A Secure and Efficient Framework for Crowdsourcing-Based Mobile Cloud Computing: Enhancing Task Management, QoS, and Real-Time Risk Monitoring

ABDULLAH A. BUKSH[1], MANSOUR H ASSAF[1], BIBHYA SHARMA[1], SATYANAND SINGH[2]
[1]School of Information Technology, Engineering Mathematics and Physics,
University of the South Pacific,
Suva,
FIJI

[2]Department of Electrical and Electronic Engineering,
Fiji National University,
Suva,
FIJI

Abstract: - This work aims to develop a secure and efficient process in crowdsourcing-based systems, such as mobile cloud computing. This project is developed to address the rising concern about security vulnerability and the degradation in such environments. The primary objective included enhancing the Quality of Service (QoS) metrics. This process includes developing a secured task management distribution and acceptance system, this additionally involves real-time monitoring of potential risks associated with task distribution and users. The system would be integrating advanced encryption mechanism protocols to protect sensitive task information which also provides real-time notifications in the administrators' dashboard, for early detection of potential risk identification and allows admins to action on such risk for mitigation. Additionally, the platforms would be offering the QoS metrics on the admin dashboard for Admin users to monitor, this would include task completion time, acceptance delay, and the count of failed task counts. This metric's purpose is to allow admins to maintain a high performance on user satisfaction. The report will highlight the project structure, design, security, implementation, and how the system was developed according to the proposal while addressing the challenges that are associated with crowdsourcing-based mobile cloud computing. The paper covers the system's architecture, security features, and real-world implementation, along with the technical challenges faced and how they were resolved to ensure a secure and reliable system.

Key-Words: - Mobile Cloud Computing, Crowdsourcing, Security, Task Management, Quality of Service.

Received: May 26, 2024. Revised: November 19, 2024. Accepted: March 11, 2025. Published: June 30, 2025.

1 Introduction

For some years, mobile cloud computing has been developing and improving on its security and efficiency over the years, this has eventually led the industry to evolve in the execution of large-scale distributed tasks by fully optimizing the capabilities of the mobile devices and cloud infrastructure. The crowdsourcing model on its own has gained widespread adoption in the ability to gather human inputs and machine ecosystem, which has led such a model to achieve its complex objective. While this remains a benefit, some hidden/identified challenges and issues are related to security which are present in this model, together with task management and the maintenance of Quality of Service.

One of the main concerns in such an area would be the crowdsourcing-based systems that are associated with risk and potential data misinformation if a malicious system or user vulnerabilities could end up in task distribution failure and exploitation. To ensure that the task is presented to users securely and efficiently all while maintaining the Quality of Service. This paper aims to tackle such challenges and issues by developing a crowdsourcing system

application, which primarily focuses on securing the task management process, with real-time risk identification and Quality of Service monitoring.

This paper proposes the implementation of encryption and password hashing mechanisms to safeguard sensitive information from unauthorized access and mitigate the risk of data loss.

It was achieved by monitoring the user behavior on the admin panel and notifying admins of unusual user behavior in real-time. Implementation of Quality of Service Metrics through the admin panel, which provides an overview of the critical event marking such as completion time, the acceptance delay, and failure count rates. The Task Management feature allows the admin to create multiple tasks and assign them to a group; users can accept or reject the tasks. Depending, on how many users are active at a time, the administrators can control which user can do their task, providing an additional level of security.

2 Background

In recent years, there have been multiple developments in crowdsourcing and mobile computing, this feature has produced many opportunities to improve resource utilization and the task distribution and management process. Although these developments were released to the public, there were still challenges present in security and the metrics in Quality of Service. To address such, issues we have developed a security protocol while maintaining a high quality of service.

This paper mentions a review that highlights on the challenges which are related to the Quality of Services in the cloud system. They can emphasize the crowdsourcing-based systems, which is essential as its able to monitor on task completion task, count the failed tasks as well. This research also aligns with our developed project, the QoS dashboard can monitor the average time completion, the task acceptance delay, and the count of failed tasks, [1].

The authors discuss the evolution of crowdsourcing on the Web, highlighting its growth as a key method for problem-solving and data collection. They explore different systems that use the collective intelligence of users to complete tasks, from content creation to data analysis.

The article concludes that crowdsourcing has greatly changed how tasks are carried out online, offering scalable and cost-effective solutions. It stresses the need to design systems that make the most of user contributions while addressing the challenges that come with it, [2].

2.1 Crowdsourcing and Mobile Cloud Computing

Crowdsourcing has developed a powerful model that enables users to solve complex problems with collective intelligence and on the efforts that are produced by users while accepting tasks.

This paper highlights the advantages of using mobile cloud computing, this could be in terms of its scalability with resource efficiency. The book also discusses the challenges which are related to security and performance, [3].

As outlined in Howe's influential work on crowdsourcing [4], this model transforms task outsourcing by shifting tasks that were once centralized to a broader online community. However, Howe doesn't delve into the technical challenges of integrating these models with mobile cloud platforms, nor does he address real-time risk or QoS monitoring—both of which are key areas we focus on in our work.

Further, explore and discuss the scalability issues and challenges, the system would be relying on anonyms and semi-anonyms users and participants. These insights affect the task management through the system scalability which in this case would be relevant to our project. The research emphasizes the importance of designing a system that can handle security threats while ensuring the QoS.

2.2 Quality of Service

Mobile cloud computing can be a critical factor that ensures that the tasks are completed efficiently and are able to meet performance standards. These environments of Quality of Services have the system's ability that deliver services that advance the performance metrics which are availability, reliability, and response time.

While prior analyses, highlight the importance of QoS in cloud computing systems—emphasizing critical metrics like availability, reliability, and responsiveness—they stop short of proposing practical solutions. Our system directly addresses these challenges through an admin dashboard that features real-time tracking and automated alerts for each of these metrics, enabling proactive management and quick response to potential issues—functionality not covered in other related literature.

2.3 Security Protocols

This paper provides a detailed breakdown and analysis of the various security threats in the cloud environment. This discusses the importance of implementing encryption in the application as it's a technique used in securing sensitive information and data. This also composes of user monitoring feature. The project also involves additional security features like when a user tries to accept more than the required number of tasks acceptances which are set by the system based on the users that are available to accept and complete the tasks, [5].

When a user attempts to accept a task that is more than the limit that is set for each active user. The system would automatically block the user from accepting any other task and create an alert for the administrator.

Highlight how their comprehensive survey covers theoretical encryption strategies and user data protection mechanisms, yet your framework takes these ideas further by implementing automatic blocking of suspicious user behavior and alert mechanisms, bridging the gap between theory and real-world application.

The study offers a strong solution for dynamic bandwidth allocation in SDN environments, tackling issues like packet loss and reduced throughput under changing traffic conditions. Future work will involve testing the algorithm in real-world networks and fine-tuning its allocation strategies to improve performance, especially for best-effort traffic, [6].

In [7], the authors focus on evaluating the performance of SIP servers through software profiling techniques. They analyze how SIP handles session management and the impact of various factors on server performance, such as processing load, response times, and scalability. The study provides insights into optimizing SIP server configurations to enhance efficiency and reliability in multimedia communication systems.

By examining the integration of SIP in CPS, the chapter contributes to understanding how real-time communication protocols can be effectively implemented in systems where physical processes are tightly coupled with computational elements.

Recent studies have made valuable progress in improving Quality of Service (QoS) in new computing environments. For example, [8] introduced a feature learning-based approach for predicting QoS in intelligent edge computing systems. By using deep feature extraction, their method helps improve decision-making by providing a clearer understanding of service quality patterns at the edge of the network. On a different front, [9] looked into enforcing QoS at the hardware level. Their study offers a detailed overview of current technologies, their real-world applications, and the research challenges involved. They underscore the role of low-level resource control in ensuring consistent performance in complex systems.

In the area of crowdsourcing, [10] conducted a comprehensive review of quality control strategies, particularly for open-ended tasks where responses can be subjective. Their work organizes various methods and assesses how well they maintain accuracy and worker reliability. Additionally, [11] explored how clearly a task is defined and how that impacts participant behavior and the quality of their work in crowdsourcing contests. Using a natural experiment, they show that well-structured task descriptions play a key role in achieving better results on online collaboration platforms.

Incorporating such features and information we were able to create a security-enhanced and robust platform that can handle the system complexities of crowdsourcing in a more secure and efficient way.

3 System Development

The system's architecture is based on a client-server model, where devices hosting the web application function as clients, enabling users to accept, reject, and complete tasks. The server acts as the admin role, responsible for allocating tasks, providing encrypted security, monitoring user behavior, and tracking service performance.

The server has been built using the ASP.Net Core Framework of version 8 which is widely supported. This framework provides additional benefits like REST API for task management, where the applications could interact with the API to allow the system to perform custom or out of the application tasks and operations.

3.1 Task Management Module

This module is designed and developed to handle the creation and management of tasks, allowing users to perform necessary actions to complete the accepted tasks, at real-time the admins can also track and monitor the actions of the users (Figure 1).

Task Name	Application	Accepted By	At Risk
Open Chat	Instagram	sas82	⚠ At Risk Task
Open Camera	Instagram	sas82	⚠ At Risk Task
Open Logs	-----		Safe
Open Cogs	Instagram		⚠ Remove Task

Fig. 1: Task Management

3.2 Security Implementation

The system permits the encryption of sensitive information related to users, admins, applications, and the task. This would make the system more resilient from hackers and any unauthorized access to the system cannot get information of such module unless the hashing key is provided to encrypt the information. Together with the possibility that allows the admin to see and be notified if any of the users hold on to the task for a period that is more than the time limit (Figure 2).

```
model.Password = _passwordHasher.HashPassword
        (model, model.Password);
```

Fig. 2: Security Module

The system's security design is built to protect sensitive data from unauthorized access, tampering, or misuse throughout the entire task processing lifecycle. To do this, we use AES-256 encryption to secure both task-related and user-sensitive information. We chose AES (Advanced Encryption Standard) because it's fast, highly secure, and resistant to brute-force attacks, making it ideal for mobile and cloud environments.

For password storage, we rely on ASP.NET Core's built-in hashing mechanism, which uses PBKDF2. This method adds a cryptographic salt and runs multiple iterations, making it tough for attackers to use rainbow tables.

On top of encryption, the platform includes real-time monitoring of user behavior. For instance, if someone tries to take on more tasks than allowed within a set time frame, the system blocks further assignments and alerts the admin. This helps prevent misuse and guards against potential denial-of-service (DoS) attacks caused by task hoarding.

We've also implemented role-based access control (RBAC) to separate permissions between standard users, admins, and super-admins, limiting access to sensitive features and maintaining system integrity.

To prepare for common threats, we simulate and defend against scenarios like:

- Impersonation: Users log in with encrypted credentials and hashed passwords.
- Task manipulation: Task data is encrypted and only decrypted during valid sessions.
- Data leakage: All internal communications happen over HTTPS, and logs are access controlled.

These layered defenses help make the platform highly resilient, ensuring secure task management in a real-time, mobile cloud environment.

3.3 Quality of Service Dashboard

This dashboard is to allow the admins and super admins to overview and monitor the service metrics allowing them to take proactive measures to ensure that their task is being completed and handled correctly. It also allows admins to handle any anomalies or risks that may be associated with users or tasks before they escalate.

3.4 Task Management Process

Task management would be the core functionality of this application and project. This feature allows users to browse through and select the available tasks that were initiated by the admins, allowing users to accept, reject a complete the available tasks. Upon accepting the task, the system logs the information and provides the updated information to the admin. The admin has the control to remove the task from the user if the admin feels that this could be a potential risk.

The pattern followed in the process is as follows:

- System pulls the available tasks as the admin submits a list of tasks associated with the application.
- The available task is displayed to users, for them to accept or reject.
- Once accepted, the system encrypts the task data and assigns it to the user.
- The system can then track the time how long it has been while completing the task after acceptance and display the metric in the admin dashboard for their action and management.

To provide a secure connection between the user information and the system users, we were able to use encryption in the fields like user password and application password. Once any users accept the task, the task details are encrypted and decrypted only when necessary. This is to allow monitoring of user behaviors, to allow admins to manage and see when a user would attempt to accept more tasks than the actual limit per day that is assigned.

In a case when a user tasks more than a specific time which was registered and assigned by the admin. This would trigger a notification to the admin or the application owner. The admin can then view that the user who has the task assigned it now at a risk stage. The administrators can then remove the task from the user if necessary.

The Security Module is a key part of the proposed system. It's built to protect task data, user credentials, and overall application integrity. It works on multiple levels, combining encryption, access control, and behavioral monitoring.

3.4.1 Encryption and Data Protection

All sensitive information—such as user passwords, task details, and admin credentials—is encrypted using the AES-256 standard. This symmetric encryption method ensures strong data confidentiality with minimal performance impact, making it ideal for both mobile and cloud environments.

To store passwords securely, the system uses PBKDF2-based hashing. Each password is salted and hashed multiple times, making it resistant to brute-force and rainbow table attacks.

3.4.2 Authentication and Session Management

User logins are secured with encrypted credential transmission over HTTPS. After a successful login, the system generates a secure session token to maintain session continuity. Idle sessions are automatically terminated to reduce the risk of session hijacking.

3.4.3 Task-Level Security and Access Control

When a task is accepted, its details are immediately encrypted using a session-specific key. This means only the assigned user, during their active session, can view the task. Any attempt to access the task from a different user or an expired session is denied and logged as a security event.

The system also enforces role-based access control (RBAC), ensuring that admin functions—like reassigning or deleting tasks—are restricted to administrators only.

3.4.4 Anomaly Detection and Alerts

A major feature of the module is real-time behavioral analysis:

- If a user tries to accept more tasks than allowed within a given time frame, the system blocks further requests.
- If a task isn't completed within its set time limit, it's flagged as "at-risk", and the admin is notified.
- The admin can then decide to reassign or cancel the task.

All unusual activity is logged with timestamps, user IDs, and action details for future investigation.

3.4.5 System Logging and Admin Tools

Security logs are stored in an encrypted database and track important events like login attempts, failed authentications, and access violations. Administrators can use a dashboard to:

- View and filter logs
- Manually deactivate users
- Adjust risk detection thresholds

3.5 Quality of Service Metrics

The admin dashboard (Figure 3) allows the administrators to monitor in real-time information, and the metrics have the following information present:

- Average Time Completion: The average time completion is developed to allow the administrators to easily understand the tasks that were accepted by users on the average time from accepting to completing the assigned task.
- Task Acceptance Delay: The metric would display the delay from the time the task was displayed to the user and the time it takes to accept that task.
- Failed Task Count: The purpose of this metric is to allow admins to view the sum or count of the tasks that were accepted by users but not completed within the allocated time.

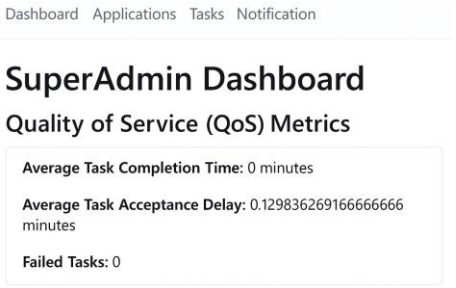

Fig. 3: Admin Dashboard

In the next section, we present the system and analysis.

4 System Implementation and Analysis

The implementation phase of this project was divided into 3 major components: the task management process and workflow, enhancing the security feature of the system, and monitoring the Quality of Service of the tasks that were submitted to users for acceptance. Each of the above-mentioned components was implemented using the combinations of ASP.Net Core, C# programming language, and some component styling using Bootstrap, a style sheet for front-end development.

4.1 Backend Development

The ASP.Net core was used to implement the backend application, infrastructure, and domain. This would be handling task management, security features, and the monitoring of Services real-time. The use of the Entity Framework Core was used for comprehensive database operations.

4.2 Front End Development

The user interface is created to allow users to interact with the database and system, this was developed using Bootstrap version 4.3.1 to ensure allow smooth and modern look and feel.

4.3 Database

The SQL Server Management tool was used to store task data, user data, and system logging information.

4.4 Security

The AES encryption is used to protect sensitive applications and user information; these data were encrypted using key hashing.

4.5 System Architecture

The architecture is developed using the MVC model and architecture, MVC is a standard for Model–View–Controller. Figure 4 shown below presents the breakdown for each principle:

Controller – This can be used to handle the information that is required by the view from the model files. It handles and connects the interface with a database.

Model – The structure holds the context of the Database, which represents the task management data, user data, admin data, and metrics for Quality of Service. Overall, this structure would be used to link the database to the applications, with server, database, and entity information.

View – This folder/structure will be used to handle all user interface implementation and html files. Which is the task list, accepted task, signup and login page, application information, admin notifications, and other views.

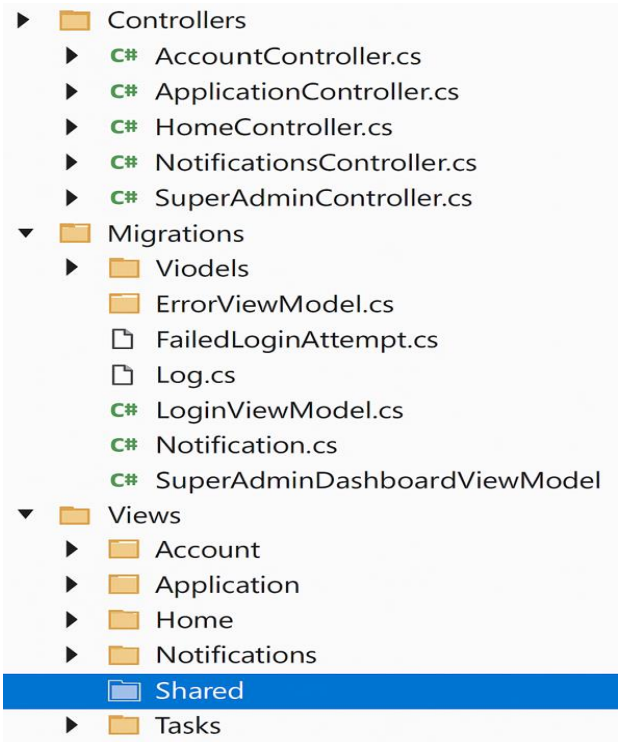

Fig. 4: Architectural Design

4.6 Results

The proposed system addresses the security gaps and enhances the current security state. The developed system was tested under various conditions. Below are the main features of the proposed system:

Secured Task Management Process:

- Task Data Encryption
- Monitoring Task Behaviour of accepted users
- Notify Application Owners of potential security risk

Quality of Service Metrics:

- Real Time Information on the following
- Task Completion Time
- Acceptance Delays
- Failed Task Counts

Risk Monitoring:

- Identify Task Risk
- Admin Notifications

5 Conclusion

To conclude, the paper contributes to both Enhancing Security and Addressing Gaps of Crowd Sourcing Based on Mobile Cloud Computing. We propose a secured and efficient scalable platform for crowdsourcing. We integrated the task management process, enhancing the security prospects and protocols with service monitoring. The proposed system is capable of addressing the vulnerabilities and challenges that are commonly associated with crowdsourcing environments in mobile cloud computing.

By implementing encryption in the application using the AES technique, the system would now be able to secure sensitive information throughout the life cycle. The real-time notification system would also be able to alert the application owners and administrators. These alerts would be to address admins, of potential risks, composes of task mishandling, and delayed completions.

The Quality-of-Service metrics are displayed and implemented on the admin dashboard, which provides detailed insights and information about the application task allocations and users, the metrics include information like task completion, acceptance delays, and the count of failed tasks. The goal of this functionality is to ensure that the admins can and will be able to maintain a high level of performance in the system which would also allow the admins to take actions upon deriving information that may arise.

The use of ASP.Net Core Framework version 8 and the use of Entity Framework Work can provide a robust and scalable structure to the project and application. Together with Bootstrap allowing the front end to look more professional and modern.

Further system improvement and added features can be incorporated such as real-time monitoring between users.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

The authors wrote, reviewed and edited the content as needed and verifies that none utilised artificial intelligence (AI) tools were used. The authors take full responsibility for the content of the publication.

References:

[1] I. G. Ali Babar, "Quality of Service (QoS) in Cloud Computing: A Critical Analysis," *Journal of Cloud Computing,* vol. 4, no. 1, pp. 1–15, 2015. doi: 10.1186/s13677-015-0043-6

[2] R. R. Anhai Doan, "Crowdsourcing Systems on the World-Wide Web," *Communications of the ACM,* vol. 54, no. 4, pp. 86–96, 2011. doi: 10.1145/1924421.1924442

[3] D. Huang, H. Wu, and J. Zhang, *Mobile Cloud Computing: Principles and Paradigms.* Hoboken, NJ: Wiley, 2013. ISBN: 9781118067359.

[4] J. Howe, *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business.* New York, NY: Crown Business, 2008. ISBN: 9780307396211.

[5] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou, "Security and Privacy in Cloud Computing: A Survey," in *Proc. 6th Int. Conf. on Semantics, Knowledge and Grids (SKG),* Beijing, China, Nov. 2010, pp. 105–112.

[6] D. Krishneel, C. Kaylash, and A. Mansour, "Adaptive Quality of Service for Packet Loss Reduction Using OpenFlow Meters," *PeerJ Computer Science,* vol. 10, 2024. doi: 10.7717/peerj-cs.1611.

[7] A. Mansour, K. M. Williams, S. R. Das, and S. Morton, *"Use of Session Initiation Protocol in Multimedia Communications," in Applied Cyber-Physical Systems,* New York, NY: Springer, 2013, pp. 149–174. ISBN: 978-1461473350. doi: 10.1007/978-1-4614-7336-7_8

[8] Y. Zhang, D. Wang, W. Zhang, L. Tan, G. Kibalya, P. Zhang & K. Igorevich, "QoS

prediction in intelligent edge computing based on feature learning," *Journal of Cloud Computing: Advances, Systems and Applications,* vol. 13, 2023, article no. 391. doi: 10.1186/s13677-023-00391-0.

[9] O. Larsson, T. Metsch, C. Klein & E. Elmroth, "Hardware-Level QoS Enforcement Features: Technologies, Use Cases, and Research Challenges," in arXiv, May 21, 2025. doi: doi.org/10.48550/arXiv.2505.15542.

[10] Lei Chai, Hailong Sun & Jing Zhang, "Quality Control in Open-Ended Crowdsourcing: A Survey," arXiv, Dec 5, 2024. doi: doi.org/10.48550/arXiv.2412.03991.

[11] Ye, Kai and Koh, Tat Koon, "Problem Specification in Crowdsourcing Contests: A Natural Experiment" (2023). ICIS 2023 Proceedings, [Online]. https://aisel.aisnet.org/icis2023/sharing_econ/sharing_econ/4 (Accessed Date: February 14, 2024).

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare.