration trust in the proposed system is based on a policy-customized migration protocol, which provides high trust, but unnecessary and frequent VM migrations add overhead, which lowers overall network efficiency and lengthens processing times.
- Although blockchain technology is suggested here to boost CSP security, it does not consider the trustworthiness of each CSP, which results in poor CSP selection and higher latency.

A secure service delivery model is offered for serving the surrounding mobile devices by the author in [21]. The mobile ad hoc cloud environment is where this paradigm is executed. When there are a lot of service requests, cloud servers are overloaded and unable to provide the services. As a result, the computations for service handling are offloaded from the edge node's lower load to its higher load. In a mobile ad hoc edge cloud environment, resource-aware service offloading is given as a solution. The three classes of cloud servers in this example are the Master Node, Serving Node, and Requester Node. If a requester node sends offloading requests to the edge, such requests are assigned to the serving nodes that are available. The genetic programming model is used to determine each node's resource level and predicted execution time to choose the available serving nodes. The following is a list of this strategy's primary drawbacks.

- It is assumed that all nodes (the master, requester, and serving nodes) are trustworthy and prepared to offload tasks (service requests) from the source node to the target node for this operation. As a result of the loss of trustworthiness computation, SLA requirements are achieved for both incoming tasks and offloaded jobs.
- In real-time crises, request demands frequently vary. In these circumstances, edge nodes are dynamically needed, and the cloud server must be in charge of new edge node deployment. However, this study failed to provide this answer, making it unsuitable for real-time scenarios.
- The genetic programming model used in this research has poor service selection due to a lack of service selection factors (e.g. trust, load), which results in high latency and decreased process efficiency for sensitive requests.
- Offloading is incomplete due to the deployment of static edge nodes. In addition, jobs are not completed within the SLA and QoS standards if there are fewer nodes (serving and requester).
- Due to resource and execution time offloading, the execution time for time-sensitive service requests is increased. Because the type of service is not taken into account, latency is higher. Furthermore, processing a genetic model takes a long time for both local and global search operations.

The author of [6] suggested an ensemble learning model to address the challenges with cloud service reliability. In particular, appropriate weights are assigned and the ensemble is created using a ''back propagation neural network (BPNN). Traditional PSO and Quantum Discrete PSO (Q-PSO)'' are employed for the computation of binary and decimal weights, respectively, to enhance the performance of BPNN. A variety of subsets are created from a collection of past historical service records concerning CSPs. Then it is applied to the BPNN with heuristic methods, a Selective Neural Network Ensemble Learning Model. Feedback is also taken into consideration when calculating the weight values and is used to update the services' level of reliability. The following is a list of this strategy's primary drawbacks.

- This work does not demonstrate how to assess the credibility of mixed (sensitive and non-sensitive) service requests. Additionally, QoS changes over time, making it crucial for dynamically calculating a service's trustworthiness.
- As the number of requests rises, the computational overhead rises as a result of the combination of three conventional algorithms, including BPNN, PSO, and Q-PSO. As a result, the latency in finding sensitive services in a cloud environment is increased by this type of service trustworthiness prediction.
- The historical data and feedback of CSP are gathered and stored in a way that is easily vulnerable to attack,

reducing the security of CSP and raising the risk of high service.

In [4], the author presented a blockchain-based safe access control method for Internet of Things (IoT) systems. A lightweight access control technique was created after the shortcomings of previously employed access control strategies were examined. Each layer of the IoT architecture has a blockchain manager installed for safe access control. Four agents—a ''signature verification agent, an authentication agent, an authorization agent, and an encryption-decryption agent''—make up each block manager. Due to its resource limitations, the local blockchain manager that manages IoT devices does not come with an encryption and decryption agent. The five classes of transactions between IoT devices, fog nodes, and cloud servers are access, update, monitor, add, and remove. Only the blockchain manager, which employs Mandatory Access Control (MAC) for multi-level security, had access to the access control. The following is a list of this strategy's primary drawbacks

- The use of block managers enhanced security at each stage of the IoT architecture, but the lack of encryption and decryption agents at the IoT device layer puts data integrity at risk before it reaches the fog layer.
- The authorization agent assigns a trust value to each IoT device, but it does not take into account the trust values of different cloud service providers, which has an impact on the data's secrecy.
- The Bell-LaPadula model, which uses static policies and was determined to be ineffective at providing multi-layer security, was used to offer access control. This had an impact on the mechanism's security.

Research Solutions:

This study suggests a thorough strategy to improve the dependability and effectiveness of CSPs in an edge cloud environment. Based on variables including historical completion records, SLA history, request processing time, service accuracy rate, and resource utilization, it calculates the trust rate of CSPs. To prioritize sensitive service requests and cut down on delays, a sensitivity-aware request scheduling approach is introduced. End-user feedback is gathered to dynamically update each CSP's trust rating. Dynamic policies ensure security and QoS requirements while reducing time complexity and communication overhead with the proposed gateway-based edge cloud architecture. The security, scalability, and effectiveness of the solution are further improved by authentication, dynamic edge server deployment, and trust assessment utilizing meta-heuristic techniques.

IV. PROPOSED METHOD

The proposed effort focuses on providing security to Cloud Service Providers (CSPs), to achieve high confidence upon service access. The entire architecture of this research is depicted in Figure 1 below. The proposed work is divided into five successive phases:

- DU Authentication
- Sensitivity aware Request Scheduling
- Policy Verification
- Trust Management
- Optimum CSP Selection

A. DU AUTHENTICATION

To increase security, the trusted authority (TA) delivers the data users' (DU) registration information, including their device ID, password, PUF, fingerprint, and IRIS. High security is achieved via the blockchain's storage of data in a hash format that cannot be altered by attackers. A secret key is sent to the DU by the TA once registration is complete. The secret key is used to perform authentication. To address this, we put out the CMCE (Chaotic Map-based Camellia Encryption) algorithm. The usage of authentication processes increases security by allowing only authorized users to access the service. We have decreased the computational complexity during request scheduling and the best service selection by removing illegitimate users.

1) CHAOTIC MAP

The fundamental idea behind encryption may be summed up in two steps: (a) scrambling the data locations. b) Modifying the intensity of the data one or both tasks are accomplished using chaotic maps. According to the subsections that follow, CAT maps are utilized for the first purpose, and logistic maps are used for the second. Chaotic maps should have a high number of parameters, be robust, and have mixing properties.

a: ARNOLD CAT 2-D MAP

The CAT map is a 1-to-1 mapping technique used to repeatedly swap data positions in the spatial domain. The data will be mixed up after using the Arnold Cat 2-Map for a certain amount of iterations, but each pixel will still retain its original value. Because of this, the logistic map and camellia S box are employed to change the data value.

b: THE LOGISTIC MAP

The logistic map has the following equation, which is a quadratic function.

$$Y_{s+1} = rY_s\,(1 - Y_s) \tag{1}$$

The logistic map's control parameter $p \in [0, 4]$ is represented by the variable $Y_s \in [0, 1]$, where s is the number of iterations used to produce the iterative values. Keys include p, y, and the number of iterations s.

2) CAMELLIA ENCRYPTION

When compared to ''Advanced Encryption Standard (AES)'', Camellia is one of the block cipher algorithms that offer Symmetric keys with 'block sizes of 128 bits and key sizes of 128, 192, and 265 bits'. To generate a secret key with a size of 192 or 256 bits, 18 or 24 moves are required. Additionally, it makes use of four 8 X 8-bit S boxes that provide affine transformation, logical operations, input, and output. Here, plaintext and cipher text are both 128 bits. Algorithms for

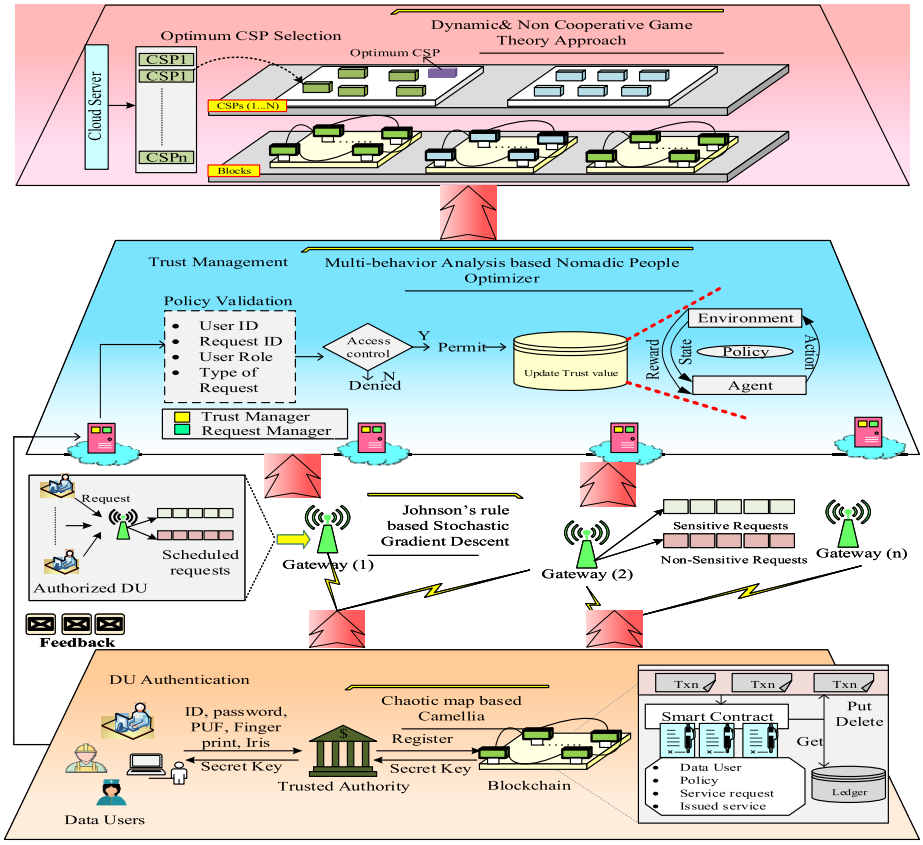

FIGURE 1. Overall architecture of this research.

encryption and decryption run repeatedly using the S box. The number of repetitions depends on the length of the current key. Figure 2 shows the Camellia Encryption and decryption process.

- If the algorithm needs to repeat six round blocks three times to generate a 128-bit secret key.
- If the algorithm requires four iterations of 6-round blocks for secret keys with 192- or 256-bit length.

The data blocks are encrypted using a secret key that comprises a 64-bit sub-key and four 64-bit variables that are explained as follows:

$$K_{LL} = 64 \text{ left bits of } K_L \tag{2}$$
$$K_{LR} = 64 \text{ left bits of } K_L \tag{3}$$
$$K_{RL} = 64 \text{ left bits of } K_L \tag{4}$$
$$K_{RR} = 64 \text{ left bits of } K_L \tag{5}$$

B. SENSITIVITY-AWARE REQUEST SCHEDULING

After completing authentication, the data user sends a service request to the CSP. Before that, the gateway collects the service requests from the DU which sends mixed types of services; hence we need to schedule the service request to reduce latency which helps to utilize the available resources. Scheduling reduces the waiting time for the service request in the cloud environment. For that, we proposed **Johnson's rule-based Stochastic Gradient Descent algorithm** that considers delay, throughput, and priority for scheduling the request. In our work, we give priority to sensitive services so it schedules first and then schedules the non-sensitive tasks.

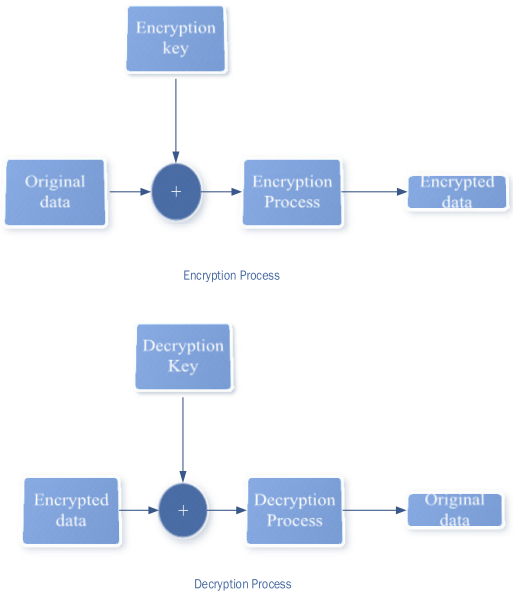

FIGURE 2. Camellia encryption and decryption process.

1) JOHNSONS RULE

This method is used to schedule requests or jobs in the work centers and it eliminates idle time between the work centers when scheduling the jobs that will shorten the makespan. Johnson's rules' methodology is summarized as follows:

Step 1: List each work center's requests or tasks along with their deadlines.

Step 2: Select the task with the lowest activity time. The algorithm will schedule the jobs based on this value.

Step 3: Remotely start the bare-bones task before continuing.

Step 4: Continue with steps 2 and 3 until the entire task in the environment is scheduled.

Johnson's rule is a conventional work sequencing rule that was first created for a flow shop with two machines. Johnson's rule can be used if N_i, a new processing time, is assumed to be the average processing time of each work for machines 1 and 2 at stage one. N_i is determined as follows:

$$N_i = \frac{\left(N_{i,1} + N_{i,2}\right)}{2\,(1)} \tag{6}$$

The final task schedule will be generated for each job by applying the standard Johnson rule to compare N_i with the processing time at the assembly machine q_i at stage two.

2) STOCHASTIC GRADIENT DESCENT ALGORITHM

The descent algorithm, which uses the gradient function to find optimal values, is used to optimize objective functions with the right attributes using this technique. It is employed in numerous applications to address optimization issues. The following is a definition of the optimization problem:

$$V := V - \mu \nabla Q\,(v) = v = \frac{\mu}{n} \sum_{i=1}^{n} \nabla Q_i\,(v) \tag{7}$$

where $Q_i\,(v)$ represents the value of the loss function $Q\,(v)$ represents the empirical risk and μ represents the learning rate. It reduces the objective function. The stochastic gradient descent algorithm is similar to the gradient descent. The difference between the two algorithms is when gradient descent the subset of parameter vector is randomly chosen for every iteration. The prediction of gradient reduces the cost of computation and increases the processing speed which increases the stochastic gradient descent performance.

The process of stochastic gradient descent is defined as follows,

Step 1: Select the initial vector of parameters V and learning rate μ

Step 2: Repeat until an estimated minimum is obtained

Step 3: shuffle the examples in the training set

Step 4: for i=1, 2,...n do

Step 5: $V := V - \mu \nabla Q\,(v)$

The SGD for scheduling is as follows. Assume that the appointment time vector is b^s at phase s of the algorithm. One draws a novel sample Y_t of the random vector Y. One then estimates the model gradient $\nabla_b D\left(b^s, Y^S\right)$ YS concerning b and takes a step in the opposed direction of the gradient. That is,

$$b^{s+1} = b^s - \eta_s \nabla_b D\left(b^s, Y^S\right) \tag{8}$$

$\eta_s > 0$ is the phase size, and we agreed it to be $\eta_s = \mathrm{O}\left(1/\sqrt{s}\right)$.

C. POLICY VERIFICATION

After completing scheduling, the policy verification process is performed; in our process we are given an access control for sensitive service requests. For that, we evaluate the service policies by Dynamic Policy-based Access Control method. The policies are generated and verified based on user ID, request, user role (behavior of the user), and type of request, if the policies are accepted then we permit to access the service otherwise the request will be denied from the environment.

We use T, P, F, and B to refer to the "subject, object, environment, and action, respectively". In Section II-C, Q represents the policy center. To represent all entities in our model, we will use the set T = {Q, T, P, F, B}. The Dynamic Policy-based Access Control model's encryption technique is defined formally using these methods.

For entities T = {Q, T, P, F, B}, a Dynamic Policy-based Access Control system includes five algorithms.

1) Setup (T) $\rightarrow$ (Qk_s, Tk_s): It receives a system parameter T and generates a public/private key pair (Qk_s, Tk_s) for entity s in T.
2) Encrypt $\left(Qk_Q, Fk\right) \rightarrow CFk$: This function uses the public key Qk_Q of policy center Q and a session key Fk to generate the ciphertext CFk of Fk.
3) The PolicyGen function converts the private key Tk_Q of policy center Q, the cipher text CFk, the access policy Π, and the nonce τ into the cryptographic representation $C\Pi$ of policy Π.
4) TokenGen (Tk_T, b, τ) produces Token Ub, τ from private key tkt, entity t attribute b, and nonce τ.
5) Decrypt $\left(Qk_s, \left\{U_{b,\tau}\right\}\right) \rightarrow Fk$: The algorithm uses public keys Qk_s, Tokens $\{U_b\}, \tau$, cipher text Cfk, and policy Π cryptographic representation $C\Pi$ to obtain the concealed session key fk in Cfk.

$$Q_r \left[\begin{array}{c} Decrypt\left(Qk_S, \left\{U_{b,\tau}\right\}_{b \in X}, C_{fk}, C_\pi\right) = fk : \\ \pi\,(X) = True \end{array} \right] = 1 \tag{9}$$

where $\pi\,(X)$ denotes to Policy decision-making procedure π by X, the sign X represents to collection of all features.

Let us focus on the practical Dynamic Policy-based Access Control construction, based on the general "bilinear map group system" S = (Q, G1, G2, G_u, f(·, ·)) of priority o. Additionally, we add 2 procedures to our classification.

The cryptographic hash function $H : \{0,1\}^n \rightarrow G_2$ handles characteristics by mapping arbitrary strings to random elements in G_2.

When two binary strings have different lengths, the symbol $\oplus$ represents the XOR operation, which uses cycle filling to line up the highest bits of the shorter string.

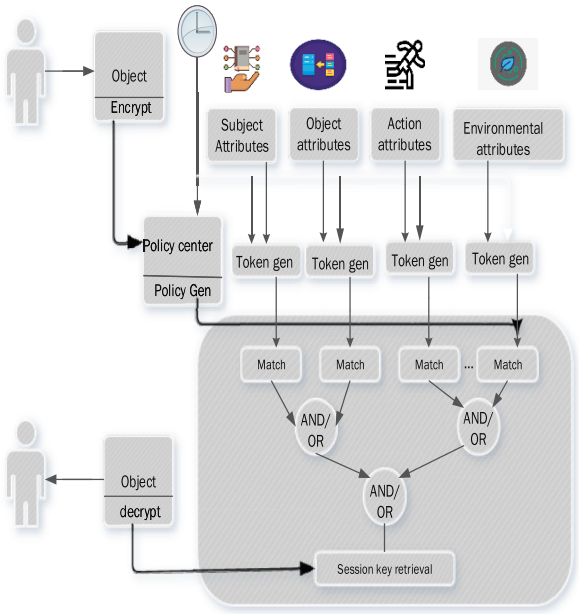

FIGURE 3. Diagram for dynamic policy-based access control method.

Figure 3 shows the Dynamic Policy-based Access Control method, we show the interactions between four attribute entities, the policy center, and the algorithms, including information transfers.

- The owner of an object p uses the public key Qk_Q of the policy center P to generate the cipher text CFk for a random session key e_k using the Encrypt algorithm. The resulting key fk is then utilized for useful Object encryption.
- The PolicyGen algorithm generates C_π for object p access requests by encapsulating policy Π in CFk using the private key TkQ of Q, the policy Π retrieved from the Policy administration point (PAP), the nonce τ, and the ciphertext CFk.
- For each policy Π attribute b, the corresponding entity uses the TokenGen algorithm to acquire Token $U_{b,\tau}$ using the secret key and nonce τ.
- The decrypt algorithm matches attributes and makes logical judgments between C_π and $\{U_{b,\tau}\}$ after acquiring all Tokens. The session key fk will be recovered if the last option is "permit," allowing the decryption of the object.

To establish synchronization, the policy decision unit (PDU) communicates the chosen nonce τ to the Policy generation unit (PGU) and all token generation units (TGU). Only cryptographic policy and attribute Tokens over this nonce succeed in verification. A clock in Figure 3 generates the nonce, which can be determined from the current time and hash value.

Detailed descriptions of the five algorithms in our Dynamic Policy-based Access Control architecture follow.

1) Initialization of the System: To design the system, managers select a prime order of "bilinear map group system S" and two random generators $g \in G_1$ and $h \in G_2$. These parameters will be shared across the system. Then, every entity uses the Setup algorithm to produce its public-private key pairs. Individual entities can to be more precise, have the following. As the private key, pick a random number.
   - The policy center Q generates a random $\alpha \in R\ Z * o$ as the private key $TkQ = (\alpha)$ and publishes the public key $Qk_Q = (g, h, g\alpha)$ as a public key certificate, such as X.509, PGP, or SKIP.
   - Each entity s in $S\backslash\{P\} = \{T, P, F, B\}$ selects a random $\beta s \in R\ Z * q$ as $TkT = (\beta s)$ and publishes the public key $Qkt = (g, h, g\beta t)$ as a certificate. For instance, the public key for the object attribute is $Qkp = (g, h, g\beta p)$.

   The fundamental characteristic of this phase is that each entity can generate its private key. The benefit of this feature is avoiding "key escrow," where a third party generates and manages the private key. The system requires complete trust in the escrowed party. Additionally, the private key can be utilized to find the authentication, preventing forgery in our system.
2) Encryption of Data Objects: The Encrypt algorithm is executable by the object owner. Any lawful user, known as a data user, can move encrypted files into the system to avoid unwanted access. The data holder must additionally declare the object properties of transferred data resources in the entity.
3) Cryptographic Policy Generation: Cryptographic Policy Generation: The AS requests the access policy from the PGU when a subject seeks access to an object, including RDU and PDU. The PGU constructs an access policy $\Pi := can\_access(T, P, B, F)$ by pulling rules from the policy repository in response to the request. This process can be done on the policy repository using the Dynamic Policy-based Access Control approach. The PGU executes the PolicyGen algorithm for acquired policy Π. The PolicyGen algorithm converts policy Π into cryptographic policy $C\Pi$, enhancing policy attack resistance. The sub cipher $c_1 = g^X$ from Cfk is used to generate p0 by wrapping it with a random encryption exponen $t \in RZ * p$. The generated cryptographic policy $C\Pi$ is only valid for the present cipher text Cfk and not for other cipher texts.
   The Linear secret sharing scheme (LSSS) is used to divide the encryption exponent into l values $\{\lambda k = Mk \cdot v\}_k^l$. The values will be hidden in $\{Qk\}k = 1$, where $Qk = (Qk1, Qk2)$ corresponds to the kth literal in policy Π.
   Note that the pair $(Qk1, Qk2)$ can contain information on the property $b\pi(k)$ of the kth literal belonging to entity s, including the nonce τ and public key $g\beta s$. Additionally, utilizing the policy center's private key α in Qk2 ensures that the PGU only builds the cryptographic policy.
4) Tokenization of Attributes in Real-Time: We utilize the TokenGen algorithm to define the production of

Algorithm 1 Policy Generation Algorithm

Input: the access policy, the nonce, the cipher text, and the private key TkQ of policy center Q;
Output: The encryption standard C of the policy;

1. Select $t \in_R \mathbb{Z}_Q^*$ and compute $Q_0 = c_1^{1/t} = \left(g^X\right)^{1/t}$;
2. Change the policy Π into (M, π)
3. Create a random vector. $w = (t, r_2, \ldots, r_n)^T \epsilon_R \mathbb{Z}_Q^n$;
4. Set l as the no of literals in Π;
5. for k = 1 to l do
6. Compute $\lambda_k = M_K.w$, here M_K is the kth row of M;
7. The best choice is a random $\gamma_k \epsilon_R \mathbb{Z}_Q^*$ and calculate

$$Q_k = \begin{cases} Q_{k1} = \left(g^{\beta s}.g^{\tau}\right)^{\gamma k} \\ Q_{k2} = \left(h^{\alpha}\right)^{\lambda k} \oplus f\left(g^{\beta s}, H\left(b_{\Pi}(k)\right)^{\gamma k}\right) \end{cases}$$

8. for the kth attribute $b\pi(k)$ fits to the object s;
9. end for
10. Create the cryptography policy for Π as

$$C_{\Pi} = \left(\Pi, (M, \pi), Q0, \{Q_k\}_{k=1}^{l}\right)$$

11. return $C\Pi$;

attribute Tokens. This token encapsulates the element a with the nonce τ using a secret key $sks = (\beta s)$ and a cryptographic hash function H. Encapsulation results in a specific signature of attribute a, known as attribute Token, under the τ model. Attribute Tokens are generated dynamically by individual authority upon PDU request. When requesting an attribute assignment, the authority first verifies that the nonce τ is genuine, then obtains the assignment from PIP via the attribute repository or runtime sensors. TokenGen is used to produce an attribute token, which serves as a signature for authenticating attribute assignment. After receiving Tokens, the PDU uses them to make cryptographic policy judgments before discarding them.

5) The decryption of Objects: The Decryption algorithm has 2 stages: Policy formulation and session key retrieval. To clarify, the PDU and RDU will implement these two steps, respectively. When making policy decisions, the PDU verifies the validity of the nonce τ. If the pattern passes, the algorithm uses the cryptography policy $c_{\pi} = \left(\Pi(, M, \pi), Q0\left\{, Q_k\right\}_{k=1}^{l}\right)$ from PGU, and all potential Tokens $\left\{u_{b,\tau}\right\}$ from TGU to generate an authorized set $U \subseteq \{1, 2, \ldots, l\}$ and an index $setI = \{i : \pi(i) \in U\}$. The policy literals $Qk = (Qk1, Qk2)$ for policy Π are evaluated with approved attributes Tokens $u_{b\pi(k)}, \tau$ for all $k \in I$. The secret m_k of literals Q_k can be obtained using it.

$$m_k = Q_{k2} \oplus f\left(Q_{k2}, u_{b\pi(k)}, \tau\right)$$

$$= Q_{k2} \oplus f\left(\left(g^{\beta s}.g^{\tau}\right)^{\gamma k}, H\left(b_{\pi(k)}\right)^{\frac{\beta s}{\beta s+\tau}}\right)$$

$$= Q_{k2} \oplus f\left(g^{\gamma k}, H\left(b_{\pi(k)}\right)^{\beta s}\right)$$

$$= \left(h^{\alpha}\right)^{\lambda k} \oplus f\left(g^{\beta s}, H\left(b_{\pi(k)}\right)\right)^{\gamma k} \oplus f\left(g^{\gamma k}, H\left(b_{\pi(k)}\right)^{\beta s}\right) = \left(h^{\alpha}\right)^{\lambda k} \quad (10)$$

The PDU calculates constants $\left\{\omega_i \epsilon \mathbb{Z}_Q\right\}^{i \varepsilon I}$ such that $\sum_{i \varepsilon I} \omega_i M_i = (1, 0, \ldots, 0)$ for M. For all values $\{m_i\}_{i \varepsilon I}$, the major secret m is found as follows:

$$m = \prod_{i \epsilon I} m_i^{\omega^i} = \left(h^{\alpha}\right)^{\sum_{I \epsilon I} \omega_i \lambda_i} = h^{\alpha u} \quad (11)$$

m may be permitted policy decision-making information. Then, the RDU receives m securely.

To retrieve the session key, the RDU uses the cipher text $Cfk = (c1, c2)$, the reverted significance m from the PDU, and the output value Q0 from the PGU. The computation of the session key ek follows:

$$fk = c_2 \oplus f(Q0, m) = c_2 \oplus f\left(\left(g^{\omega}\right)^{1/u}, h^{\alpha u}\right)$$
$$f_k \oplus f\left(g^{\alpha}, h\right)^{\omega} \oplus f\left(g^{\omega}, f^{\alpha}\right) \quad (12)$$

For valid fk, the RDU can decrypt the object $o = Decfk(Co)$. If yes, the decrypted object o can perform the subject's desired operations.

D. TRUST MANAGEMENT

The edge server contains two entities which are the trust manager and request manager. The trust manager computes and updates the trust value of the CSP and the request manager maintains the DU request. The trust manager computes the CSP trust based on sensitive service history, history of SLA, delay-sensitive requests handling, service correctness rate, number of requests received and handled, and resource utilization rate per request. For calculating trust value we proposed **Multi Behavior Analysis based Nomadic People Optimizer**.

Trust value is calculated based on the behavior of the CSP. Also, this algorithm balances the local and global search, and it addresses high dimensional space which handles the request at any type (low request, high request), and it has high convergence speed thus reducing trust calculation latency.

The trust value of each CSP is updated based on the feedback over the time interval and the user feedback is collected and stored in the blockchain with a hash format which cannot be compromised by the attackers thus leading to high security. It is one of the metaheuristic algorithms that depend on the behaviors of the nomadic people and their movement and search for life sources. It is designed based on the multi-swarm approach the optimal solution depending on the leader position. Nomads travel from one place to another for searching natural sources of food and water, based on this concept this algorithm provides the optimal solution for our CSP selection. The nomadic people optimizer algorithm includes five processes such as initial meeting, semi-circular

distribution, family searching, leadership transition, and periodic meeting.

The detailed explanation of these processes is defined as follows,

Step 1: Initial meeting

At first, the set of leaders $l = \{l1, l2, \ldots ln\}$ are initialized randomly using the formula

$$I = (Ub - Lb) \times Rand + LB \tag{13}$$

where Ub represents the upper bound Lb represents the lower bound and Rand represents the random value between 0 and 1. I represent the leader position.

Step 2: Semi-circular distributions

A set of families is distributed to the corresponding leader L which makes it possible to distribute the points randomly with a radius which are defined as follows,

$$Y = \left(R \times \sqrt{R_1}\right) \times \cos(\theta) + y \tag{14}$$

$$Z = \left(R \times \sqrt{R_2}\right) \times \cos(\theta) + z \tag{15}$$

where y and z represent the coordinates of the origin point. R1, and R2 represent the random coordinates of the point. The point of family is determined based on the location of the leader.

$$\overrightarrow{Y_c} = \overrightarrow{\sigma_c} \times \sqrt{R} \times cos(\theta) \tag{16}$$

where $\overrightarrow{Y_c}$ represents the point of a family, $\overrightarrow{\sigma_c}$ denotes the position of the leader for the similar clan (c), and R denotes a random digit in the range [0, 1].

Step 3: Families searching

In this stage, the families are searching for the best position far from their current position. In searching time, all of the families are being pushed in various directions. The levy flight formula generates random movements and directions as follows,

$$Y_{new} = Y_{old} + (a * (d - Y_{old}) \oplus Levy) \tag{17}$$

where Y_{new} and Y_{old} represent the new and old position of the current family respectively a represents the area and d represents the distance between the normal families. This equation is used to produce the random walk which depends on its current location.

$$a_c = \frac{\sum_{i=1}^{\phi} \sqrt{\left(\overrightarrow{\sigma_c} - \overrightarrow{y_i^{old}}\right)^2}}{\phi} \tag{18}$$

where ϕ denotes the number of families in each clan, $\overrightarrow{\sigma_c}$ and $\overrightarrow{y_i^{old}}$ denote the positions of the leader and the ordinary families, individually. The distance between y_c and σ_c gets closer when families are dispersed everywhere σ_c in a minor circle (i.e., semicircular distribution), this causes the search space to be explored in small steps.

While the great distances between all y_c and σ_c improves the capacity of the families to discover the pursuit space far from current rc. Thus, the value of ac has an excessive effect on the searching process.

Step 4: Change in Leadership

Check each clan to see whether there is a new family that is fitter than the clan's leader, and if so, the family assumes the leader and the other way around.

Step 5: Meetings are held regularly

Periodical meetings at this level are not identical to the first gathering for the distribution of power in the middle of nowhere. The leader searches for the best location for relocation. The leader updated their best location which performed by adding the variance among the strongest positions. The position of the normal leader is defined as follows,

$$Pos = \varepsilon \left(\frac{\sqrt{\sum_i^d \left(\alpha^e - \alpha_c^n\right)^2}}{\#d} \right) \tag{19}$$

where α^e represents the best leader position α^n represents the normal leader position d represents the dimension of the problems and ε represents the direction. *Pos* Represents the distance between a normal leader and the best leader.

Algorithm 2 Multi Behavior Analysis Based Nomadic People Optimizer

Step 1: Input: no of clans (c), no of families (f), no of iterations (ni)
Step 2: Output: Optimal solution
Step 3: Define an objective function
Step 4: Leader initialization ($L = \{L1, L2, \ldots .Ln\}$)
Step 5: Compute fitness value for every leader
Step 6: Repeat
Step 7: for L= 1 to i do
Step 8: Solution distribution (families around the leader)
Step 9: Compute the fitness value for every solution
Step 10: if best leader then swap best leader to original leader
Step 11: else: explore search space
Step 12: Determine the average distance between all families.
Step 13: Transfer the family to a new location.
Step 14: Determine the fitness value of each solution.
Step 15: choose the best clan
Step 16: If the clan is superior to the original, switch the original for the better.
Step 17: end if
Step 18: end for
Step 19: loop unit (iteration > i)
Step 20: return the best leader

E. OPTIMUM CSP SELECTION

Random selection of CSP increases latency while it processes multiple requests thus reducing overall performance, hence need to select optimal CSP for satisfying the request with minimum amount of time and high security. For that, we proposed an optimum CSP selection process. The optimum CSP is selected for providing services to the

DUs. An optimal CSP is selected based on trust available resources (CPU, memory, I/O, and bandwidth), execution time, extra resource utilization, SLA & QoS requirements, No. of requests under processing. For selecting optimal CSP we proposed **the Dynamic & Non Co-operative Game theory Approach** which selects optimal CSP from the list of CSP for providing services and Figure 4 shows the Optimum CSP Selection.

A Dynamic & non-cooperative game theory Approach is necessary to assess the objective function of a CSP with a specific strategy. It identifies population tactics. Many games have a restricted number of winning tactics. If we identify each participant with their chosen strategy, population growth does not impact the complexity of the challenge.

$$y_s^* = y_s \left[f_i \left(x_s \right) - \pi \left(y \right) \right], \pi \left(y \right) = \sum_{i=1}^{n} x_i f_i \left(y \right) \tag{20}$$

We use y_s to represent the ratio of providers utilizing a specific method to the total number of participants. $f_i \left(y_s \right)$ is the average profit of a CSP using strategy s and $\pi \left(y \right)$ is the auction servers' average profit.

$$y_{k,i}^* = x_{k,i}. \left(\bar{\pi}_k \left(x_{k,i} \right) - \bar{\pi}_i \right), \quad p_i, p_k \epsilon P \tag{21}$$

Let $x_{k,}^{I*}$ represent the valuation of a service provider p_j for the auction held by the provider p_i in equilibrium. In equilibrium, we attempt to obtain the best valuation vector $x_i^* = \left(x_{1,i}^*, x_{2,i}^*, i, \ldots, x_{N,i}^* \right)$ for the service provider auction p_i.

At the equilibrium point, the difference between VM supply and demand must be positive for each CSP. The replicator equation must be zero to find this position and this condition is enough for strategy stability.

$$\left(\bar{\pi}_k \left(x_{k,i} \right) - \bar{\pi}_i \right) = 0, p_i, p_k \epsilon P \tag{22}$$

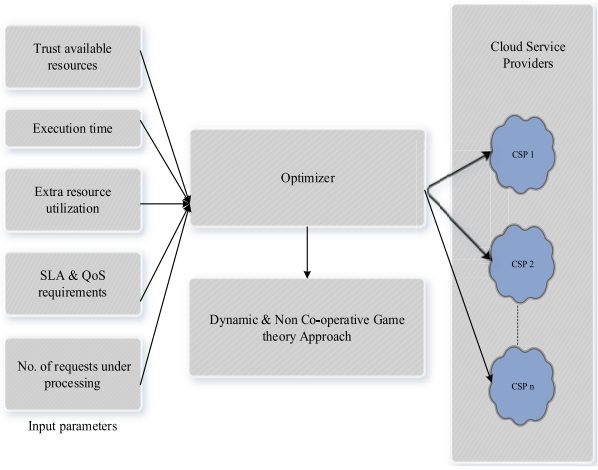

FIGURE 4. Diagram for optimum CSP selection.

V. EXPERIMENTAL RESULTS

This section demonstrates experimental research on the overall security of the cloud-IoT environment, demonstrating resistance to various internal and external threats. This section is divided into two subsections, the comparison analysis and the research summary.

A. COMPARATIVE ANALYSIS

Our method is assessed by contrasting it with the existing methods in terms of No of Requests vs. CSP Security Loss, No of Requests vs. # of Insecure CSPs, No of CSPs vs. SLA Violation Rate, No of CSPs vs. Latency, No of Requests vs. Latency, No of Requests vs. Response time, No of Requests vs. Throughput, No of CSPs vs. Throughput, No of Requests vs. Request Success Ratio, No of CSPs vs. Response Time, No of Feedbacks vs. Request Success Ratio. These existing methods include Service Level Agreement (SLA-AS), Policy customized trusted cloud service (PC-TCS), Elliptic curve cryptography-advanced encryption standard (ECC-AES), Genetic algorithm(GA), Sensitive aware deep elliptic curve cryptography(SAD-EC), Rough set (RS), Gaussian distribution –Technique for order of preference by similarity to ideal solution(G-TOPSIS).

1) NUMBER OF REQUESTS VS. CSP SECURITY LOSS

The following equation can be used to depict the link between the quantity of requests (R) and the security loss suffered by cloud providers (S):

$$S = k * R \tag{23}$$

where:

S - Security loss incurred.

R - Number of requests made.

k - Constant that represents the security risk associated with each request.

FIGURE 5. Number of requests vs. CSP security loss.

Figure 5 depicts the proposed and existing methodologies for comparing the number of requests vs. CSP security loss. At first, the proposed approach had zero precision. In a 400 request, PC-TCS has 38% security and SLA-AS has 37% security; our suggested method will increase it to 39% security. The SLA-AS has 57% security while the PC-TCS has 58% security with 600 requests, but our proposed method would reduce it to 59% security. When the SLA-AS receives 1000 requests, PC-TCS has 80% security, however, our proposed upgrade would increase this to 100% security every 1000 requests.

2) NUMBER OF REQUESTS VS. # OF INSECURE CSPs

The following equation can be used to depict the relationship between the quantity of requests (R) and the quantity of unreliable cloud service providers (CSPs):

$$I = f(R) \tag{24}$$

where:

I = quantity of unsafe CSPs.

R = total number of requests.

The function f(R) = illustrates the relationship between the number of requests and the number of insecure CSPs.

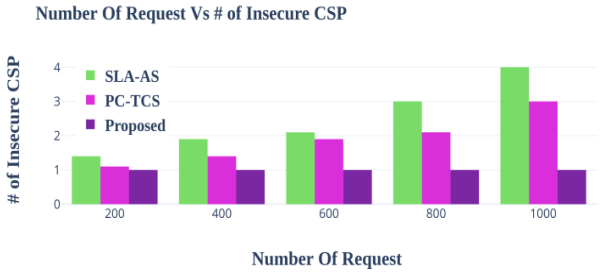

FIGURE 6. Number of requests vs. # of insecure CSP.

The suggested and existing approaches to compare the number of insecure CSPs are shown in Figure 6. The proposed method has zero precision in the beginning. While PC-TCS has 1.4 insecure CSP and SLA-AS has 1.9 insecure CSP in a 400 request, our suggested method will increase it to 1 insecure CSP. With 600 requests, the SLA-AS has 2.1 insecure CSP and the PC-TCS has 1.9 insecure CSP, but our proposed solution would reduce to 1 insecure CSP. When the SLA-AS receives 1000 requests, there are 3 insecure CSPs and 3 insecure CSPs for PC-TCS, but our suggested method would reduce to 1 insecure CSP per 1000 requests. This makes it quite evident that the proposed work contains less insecure CSP than the current work does.

3) NUMBER OF CSPs VS. SLA VIOLATION RATE

The following equation can be used to represent the link between the number of CSPs (N) and the rate of SLA violations (V):

$$V = a * N \wedge b \tag{25}$$

where:

The SLA violation rate is V.

The number of CSPs is N.

With "a" standing for the baseline violation rate per CSP and "b" for the exponent that describes how SLA violations evolve with the number of CSPs, "a" and "b" are constants that dictate the nature of the connection.

The suggested and current approaches to assess the number of CSP vs. SLA Violation Rates are shown in Figure 7. The suggested method begins with a zero violation rate. In 4 CSP, the SLA-AS has a 4.1 violation rate, and in 4 violation rate, the PC-TCS has a 4 violation rate. However, the suggested approach reduces these numbers to 1 violation rate. While

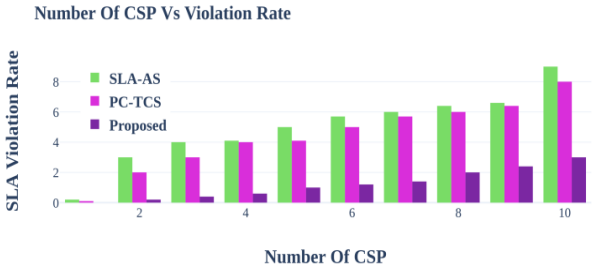

FIGURE 7. Number of CSP vs. SLA violation rate.

the SLA-AS has a 5.7 violation rate and the PC-TCS has a 5 violation rate while the SLA-AS is in CSP 6, our proposed work would increase these numbers to 1.2 violation rate in CSP 6. The PC-TCS has an 8 violation rate and a 9 violation rate when the SLA-AS is in 10 CSP, but our proposed improvement would increase these numbers to 3 violations every 10 CSP.

4) NUMBER OF CSPs VS. LATENCY

The following equation can be used to visualize the link between latency (L) and the quantity of cloud service providers (N):

$$L = c * log(N) \tag{26}$$

where:

L stands for latency.

The number of CSPs is N.

A constant called "c" affects how CSPs and latency are related.

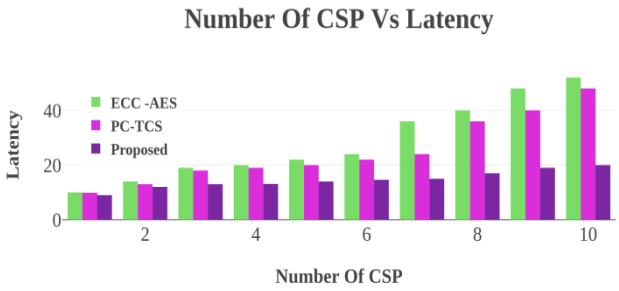

FIGURE 8. Number of CSP vs. latency.

The suggested and current approaches to compare the number of CSP with latency (ms) are shown in Figure 8. The proposed method has no initial delay. While PC-TCS is 19 ms and ECC-AES has 20 ms in 4 CSP, our proposed effort would increase that to 13.1 ms. While PC-TCS is 36 ms and ECC-AES has 40 ms in 8 CSP, our proposed effort would increase that to 17 ms. While PC-TCS has 48 ms and ECC-AES has 52 ms when in 10 CSP, our proposed effort would increase that to 20 ms. When compared to previous work, our solution reduces the number of CSPs by 20ms.

5) NO OF REQUESTS VS. LATENCY

The following equation can be used to visualize the relationship between latency (L) and the quantity

of requests (R):

$$L = a * log(R) + b \tag{27}$$

where:

L stands for latency.

The total number of requests is R.

Constants ''a'' and ''b'' establish the relationship. With the logarithm of the request count, ''a'' denotes the rate at which latency grows, and ''b'' is an offset term.

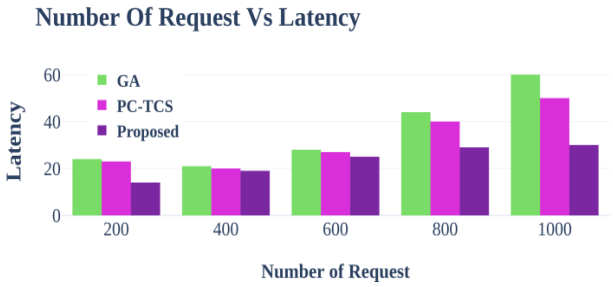

FIGURE 9. Number of requests vs. latency.

The ways to assess the number of requests vs. latency (ms) are suggested and shown in Figure 9. The proposed method begins with zero latency. In a 200 request, the GA takes 24 ms, and the PC-TCS takes 23 ms, but our proposed solution reduces that to 14 ms. When the GA receives 600 requests, it takes 28 ms, and PC-TCS takes 27 ms, however, our suggested solution reduces this to 25 ms. In 1000 requests, the GA takes 60 ms and the PC-TCS takes 50 ms, however, our suggested improvement would reduce this to 30 ms in 1000 CSP. By choosing the best CSP and figuring out the CSP's trust value, our suggested method achieves minimal response time in comparison to previous studies.

6) NO. OF REQUESTS VS. RESPONSE TIME

A straightforward linear equation can be used to model the relationship between the quantity of requests (R) and the response time (T):

$$T = a * R + b \tag{28}$$

where:

The response time is T.

The number of requests is R.

The slope of the line, denoted by the letter ''a,'' shows how response times grow longer as more requests are made. The base response time, or ''b,'' is the y-intercept and is represented by the absence of queries.

This equation offers insights into system performance and scalability and helps you forecast response time based on the number of requests.

The ways to compare the number of requests and the response time are shown in Figure 10. The proposed method has no initial delay. SAD-ECC and PC-TCS both have response times of 5.2 and 5.3 for 200 requests, respectively, but our suggested solution would increase to 5 for

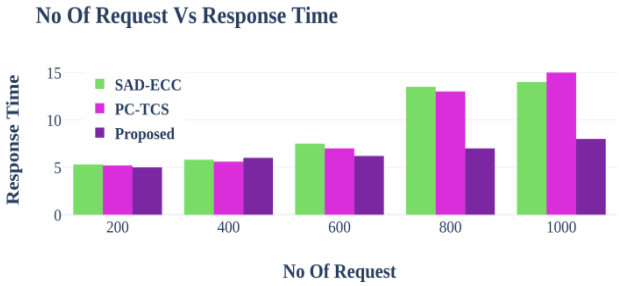

FIGURE 10. Number of requests vs. response time.

200 requests. The SAD-ECC has a response time of 7.5 in 600 requests, while the PC-TCS has a response time of 7, but our suggested improvement would reduce that to 6.2 in 600 requests. Our suggested improvement would increase response times to 8 in 1000 CSP from the GA's 14 and PC-TCS's 15 in 1000 requests, respectively. By choosing the best CSP and determining the CSP's trust value, our suggested method reduces response time in comparison to previous work.

7) NUMBER OF CSPs VS. RESPONSE TIME

A straightforward equation can be used to characterize the link between the quantity of cloud service providers (CSPs) and response time (T):

$$T = k/CSPs. \tag{29}$$

where:

The response time is T. The quantity of Cloud Service Providers is known as CSPs. The constant 'k' determines how the response time varies with the number of CSPs. This equation shows that when the number of CSPs rises, the response time often falls in an inverse relationship to the CSP count, indicating improved parallelism and potential performance advantages.

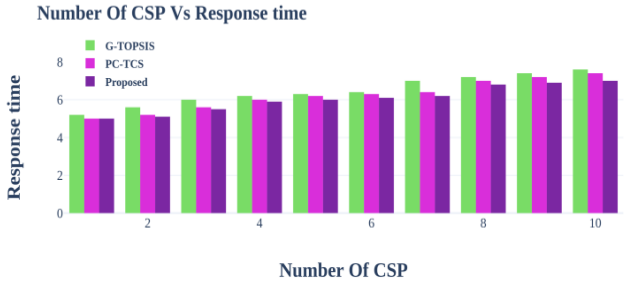

FIGURE 11. Number of CSPs vs. response time.

The recommended and current approaches to compare the number of CSPs with the response time are shown in Figure 11. The proposed method has no initial delay. The G-TOPSIS has a response time of 5.6 in 2 CSP, while the PC-TCS has a response time of 5.2, but our suggested method would increase to 5.1 in 2 CSP. The G-TOPSIS

has a 6.4 response time in 6 CSP and the PC-TCS has a 6.3 response time, but our suggested effort will increase them to 6.1 in 6 CSP. The GA has a response time of 7.6 in 10 CSP, while PC-TCS has a response time of 7.4, but our suggested improvement would increase that to 7 in 10 CSP.

8) NUMBER OF REQUESTS VS. THROUGHPUT

A simple equation can be used to express the relationship between throughput (TP) and the quantity of requests (R):

$$TP = R/T \tag{30}$$

where:

The throughput, or TP, measures how many requests are handled per unit of time.

The number of requests is R.

The processing of the requests takes time (T)

This equation shows that throughput is inversely proportional to processing time and directly proportional to the number of requests. It measures the effectiveness with which a system can manage incoming requests.

FIGURE 12. Number of requests vs. throughput.

Figure 12 displays the proposed and existing approaches for evaluating the number of requests vs. throughput. The proposed method begins with a precision of 0. ECC-AES has 70 kbps in 400 requests and PC-TCS has 100 kbps, however our proposed work expands to 150 kbps in 400 requests. SLA-AS has 100kbps in 600 requests and PC-TCS has 150kbps, however, our suggested work expands to 230kbps in 600 requests. SLA-AS has 170kbs in 1000 requests and PC-TCS has 200kbps, however, our planned work will increase to 300kbps in 1000 requests.

9) NO OF CSPs VS. THROUGHPUT

A straightforward equation can be used to illustrate the connection between the quantity of Cloud Service Providers (CSPs) and throughput (TP):

$$TP = CSPs * a \tag{31}$$

where:

Throughput (TP) is the measure of how much data or how many requests are processed in a given amount of time.

The quantity of Cloud Service Providers is known as CSPs.

The constant "a" stands for the effect on the throughput of adding or removing CSPs.

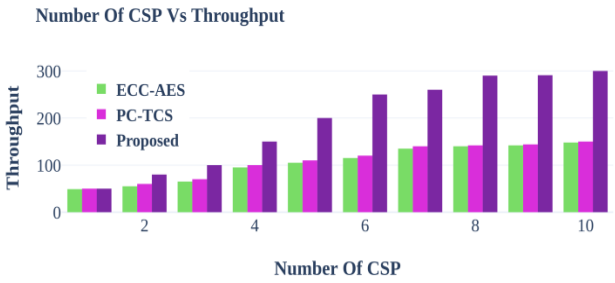

FIGURE 13. No of CSPs vs. throughput.

The suggested and existing approaches to compare the number of CSPs with throughput (kbps) are shown in Figure 13. The proposed method has no initial delay. While PC-TCS is 60 kbps and ECC-AES has 55 kbps in 2 CSP, our proposed effort would increase that to 80 kbps. While PC-TCS is 250 kbps and ECC-AES has 115 kbps in 6 CSP, our proposed effort would increase that to 250 kbps. The GA has 148 kbps in 10 CSP, and PC-TCS has 150 kbps, but our suggested work would increase that to 300 kbps in 10 CSP.

10) NO OF REQUESTS VS. REQUEST SUCCESS RATIO

The following equation can be used to represent the relationship between the quantity of requests (R) and the request success ratio (RSR):

$$RSR = 1 - (F/R) \tag{32}$$

where:

The RSR, or request success ratio, is a measure of the percentage of requests that are successful.

The overall number of requests is R.

F stands for the number of unsuccessful requests.

RSR is expressed in this equation as the failure rate's complement, giving information on the general effectiveness of requests in a system.

FIGURE 14. No of requests vs. request success ratio.

The suggested and current approaches to assess the No of Requests vs. Request Success Ratio are shown in Figure 14. The proposed method has no initial delay. Our proposed work is extending to a 0.2 success ratio in 2 CSP while the RS

TABLE 2. Numerical outcomes of suggested and existing methods.

SUCCESS METRICS	SLA-AS	ECC-AES	GA	SAD-ECC	RS	G-TOPSIS	PC-TCS	PROPOSED
CSP Security Loss	70						80	100
# od Insecure CSP	4						3	1
SLA Violation Rate	9						8	3
CSP vs. Latency		52					48	20
Request vs. Latency			60				50	30
Request vs. Response Time				16			15	8
CSP vs. Response Time						7.6	7.4	7
Request vs. Throughput		170					200	300
CSP vs. Throughput		148					150	300
Request vs. Request Success Ratio					0.4		0.5	1
Feedback vs. Request Success Ratio					0.25		0.2	1

is in 200 requests, while PC-TCS is at 0.12 success ratio. Our suggested work will increase the RS's success rate in 600 requests from its current 0.2 to 0.6. PC-TCS's success rate in 600 requests is 0.4. Our suggested effort is increasing to a success ratio of 1 in 1000 requests from the GA's 0.4 in 1000 requests and the PC-TCS's 0.5 in 1000 requests.

11) NO OF FEEDBACK VS. REQUEST SUCCESS RATIO

A straightforward equation can be used to model the link between the quantity of feedback (F) and the request success ratio (RSR):

$$RSR = 1 - (F/N) \tag{33}$$

where:

Request Success Ratio, or RSR, is a measure of the percentage of requests that are successful.

F is the number of failure-related feedbacks.

The overall number of requests is N.

This equation shows that RSR is inversely proportional to the quantity of feedback, with an increase in feedback (failures) corresponding to a drop in the success ratio, giving users information about the system's dependability and performance.

Figure 15 displays the proposed and existing approaches for calculating the number of feedbacks vs. request success ratio. The proposed method begins with a precision of 0. The RS has a 0.175 success ratio in 20 feedbacks and the PC-TCS has a 0.15 success ratio, however, our proposed work expands to a 0.5 success ratio in 400 requests. The

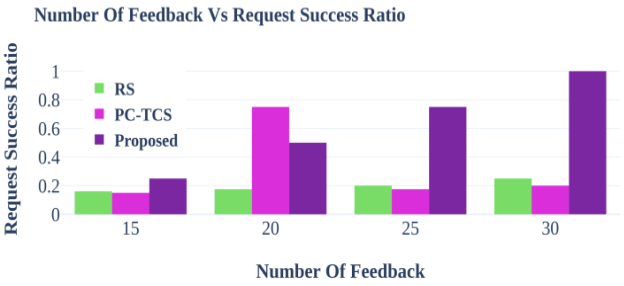

FIGURE 15. No of feedback vs. request success ratio.

RS has a 0.25 success ratio in 30 feedbacks, and the PC-TCS has a 0.2 success ratio, but our proposed work expands to a 1 success ratio in 30 feedbacks. In comparison to existing work, our proposed work achieves a high success percentage by conducting authentication, request scheduling, access control, trust calculation, and optimal CSP selection. Users' feedback is recorded and kept in the blockchain, which provides excellent security and a high success rate.

B. RESEARCH SUMMARY

To give cloud services to legitimate DUs and customize cloud security, biometric qualities along with PUF from DU's device can be employed. Managing diverse service request types (sensitive and non-sensitive) in a cloud environment presents challenges. Some of the security issues can be solved by using blockchain technology and trust management.

To achieve high performance through high-security levels, blockchain technology is used to gather policy, trust, and service history information from Edge and CSP. A smart and secure cloud service discovery framework takes into account both SLA and QoS requirements. Finally, Table 2 illustrates the Numerical outcomes of suggested and existing outcomes. Figure 5 – Figure 15 represents the success metrics of the suggested approaches.

VI. CONCLUSION

Data user authentication is carried out in the first step to identify and remove allowed users. To strengthen security, we developed the chaotic map-based camellia encryption method (CMCE). The service request is gathered by the gateway in the second phase, and it is scheduled using Johnson's rule-based stochastic gradient descent technique while taking priority, throughput, and delay into account. It divides the request into sensitive and non-sensitive services for scheduling. The third phase involves performing policy verification using a dynamic policy-based access control approach that restricts access to just sensitive requests. To boost security, we compute the trust value for the CSP in the fourth phase. To do this, we put out the Multi Behavior Analysis based Nomadic People Optimizer method, which is based on CSP behavior. Additionally, every CSP has its trust value modified based on user feedback collected over time. Finally, the best CSP is chosen to offer the data user the best service. To do that, we put forth a dynamic and non-cooperative game theory approach that chooses the best CSP from a list of CSPs. Finally, the CloudSim environment is used to run the simulation and evaluate it. The assessment of our method is carried out by comparing the existing approaches and the proposed methods and the Numerical study shows that our solution surpasses all existing approaches in all parameters.

REFERENCES

[1] D. K. Murala, S. K. Panda, and S. K. Sahoo, "Securing electronic health record system in cloud environment using blockchain technology," in *Recent Advances in Blockchain Technology: Real-World Application*. Cham, Switzerland: Springer, 2023, pp. 89–116.

[2] W. Li, J. Wu, J. Cao, N. Chen, Q. Zhang, and R. Buyya, "Blockchain-based trust management in cloud computing systems: A taxonomy, review and future directions," *J. Cloud Comput.*, vol. 10, no. 1, pp. 1–34, Jun. 2021.

[3] J. Huang, W. Susilo, F. Guo, G. Wu, Z. Zhao, and Q. Huang, "An anonymous authentication system for pay-as-you-go cloud computing," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1280–1291, Mar. 2022.

[4] W. Tian, Q. Liu, R. Li, Z. Xu, Y. Zhang, and Y. Huang, "A blockchain-based secure searching strategy for metadata in mobile edge computing," *IEEE Internet Things J.*, 2023.

[5] A. G. Gad, E. H. Houssein, M. Zhou, P. N. Suganthan, and Y. M. Wazery, "Damping-assisted evolutionary swarm intelligence for industrial IoT task scheduling in cloud computing," *IEEE Internet Things J.*, 2023.

[6] C. Huang, W. Chen, L. Yuan, Y. Ding, S. Jian, Y. Tan, H. Chen, and D. Chen, "Toward security as a service: A trusted cloud service architecture with policy customization," *J. Parallel Distrib. Comput.*, vol. 149, pp. 76–88, Mar. 2021.

[7] S. Algarni, F. Eassa, K. Almarhabi, A. Almalaise, E. Albassam, K. Alsubhi, and M. Yamin, "Blockchain-based secured access control in an IoT system," *Appl. Sci.*, vol. 11, no. 4, p. 1772, Feb. 2021.

[8] C. Mao, R. Lin, D. Towey, W. Wang, J. Chen, and Q. He, "Trustworthiness prediction of cloud services based on selective neural network ensemble learning," *Expert Syst. Appl.*, vol. 168, Apr. 2021, Art. no. 114390.

[9] A. S. Abohamama, A. El-Ghamry, and E. Hamouda, "Real-time task scheduling algorithm for IoT-based applications in the cloud–fog environment," *J. Netw. Syst. Manage.*, vol. 30, no. 4, p. 54, Oct. 2022.

[10] B. Prabhu Kavin, S. Ganapathy, U. Kanimozhi, and A. Kannan, "An enhanced security framework for secured data storage and communications in cloud using ECC, access control and LDSA," *Wireless Pers. Commun.*, vol. 115, no. 2, pp. 1107–1135, Nov. 2020.

[11] B. Alouffi, M. Hasnain, A. Alharbi, W. Alosaimi, H. Alyami, and M. Ayaz, "A systematic literature review on cloud computing security: Threats and mitigation strategies," *IEEE Access*, vol. 9, pp. 57792–57807, 2021.

[12] G. Ragu and S. Ramamoorthy, "A blockchain-based cloud forensics architecture for privacy leakage prediction with cloud," *Healthcare Anal.*, vol. 4, Dec. 2023, Art. no. 100220.

[13] M. A. Darwish, E. Yafi, M. A. Al Ghamdi, and A. Almasri, "Decentralizing privacy implementation at cloud storage using blockchain-based hybrid algorithm," *Arabian J. Sci. Eng.*, vol. 45, no. 4, pp. 3369–3378, Apr. 2020.

[14] F. Nadeem, "A unified framework for user-preferred multi-level ranking of cloud computing services based on usability and quality of service evaluation," *IEEE Access*, vol. 8, pp. 180054–180066, 2020.

[15] P. Sun, "A trust game model of service cooperation in cloud computing," *J. Netw. Comput. Appl.*, vol. 173, Jan. 2021, Art. no. 102864.

[16] M. A. Elmagzoub, D. Syed, A. Shaikh, N. Islam, A. Alghamdi, and S. Rizwan, "A survey of swarm intelligence based load balancing techniques in cloud computing environment," *Electronics*, vol. 10, no. 21, p. 2718, Nov. 2021.

[17] S. Jayaprakash, M. D. Nagarajan, R. P. D. Prado, S. Subramanian, and P. B. Divakarachari, "A systematic review of energy management strategies for resource allocation in the cloud: Clustering, optimization and machine learning," *Energies*, vol. 14, no. 17, p. 5322, Aug. 2021.

[18] Z. Shi, H. Zhou, C. de Laat, and Z. Zhao, "A Bayesian game-enhanced auction model for federated cloud services using blockchain," *Future Gener. Comput. Syst.*, vol. 136, pp. 49–66, Nov. 2022.

[19] S. A. Murad, A. J. M. Muzahid, Z. R. M. Azmi, M. I. Hoque, and M. Kowsher, "A review on job scheduling technique in cloud computing and priority rule based intelligent framework," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 6, pp. 2309–2331, Jun. 2022.

[20] R. Maeser, "Analyzing CSP trustworthiness and predicting cloud service performance," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 73–85, 2020.

[21] T. Dbouk, A. Mourad, H. Otrok, H. Tout, and C. Talhi, "A novel ad-hoc mobile edge cloud offering security services through intelligent resource-aware offloading," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 4, pp. 1665–1680, Dec. 2019.

[22] H. Hammami, S. B. Yahia, and M. S. Obaidat, "A lightweight anonymous authentication scheme for secure cloud computing services," *J. Supercomput.*, vol. 77, no. 2, pp. 1693–1713, Feb. 2021.

[23] A. E. Youssef, "An integrated MCDM approach for cloud service selection based on TOPSIS and BWM," *IEEE Access*, vol. 8, pp. 71851–71865, 2020.

[24] A. Hussain, J. Chun, and M. Khan, "A novel customer-centric methodology for optimal service selection (MOSS) in a cloud environment," *Future Gener. Comput. Syst.*, vol. 105, pp. 562–580, Apr. 2020.

[25] S. Gu, X. Luo, D. Guo, B. Ren, G. Tang, J. Xie, and Y. Sun, "Joint chain-based service provisioning and request scheduling for blockchain-powered edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2135–2149, Feb. 2021.

[26] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid fog–cloud-computing," *Future Gener. Comput. Syst.*, vol. 111, pp. 539–551, Oct. 2020.

[27] S. S. Chauhan, E. S. Pilli, and R. C. Joshi, "BSS: A brokering model for service selection using integrated weighting approach in cloud environment," *J. Cloud Comput.*, vol. 10, no. 1, pp. 1–14, Dec. 2021.

[28] W. Ran and H. Liu, "Cloud service selection based on QoS-aware logistics," *Soft Comput.*, vol. 24, no. 6, pp. 4323–4332, Mar. 2020.

[29] S. Rizvi, J. Mitchell, A. Razaque, M. R. Rizvi, and I. Williams, "A fuzzy inference system (FIS) to evaluate the security readiness of cloud service providers," *J. Cloud Comput.*, vol. 9, no. 1, pp. 1–17, Dec. 2020.

[30] R. K. Tiwari and R. Kumar, "G-TOPSIS: A cloud service selection framework using Gaussian TOPSIS for rank reversal problem," *J. Supercomput.*, vol. 77, no. 1, pp. 523–562, Jan. 2021.

[31] P. S. Challagidad and M. N. Birje, "Multi-dimensional dynamic trust evaluation scheme for cloud environment," *Comput. Secur.*, vol. 91, Apr. 2020, Art. no. 101722.
[32] G. Aghaee Ghazvini, M. Mohsenzadeh, R. Nasiri, and A. M. Rahmani, "A new multi-level trust management framework (MLTM) for solving the invalidity and sparse problems of user feedback ratings in cloud environments," *J. Supercomput.*, vol. 77, no. 3, pp. 2326–2354, Mar. 2021.
[33] Z. Chen and Z. Yu, "Intelligent offloading in blockchain-based mobile crowdsensing using deep reinforcement learning," *IEEE Commun. Mag.*, vol. 61, no. 6, pp. 118–123, Jun. 2023.
[34] J. Xiao, H. Huang, C. Wu, Q. Chen, and Z. Huang, "A collaborative auditing scheme with dynamic data updates based on blockchain," *Connection Sci.*, vol. 35, no. 1, Dec. 2023, Art. no. 221383.
[35] M. Sharma, J. Singh, A. Gupta, S. Tanwar, G. Sharma, and I. E. Davidson, "Intercloud resource discovery using blockchain," *IEEE Access*, vol. 9, pp. 161224–161247, 2021.
[36] A. Ali, M. M. Iqbal, H. Jamil, F. Qayyum, S. Jabbar, O. Cheikhrouhou, M. Baz, and F. Jamil, "An efficient dynamic-decision based task scheduler for task offloading optimization and energy management in mobile cloud computing," *Sensors*, vol. 21, no. 13, p. 4527, Jul. 2021.
[37] M. Uddin, A. Khalique, A. K. Jumani, S. S. Ullah, and S. Hussain, "Next-generation blockchain-enabled virtualized cloud security solutions: Review and open challenges," *Electronics*, vol. 10, no. 20, p. 2493, Oct. 2021.
[38] Y. Khan and S. Verma, "An intelligent blockchain and software-defined networking-based evidence collection architecture for cloud environment," *Sci. Program.*, vol. 2021, pp. 1–19, Sep. 2021.
[39] A. Pinheiro, E. D. Canedo, R. T. De Sousa, and R. De Oliveira Albuquerque, "Monitoring file integrity using blockchain and smart contracts," *IEEE Access*, vol. 8, pp. 198548–198579, 2020.
[40] X. Zhou, W. Liang, K. Yan, W. Li, K. I. Wang, J. Ma, and Q. Jin, "Edge-enabled two-stage scheduling based on deep reinforcement learning for Internet of Everything," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3295–3304, Feb. 2023.
[41] R. R. Kumar, A. Tomar, M. Shameem, and M. N. Alam, "OPTCLOUD: An optimal cloud service selection framework using QoS correlation lens," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–16, May 2022.
[42] Y. Choi, Y. Kim, and M. Rhu, "Lazy batching: An SLA-aware batching system for cloud machine learning inference," in *Proc. IEEE Int. Symp. High-Perform. Comput. Archit. (HPCA)*, Feb. 2021, pp. 493–506.

MUZAMMIL AHMAD KHAN received the B.S. and M.S. degrees in computer engineering from the Sir Syed University of Engineering and Technology, Karachi, Pakistan. He is currently pursuing the Ph.D. degree with the NED University of Engineering and Technology, Karachi. He is an Assistant Professor with the Department of Computer Engineering, Sir Syed University of Engineering and Technology. His research interests include cloud computing, the IoT, cyber security, networks, artificial intelligence, data science, e-commerce, and project management.

SHARIQ MAHMOOD KHAN received the bachelor's and master's degrees in computer science from the NED University of Engineering and Technology, Pakistan, in 2005 and 2007, respectively, and the Ph.D. degree from Brunel University London, U.K., in 2015. He is currently a Professor with the Department of Computer Science and Information Technology, NED University of Engineering and Technology. His research interests include network security, vehicular ad-hoc networks, the IoT security, and wireless sensor networks.

SIVA KUMAR SUBRAMANIAM received the bachelor's degree in electronic engineering from Kolej Universiti Teknikal Kebangsaan Malaysia, in 2006, the M.Sc. degree in electronics from Universiti Teknikal Malaysia Melaka, Malaysia, in 2009, and the Ph.D. degree from Brunel University London, U.K., in 2017. He is currently a Senior Lecturer with the Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka. His research interests include the IoT, network security, and wireless sensor networks.