Design and Implementation of Security-Conscious, Location-Sharing in a Geosocial Network

Thesis submitted by

Manas Pratim Biswas (002011001025)
Anumoy Nandy (002011001121)
Kunal Pramanick (302111001005)

under the guidance of

Dr. Munmun Bhattacharya

in partial fulfilment of the requirements
for the award of the degree of

Bachelor of Engineering in Information Technology

Department Of Information Technology
Faculty of Engineering & Technology
Jadavpur University

2020 - 2024

This page has been intentionally left blank

BONAFIDE CERTIFICATE

This is to certify that this project titled ***"Design and Implementation of Security-Conscious, Location-Sharing in a Geosocial Network"***, submitted to the **Department of Information Technology, Jadavpur University, Salt Lake Campus, Kolkata**, for the award of the degree of **Bachelor of Engineering**, is a bonafide record of work done by **Manas Pratim Biswas (Regn. No.: 153760 of 2020-2021)**, **Anumoy Nandy (Regn. No.: 153823 of 2020-2021)** and **Kunal Pramanick (Regn. No.: 159991 of 2021-2022)**, under my supervision.

Countersigned By:

Dr. Munmun Bhattacharya
Assistant Professor
Information Technology

Prof. Bibhas Chandra Dhara
HOD, Information Technology
Jadavpur University

Acknowledgements

We would like to express our sincere gratitude to Dr. Munmun Bhattacharya for allowing us to pursue our final-year project under her supervision. In addition to the meticulous research guidance, her encouraging support for our ideas as well as the constructive criticisms during our experimental and initial developmental stages of the project would not have been possible.

We would also like to thank the entire Department of Information Technology including all the professors, lab assistants and non-teaching staff for being extremely cooperative with us throughout the developmental stages of the project.

Above all, we thank our parents for their support and encouragement in our academic pursuits.

Department of Information Technology
Jadavpur University
Salt Lake Campus, Kolkata

Manas Pratim Biswas   Anumoy Nandy   Kunal Pramanick

JADAVPUR UNIVERSITY

Dept. of Information Technology

Vision:

To provide young undergraduate and postgraduate students a responsive research environment and quality education in Information Technology to contribute in education, industry and society at large.

Mission:

M1: To nurture and strengthen the professional potential of undergraduate and postgraduate students to the highest level.

M2: To provide international standard infrastructure for quality teaching, research and development in Information Technology.

M3: To undertake research challenges to explore new vistas of Information and Communication Technology for sustainable development in a value-based society.

M4: To encourage teamwork for undertaking real life and global challenges.

Program Educational Objectives (PEOs):

Graduates should be able to:

PEO1: Demonstrate recognizable expertise to solve problems in the analysis, design, implementation and evaluation of smart, distributed, and secured software systems.

PEO2: Engage in the engineering profession globally, by contributing to the ethical, competent, and creative practice of theoretical and practical aspects of intelligent data engineering.

PEO3: Exhibit sustained learning capability and ability to adapt to a constantly changing field of Information Technology through professional development, and self-learning.

PEO4: Show leadership qualities and initiative to ethically advance professional and organizational goals through collaboration with others of diverse interdisciplinary backgrounds.

Mission - PEO matrix:

Ms/ PEOs	M1	M2	M3	M4
PEO1	3	2	2	1
PEO2	2	3	2	1
PEO3	2	2	3	1
PEO4	1	2	2	3

(3 – Strong, 2 – Moderate and 1 – Weak)

Program Specific Outcomes (PSOs):

At the end of the program a student will be able to:

PSO1: Apply the principles of theoretical and practical aspects of ever evolving Programming & Software Technology in solving real life problems efficiently.

PSO2: Develop secure software systems considering constantly changing paradigms of communication and computation of web enabled distributed Systems.

PSO3: Design ethical solutions of global challenges by applying intelligent data science & management techniques on suitable modern computational platforms through interdisciplinary collaboration.

This page has been intentionally left blank

Abstract

The recent advancements in mobile and internet technology, coupled with cheap internet data, have witnessed an exponential increase in the usage of internet and people connecting via social media networks. However, the traditional social networking sites such as *Facebook*, *Instagram* or *Twitter* have had witnessed multiple allegations of capturing and selling the user data for their corporate and business purposes. Most importantly, a breach in the user's location data can expose sensitive information regarding that user's frequent places of visit, acquaintances they meet with, food habits and political preferences.

Our project explores the applicability and implementation of a Privacy Network as a scalable Geosocial Networking website. A user-centric design for a secure location sharing is implemented into our project. The final product, *TraceBook*, is a web application that allows the users to register into our website with their details, add or remove friends and most importantly, decide and set their visibility and other privacy parameters very precisely to ensure a secure and robust social media usage. Even more, the users can query their friends based on parameters including but not limited to *age*, *gender*, *college* and *distance*.

The user auth and data is handled by a MongoDB backend, and the user location attributes are handled by a Postgres backend. PostGIS along with PostgreSQL is utilized to create custom SQL queries to fetch the user location details on-demand. Real-time location broadcasting and multicasting is achieved by upgrading the *HTTP* protocol to *WebSocket* protocol and utilizing raw *WebSockets* natively available in the client's browser. A separate WebSocket backend server handles all the WebSocket connections from the client side. Services such as *Vercel* and *Render* are utilized to host the frontend and backend servers. A complete documentation of all the backend API end-points, in compliance with the standardized OpenAPI specifications, is available as a Swagger documentation.

However, scaling WebSockets is challenging. Therefore, a *distributed Redis Pub-Sub* based architecture along with a *HAProxy load balancer* is proposed. Moreover, a *change-data-capture* mechanism using *Apache Kafka* is proposed to keep the MongoDB and PostgreSQL databases consistent and in synchronization with each other.

Keywords: typescript, websockets, mern, postgresql, postgis, redis, haproxy, kafka, social-network-analysis

Contents

List of Tables

List of Figures

Abbreviations

OSN	Online Social Network
POI	Position Of Interest
MERN	MongoDB, Express, React, Node
API	Application Programming Interface
IO	Input/Output
CLI	Command Line Interface
HTTP	Hyper Text Transfer Protocol
`req-res`	Request-Response
JSON	JavaScript Object Notation
URL	Uniform Resource Locator
CRUD	Create, Read, Update, Delete
MVVM	Model-View-View-Model
DOM	Document Object Model
UI	User Interface
`tsc`	TypeScript Compiler
`Pub-Sub`	Publisher-Subscriber
ORM	Object Relational Mapping
ERD	Entity Relationship Diagram
DNS	Domain Name System
SPOF	Single Point Of Failure

This page has been intentionally left blank

Chapter 1

Introduction

This chapter gives a basic introduction about the background of a Privacy Network, its necessity and inception. The later subsections give the consequent flow of the project report.

1.1 Geosocial Networking

Geosocial networking[1] is a type of social networking that integrates geographic services and capabilities such as geolocation to enable additional social dynamics. This technology allows users to connect and interact based on their physical locations, enhancing the social networking experience by facilitating real-time, location-based interactions. Popular applications of geosocial networking include location-based services like check-ins, geotagging, and finding nearby friends or events. These functionalities are supported by technologies such as GPS, Wi-Fi positioning, and cellular triangulation. By leveraging these technologies, geosocial networking can provide users with personalized content and recommendations based on their current location, thereby enriching the user experience. However, it also raises significant privacy concerns, as the collection and sharing of location data can lead to potential risks such as unwanted tracking and profiling. Consequently, there is a growing emphasis on developing secure geosocial networking applications that prioritize user privacy and data protection, incorporating advanced encryption and user consent mechanisms to safeguard sensitive location information.

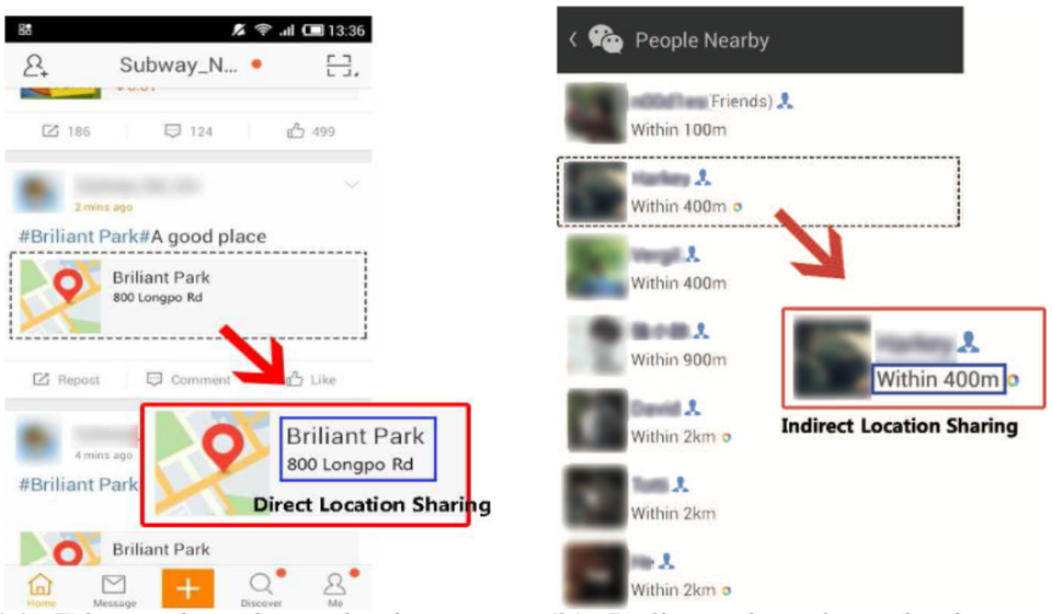
The past decade has seen unprecedented advancements in mobile and internet technologies, resulting in an exponential increase in internet usage and social media connectivity. Affordable internet data has made the online world more accessible than ever before, fundamentally transforming how people interact and communicate. Social media platforms such as Facebook, Instagram, and Twitter have become integral to daily life, providing users with the means to connect, share, and engage with others on a global scale. These platforms, however, have also come under intense scrutiny for their handling of user data, with numerous allegations of data misuse for corporate gain.

1.1.1 Security Vulnerabilities in Geosocial Networking

The convenience and connectivity offered by social media come at a significant cost to user privacy. Traditional social networking sites have been accused of capturing and monetizing user data, leveraging personal information for targeted advertising and other business purposes. This practice has raised serious concerns about data security and user consent, highlighting a critical need for more transparent and ethical data management practices.

Among the various types of data collected, location data is particularly sensitive. A breach in location data can reveal detailed insights into a user's daily routines, including frequent places of visit, social circles, dietary preferences, and even political affiliations. For instance, if a social media platform tracks a user's location, it can infer whether the user visits specific types of restaurants, attends political rallies, or frequents particular neighborhoods. Such information, if exposed, can lead to a myriad of privacy invasions and security risks, including unwanted surveillance, targeted harassment, and identity theft.

Direct and indirect location sharing pose significant risks. Attackers can infer user demographics, such as age, gender, and education, from shared location profiles. Many social networks offer geolocation tags and check-in services, which attackers exploit by crawling web pages to extract location information and Points of Interest (POIs). For instance, direct location sharing on Online Social Networks (OSNs) exposed 16% of users' real POIs, while indirect sharing, like geolocation tags, exposed up to 33%.



(a) Direct location sharing on Weibo (b) Indirect location sharing on Wechat

Figure 1.1: Direct & Indirect Location Sharing

Intentional location spoofing is another major concern. Privileged insiders can use apps like *FakeGPS*[2] to provide fake locations. We address this issue by ensuring that location updates and friends' location queries are securely encrypted end-to-end. Sharing location with a large number of friends may also pose security hazards; hence, we allow users to set distance thresholds for better control over location sharing.

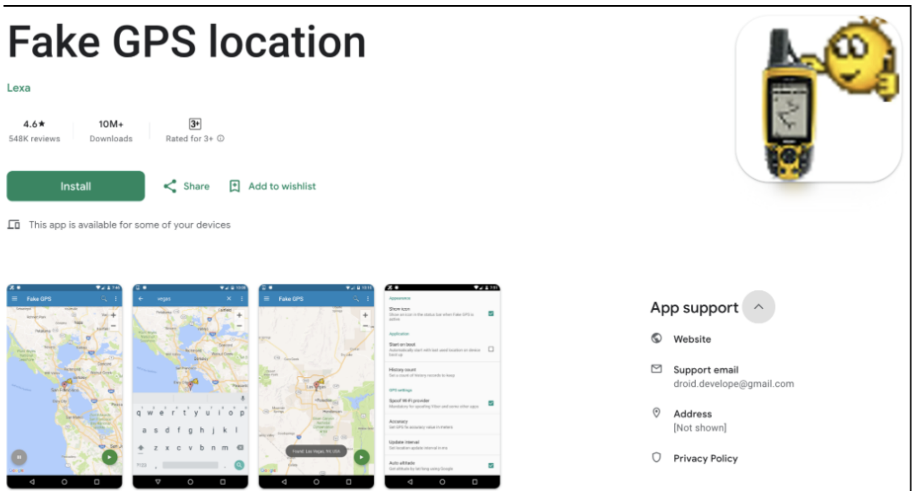


Figure 1.2: Location Spoofing

A three-week real-world experiment[3] with 30 participants revealed alarming results regarding location sharing on OSNs. The study found that direct location sharing exposed 16% of users' real Points of Interest (POIs), while indirect sharing exposed even more, reaching 33%. This underscores the critical need for robust privacy measures in geosocial networking applications.

The exploitation of user data by social media giants has prompted a growing public outcry and a demand for greater privacy protections. High-profile incidents, such as the Cambridge Analytica scandal[4], have underscored the potential for misuse of personal data and the far-reaching consequences of such breaches. These events have sparked a broader conversation about the ethical responsibilities of technology companies and the need for robust regulatory frameworks to safeguard user privacy.

1.1.2 Addressing Privacy Tradeoffs

In response to these concerns, there has been a surge in the development of privacy-centric social networking alternatives. These platforms prioritize user data protection and transparency, offering users greater control over their personal information.

In our current approach, we utilize two security parameters that the user can fine tune to filter his/her location. The user can set their own visibility to either *on/off* and can also *limit their visibility* till a certain range of distance. Thereby, The user can control the location shared to the rest of the users in the social network. We utilize a variable called ***isVisible*** which the user can update to control his/her visibility. *isVisible true* implies that the user wishes to share his/her locations, and *isVisible false* denotes that the user does not wish to share his/her location. The idea is that a particular pair of user can have four possibilities for the privacy parameters, as shown below:

Table 1.1: Privacy Filtration

$Connection$	$isVisible$	$VisibilityLevel$
Friend	True	High (Marker)
Friend	False	Medium (Blue Patch)
Non-Friend	True	Low (Red Patch)
Non-Friend	False	None

- A **Marker** shows the exact location of the user's friend, since the user's friend has allowed his/her location to be shared.
- A **Blue Patch** or a Medium visibility means that the exact location will not be shared to the user's friend, rather a Blue Patch would be shown which denotes the user to be present **within 10 *kms* radius.**
- A **Red Patch** or a Low visibility means that the exact location of the stranger will not be shared to the user, rather a Red Patch would be shown which denotes the user to be present **within 20 *kms* radius.**

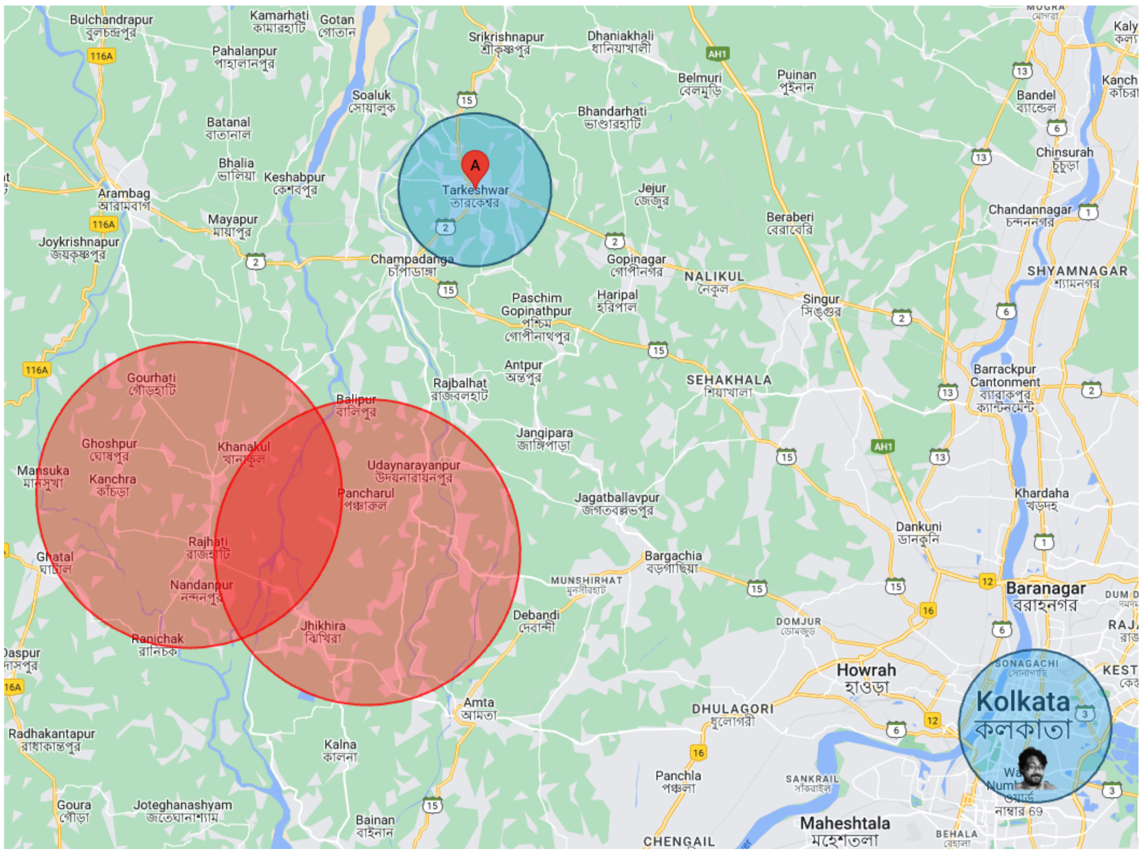


Figure 1.3: Privacy Filtration

1.2 Report Outline: The Privacy Network Approach

The report is organized as follows:

Chapter 2 presents the existing literature on privacy preserving schemes and techniques of location sharing for online mobile online social networks. Further, we summarize the features and technicalities of a similar web application based social networking site called *Loopt* that aimed to solve the similar issue.

Chapter 3 explores the architecture and the implementation details of a Privacy Network - *TraceBook*. We briefly discuss the history and relevance of the MERN stack followed by a thorough scrutiny of the proposed architecture and the consequent coding implementation, CI/CD pipelines and deployment of the web application.

Chapter 4 and Chapter 5 provides a summary of the implementation outputs through a conclusion and discusses various potential future features and proposes advanced architectural to attain a distributed and scalable system.

Chapter 2

Literature Review

This chapter starts by briefly covering the relevant works in designing privacy preserving and efficient location sharing schemes for mobile online social networks. We discuss the case studies and vulnerabilities found through previous works. This is followed by the introduction of a product - Loopt, now discontinued, that tried to solve a very similar problem as that of ours.

2.1 Related Works

In the domain of mobile Online Social Networks (mOSNs), privacy and security have garnered significant research interest. Various privacy-preserving schemes have been proposed, each with distinct advantages and limitations. Early research primarily focused on information privacy, user anonymity, and location privacy. Techniques for maintaining location anonymity often involve encrypting the current location before transmission. K-anonymity methods obfuscate user locations, while dummy locations are used to mask real ones. Location encryption and pseudonym methods, such as mix zones and m-unobservability, are other prominent approaches.

K-anonymity for location privacy involves obfuscating a user's actual location, as demonstrated by researchers like Gruteser and Grunwald[5]. The use of dummy locations, where fake locations are sent along with the real one, has been explored by Kido et al.[6]. Location encryption methods, such as those by Khoshgozaran et al.[7], offer another effective means of protecting location privacy. Pseudonym-based methods, mix zones, and m-unobservability, as explored by Beresford and Stajano[8], have also been developed.

Rahman et al.[9] proposed privacy context obfuscation based on various location parameters to achieve location obscurity. The concept of location sharing with privacy protection in online social networks was initially addressed by SmokeScreen, which facilitated location sharing between friends and strangers. This approach was later enhanced by Wei et al.[10] with MobiShare, which separated social and location information into Social Network Services (SNS) and Location-Based Services (LBS), respectively. However, MobiShare faced the issue of LBS potentially revealing social network topologies during queries.

Li et al.[11] advanced this concept with MobiShare+, introducing dummy queries and private set intersection to prevent LBS from accessing users' social information. BMobiShare

Chapter 3

System Architecture

In this chapter, we go through a brief overview and history of the MERN stack and then have a detailed discussion about the architecture of TraceBook - the frontend, backend, deployment and scaling.

3.1 MERN Stack with TypeScript

MERN stands for *MongoDB*, *Express.js*, *React.js* and *Node.js*. With *MongoDB* as the Database, *Express.js* acts a web server framework (integrated with Node.js), React.js is the web client library, and Node.js is the server-side JavaScript runtime. It helps developers to develop Web apps based solely on full-stack JavaScript. Due to the same JavaScript platform in both the frontend and the backend, uniformity in the codebase is maintained. Naturally, the stack is supported by a vast number of open-source packages and a dedicated community for programmers to increase scalability and maintain software products.

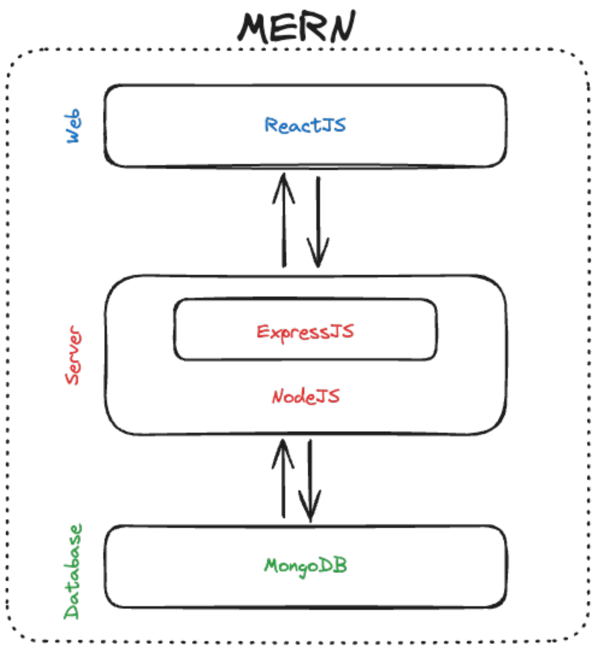


Figure 3.1: MERN Stack Architecture

Chapter 4

Conclusion

This thesis focuses on the implementation of a security-conscious Geo Social Networking web application, named *TraceBook*. To address privacy concerns, *TraceBook* prioritizes user control over location data. Users can define their visibility settings with granular control, ensuring a secure and customizable social media experience.

The front-end technologies used to develop the web application are varied and robust. React, a component-based UI library, is employed to create a smooth and interactive user experience. For styling, the application utilizes *Material UI*, *TailwindCSS*, and raw CSS to ensure a cohesive and visually appealing design. Additionally, *Vite* is used as the build tool, providing a lightning-fast development server and contributing to an overall smoother development experience.

The backend of the web application employs a distributed architecture to ensure scalability and efficiency. A MongoDB server is used to store user authentication and profile data, providing a reliable and efficient database solution for these critical components. For handling user location attributes, a Postgres server is implemented, utilizing PostGIS to enable precise location-based queries. Real-time communication is facilitated by a WebSocket server, which allows for immediate location updates and interactions through a bidirectional full-duplex communication channel. Furthermore, comprehensive documentation for all backend API endpoints is provided using OpenAPI Specs. This documentation adheres strictly to standardized OpenAPI specifications and is accessible in the well-known Swagger format, ensuring clarity and consistency for developers.

Git is used for version controlling, and GitHub hosts the git repository of our application codebase. Platforms like *Vercel* and services like *Render* are utilized to deploy the web application's frontend and backends respectively, while GitHub Actions CI/CD pipelines ensuring smooth deployments.

By prioritizing user privacy and utilizing scalable architecture, *TraceBook* lays the groundwork for a secure and dependable Geosocial Networking platform.

TraceBook can be accessed at ***https://privacynetwork.sanam.live***

Chapter 5

Limitations and Future Work

In this chapter, we discuss a few of the bottlenecks, limitations and technical difficulties that exist in our current approach of building *TraceBook*. In particular, challenges in scaling the system, making it consistent, available and distributed are touched upon. We suggest some of the possible remediation for the same. In the later part of the chapter, we discuss the future scope of the application and propose some more challenging yet relevant features that can be incorporated into *TraceBook*.

5.1 Limitations

TraceBook uses a Websocket architecture for the real-time location sharing among the users. However, WebSockets are difficult to scale.

5.1.1 Scalability

While a single server can theoretically handle 65,536 TCP connections[29] due to the maximum number of available ports, this doesn't directly translate to the same limit for WebSocket connections. A single connection via the `ws:// protocol`[30] is stateful and reserves resources in our *TraceBook* WebSocket server throughout the time a client remains connected with our backend. The connection is blocking in nature. Roughly 20,000 concurrent connections are possible for a 8GB RAM machine, and vertical scaling is capped by hardware limitations.

Resource Consumption: Each open WebSocket connection consumes server resources, including memory, CPU, and network bandwidth. The amount of resources used per connection depends on factors like message frequency, message size, and processing complexity. A server with limited resources may reach its capacity well before reaching the port limit, resulting in performance degradation or connection drops.

Thread Management: Traditional server models often rely on threads to manage individual connections. However, creating and managing a large number of threads can be inefficient and lead to overhead. This is because context switching between threads consumes resources and can become a bottleneck for high connection counts.

Appendix A

Appendix

Table A.1: Resources and URLs

Name of the Server	**URL**
`Privacy Network`	`https://privacynetwork.sanam.live`
GitHub Repository	`https://github.com/sanam2405/PrivacyNetwork`

Bibliography

[1] Wikipedia, "Geolocation networking," 2008-Present.

[2] Lexa and G. P. Store, "Geolocation networking."

[3] H. Li, H. Zhu, S. Du, X. Liang, and X. Shen, "A privacy leakage of location sharing in mobile social networks: Attacks and defense," in *IEEE Transactions on Dependable and Secure Computing*, 2018.

[4] BBC, "Meta settles cambridge analytica scandal case for $725m," 2022.

[5] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. 1st Int. Conf. Mobile Syst., Appl. Services (MobiSys)*, 2003.

[6] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Proc. 21st Int. Conf. Data Eng. Workshops (ICDEW)*, 2005.

[7] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Springer*, 2007.

[8] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervas. Comput*, 2003.

[9] M. Rahman, M. Mambo, A. Inomata, and E. Okamoto, "An anonymous on-demand position-based routing in mobile ad hoc networks," in *Proc. Int. Symp. Appl. Internet (SAINT)*, 2006.

[10] W. Wei, F. Xu, and Q. Li, "Mobishare: Flexible privacy-preserving location sharing in mobile online social networks," in *Proc. IEEE INFOCOM*, 2012.

[11] X. Chen, Z. Liu, and C. Jia, "Mobishare+: Security improved system for location sharing in mobile online social networks," *J. Internet Serv. Inf. Secur.*, 2014.

[12] J. Li, H. Yan, Z. Liu, X. Chen, X. Huang, and D. S. Wong, "Location-sharing systems with enhanced privacy in mobile online social networks," *IEEE Syst. J.*, 2017.

[13] X. Xiao, C. Chen, A. K. Sangaiah, G. Hu, R. Ye, and Y. Jiang, "Cenlocshare: A centralized privacy-preserving location sharing system for mobile online social networks," *Future Gener. Comput. Syst.*, 2018.

[14] M. Bhattacharya, S. Roy, K. Mistry, H. P. H. Shum, and S. Chattopadhyay, "A privacy-preserving efficient location-sharing scheme for mobile online social network applications," 2020.

[15] Y. Combinator, "Loopt," 2005.

[16] B. B. Nerd, “Y combinator’s first batch (yc so5),” 2024.

[17] LoopTMix, “Loopt: The geo social network.”

[18] WikiPedia, “Node.js.”

[19] I. Vilhelmsson, “A performance comparison of an event driven node.js web server and multi-threaded web servers,” 2021.

[20] Nodejs, “What is the event loop?.”

[21] T. F. Stack, “The unbelievable history of the express javascript framework,” 2016.

[22] P. Newswire, “Ibm acquires strongloop to extend enterprise reach using ibm cloud,” 2015.

[23] M. P. Prasad and U. Padma, “Performance analysis of expressjs and fastify in nestjs,” in *Intelligent Control, Robotics, and Industrial Automation* (S. Sharma, B. Subudhi, and U. K. Sahu, eds.), (Singapore), pp. 1037–1049, Springer Nature Singapore, 2023.

[24] Expressjs, “Writing middleware for use in express apps.”

[25] T. Dory, B. Mejías, P. V. Roy, and N.-L. Tran, “Comparative elasticity and scalability measurements of cloud databases,” 2021.

[26] R. Chopade and V. Pachghare, “Mongodb indexing for performance improvement,” in *ICT Systems and Sustainability* (M. Tuba, S. Akashe, and A. Joshi, eds.), (Singapore), pp. 529–539, Springer Singapore, 2020.

[27] M. Grov, “Building user interfaces using virtual dom,” 2015.

[28] B. Cherny, *Programming TypeScript: making your JavaScript applications scale*. O’Reilly Media, 2019.

[29] uNetworking AB, “Millions of active websockets with node.js,” 2019.

[30] I. Fette and A. Melnikov, “The websocket protocol,” tech. rep., 2011.

[31] Alby, “The challenge of scaling websockets,” 2023.

[32] HAProxy, “Haproxy configuration for websocket.”

[33] Confluent, “What is change data capture?.”

[34] F. M. Imani, Y. D. L. Widyasari, and S. P. Arifin, “Optimizing extract, transform, and load process using change data capture,” in *2023 6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pp. 266–269, IEEE, 2023.

[35] Redis, “Distributed redis caching.”

[36] V. Zakhary, D. Agrawal, and A. El Abbadi, “Caching at the web scale: [tutorial],” in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 909–912, 2017.

[37] A. Makris, K. Tserpes, D. Anagnostopoulos, and J. Altmann, "Load balancing for minimizing the average response time of get operations in distributed key-value stores," in *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 263–269, IEEE, 2017.

[38] S. Chen, X. Tang, H. Wang, H. Zhao, and M. Guo, "Towards scalable and reliable in-memory storage system: A case study with redis," in *2016 IEEE Trustcom/BigDataSE/ISPA*, pp. 1660–1667, IEEE, 2016.