# Privacy Enhanced Architecture for Community Key Management in Online Social Networks

**Sudipta Das [1], Dr. Priya Vij [2]**

[1, 2] Department of Computer Science and Engineering, Kalinga University, Raipur, Chhattisgarh.

## ABSTRACT

The exponential growth of online social networks (OSNs) has dramatically altered the ways in which people communicate, share information, and engage with one another. Concerns about data privacy, unauthorized access, and user monitoring have grown in tandem with the use of this link. This research introduces a new paradigm for community key management in online social networks (OSNs) that prioritizes privacy. The suggested model integrates safe data-sharing functions by using the application programming interfaces (APIs) of current OSN platforms, such as Facebook or Orkut. It works atop these platforms. By incorporating a cryptographic module into the user's browser, our design allows application providers to communicate with users while protecting the privacy of their data. A community-based key management method is at the heart of the proposed system. It allows users to autonomously govern access and resource rights, apart from the OSN platform. An encrypted User Register algorithm generates a private key for each user; this key cannot be accessed by the OSN. The three main algorithms used in community management are Build Community, Delegate Permission, and Revocation. These algorithms handle membership management, role assignment, and access control enforcement. By using encrypted storage and retrieval operations, two more algorithms—Upload Resource and Download Resource—handle data exchanges. Kernel members may safely alter access credentials and revoke users using the system's Community Member List (CML). A strong basis for privacy-enhanced online social networking is provided by this adaptable and extensible architecture, which guarantees user autonomy, facilitates safe sharing, and reduces the threat of unreliable third-party platforms.

***Keywords:*** *Social Networks, Key Management, Cryptographic Access, Security Models, Encryption, Online Social Networks.*

## INTRODUCTION

Online social networks (OSNs) play a crucial role in the modern day for sharing information, collaborating with others, and communicating with others. Social media platforms such as LinkedIn, Instagram, Twitter (X), and Facebook enable people from all over the globe to connect with one another, regardless of their physical location or cultural background. But there are major privacy problems due to the fast expansion of OSNs. Location, hobbies, social networks, and employment details are just a few examples of the mountains of personal data that users often divulge. The ease and connectedness offered by these platforms comes at the cost of inadequate security for user data, which leaves users vulnerable to spying, abuse, and unwanted access. Because of this, there is a rising need for privacy-preserving algorithms in OSNs, such as cryptographic methods and effective key management systems. Online social networks that prioritize user privacy (PP-OSNs) seek to keep user information private, protected, and available to authorized parties only. In contrast to conventional OSNs, PP-OSNs prioritize user autonomy by allowing them to maintain control over their own data via decentralized, user-centric models. To keep information private and allow for safe connections, these systems often use sophisticated cryptographic algorithms including homomorphic encryption, public-key encryption, and attribute-based encryption (ABE). Despite these advancements, PP-OSNs still face a significant obstacle in the form of efficient and scalable key management.

The term "key management" describes the steps used to create, distribute, store, update, and revoke cryptographic keys. Key management efficiency is of the utmost importance in OSNs due to the dynamic nature of user groups and the frequent nature of interactions (e.g., friendships are added or withdrawn). While keeping the communication and processing cost to a minimum, a strong key management technique restricts decryption to only the intended receivers. Security breaches, illegal access, and significant performance issues may result from poorly handled keys, particularly in massive networks with millions of users. In the past, key management has been handled by centralized key distribution centers (KDCs). But there are privacy concerns and potential points of failure with centralization, especially when the operator of the service has access to users' unencrypted data. Recent studies have investigated distributed trust models, identity-based encryption (IBE), and blockchain as decentralized solutions to these problems. These approaches aim to decrease dependence on centralized infrastructures while maintaining privacy, usability, and performance. Furthermore, privacy-preserving OSNs cannot be implemented without effective access control methods. Using parameters like role, group membership, or relationship type, users may establish who has access to what material using attribute-based access control (ABAC). In conjunction with attribute-based encryption, ABAC allows for the creation of granular and adaptable access controls that meet the needs of users in terms of their privacy.

On the other hand, effective methods of key generation and distribution that can handle intricate and ever-changing social networks are required for large-scale implementation of these restrictions. The concept of forward and backward secrecy is also crucial. While backward secrecy prevents newly joined users from seeing previously shared material, forward secrecy guarantees that the compromising of a user's present key does not impact the confidentiality of prior conversations. To keep one's privacy intact in ever-changing OSN settings, both are necessary. To provide these features, one needs complex protocols for managing keys that may update or revoke keys with little effort and little impact on users. The importance of privacy-preserving OSNs' usability and user experience has been highlighted in recent study as well. Strong security assurances can be achieved via the use of cryptographic methods, but only if these techniques are applied in ways that end users can understand and use without any hassle. Systems that need intricate key management or regular human interventions are prone to encounter low adoption rates. As a result, making sure PP-OSNs are practical and scalable requires building automated key management systems that are both visible.

This research intends to evaluate several frameworks, protocols, and designs that provide effective key management in privacy-preserving online social networks in light of the above. Analyzed are the security assurances, scalability, and usefulness of both centralized and decentralized systems. To further understand how these technologies support privacy preservation and efficient key management, the research also delves into current breakthroughs in lightweight encryption, blockchain integration, identity federation, and group-based key derivation. This research aims to elucidate the important privacy, performance, and usability trade-offs in PP-OSN design by doing a thorough literature review, system model analysis, and framework proposal analysis. Additionally, it emphasizes potential topics for further study, such as post-quantum cryptography, user-centric key lifecycle management, and access control using artificial intelligence. The ultimate goal of this research is to aid in the creation of OSNs that are safe, efficient, and easy to use while also protecting the digital privacy of their users.

## REVIEW OF LITERATURE

Gao, Yuan et al., (2022) For example, Facebook has more than 1.23 billion active users per month, demonstrating how pervasive social networks have become as a means of human interaction and communication. Massive amounts of user data have been produced by the meteoric rise of social media platforms. For instance, consider Twitter, which produces 500 million tweets daily and around 200 billion

tweets annually. Advertisers, app developers, and others stand to gain from this data, and it may also aid people's daily lives. But using learning algorithms to deduce private information from public data can be a privacy breach waiting to happen. As a result, protecting users' privacy online is a top priority for social media researchers.

Voloch, Nadav & Gal-Oz, Nurit. (2021). The rise of OSNs has made them indispensable for global communication and engagement. Because of the advantages and increased social exposure brought about by technology advancements in the last 20 years, the whole concept of privacy has been put to the test in online communities where members actively exchange material. Though they may want to share private information with friends and coworkers, users of online social networks (OSNs) may unwittingly expose it to malicious actors, social bots, imposters, spammers, or data-harvesters. Many security models, such as Access Control, Relationship based models, Trust based models, and Information Flow control, have been created for OSNs with the goal of preventing this information leakage. In line with other studies, we state that in order to overcome the limitations of each model, a combined strategy is necessary. This study introduces a novel privacy protection paradigm that consists of three distinct stages that handle the model's three primary components: trust, role-based access control, and information flow. The model takes into account the user's sub-network and sorts the user's direct links to roles. To define the strength of the connections in the network, it uses publicly available data such the sum of all friends, the age of the user's account, and the length of time that friends have been in the network. It also calculates the degree of familiarity or animosity between a user and their network members based on the pathways that information flows between them, which is an evaluation of trust. Last but not least, it allows for improved privacy management inside the social network and gives more accurate and practical choices about information sharing. In order to prove that our approach can effectively safeguard the privacy of an unsuspecting user, we have conducted thorough tests with both simulated and actual user networks. We have checked each part of the model independently and compared the results from two methods. The results demonstrate a robust relationship between the algorithmic judgments and the user-initiated decisions.

Guo, Guanglai et al., (2020) the public and academic communities have taken an interest in the issue of data privacy as a result of the proliferation of online social networks (OSNs). In order to address the privacy concerns associated with OSNs, we provide a novel approach to encrypting content shared on social networking websites, one that takes into consideration the unique characteristics of these platforms. One or more trustworthy users work together to establish autonomous private communities, or zones, under this architecture. No outside party is involved in this process. Those who are officially permitted to enter a zone cannot see any of the sensitive content (posts, images, etc.) included inside. To get access to the contents of a zone, a user has to join the zone with the permission of an authorized member. Then, she may use her private key in conjunction with this authorization to access the zone. Our permission concept stands out since it does not involve secret information and may be stored in the user's OSN account. Since a user only needs to keep track of one secret key and may utilize as much public permission as she wants to access information in multiple zones, this design of public permission significantly minimizes user-side overhead on secret key administration compared to earlier work. Additionally, our system enables the easy delegation and revocation of access permissions. To test its computational capability to a satisfactory degree, we build a prototype. Along the same lines, we demonstrate that our structure is semantically protected against selected plaintext, existential forgery, and key forgery attacks.

Siddula, Madhuri et al., (2018) more and more people are worried about their privacy on social media. It alludes to a number of problems with social networks, such as users' privacy, connections, and qualities. Every facet of social network privacy is complex and multi-faceted. Take user privacy as an example; it encompasses a wide range of issues, such as the confidentiality of users' location and other personally

identifiable information. We hope that our study on social network privacy will serve as a springboard for future studies in the field. We showcase a range of models and strategies that preserve privacy, such as naïve anonymization, perturbation, or the construction of an entirely new network. Our study builds on previous efforts by other scholars who saw social networks as graphs where individuals are nodes and friendships between users are linkages. Protecting these network nodes and linkages is our area of research. In order to help future scholars in this field, we also go over all the databases that are now accessible and alternative systems that have been suggested.

Jagannathan, Sharath Kumar & N, Maheswari. (2017). an integral part of everyone's lives in the modern period is online social networks such as Facebook, Twitter, etc. The potential for the disclosure of personally identifiable information is always there in the public domain when it comes to social media data pertaining to any particular company. Before data is made public online, it is necessary to ensure that individual organizations and enterprises' privacy is protected. As a result, studies in this field have persisted for a long time and continue to this day. Tabular data, often known as relational data, and social network data, represented in graphs, may both benefit from the many current strategies that address the issue of privacy. Structured complicated graph data, which includes vertices (individuals) and edges (connections or relationships between them), is inherently more difficult to work with than tabular data. For nodes in social networks, there are a number of approaches that guarantee their privacy, such as K-anonymity, L-diversity, and T-closeness; for edges, there are methods like edge perturbation and edge randomization that guarantee their privacy. Ongoing efforts to develop new methods that integrate with existing ones, such as K-anonymity, edge perturbation, edge randomization, and L-Diversity, are dedicated to improving the privacy of relational and social network data.

Samanthula, Bharath Kumar et al., (2015) the many benefits of online social networks (OSNs), such as the ability to communicate with friends and family in real time and share relevant content, are driving their ever-increasing popularity. In order to broaden their social circle and get information from a variety of sources, users often seek out new acquaintances. Extensive research has focused on friend recommendation recently because of its importance as an application in many OSNs. There is an urgent need to create buddy recommendation algorithms for social networks that protect users' privacy in light of rising privacy concerns. In this research, we provide two new approaches to privacy-preserving buddy recommendations for specific users based on the common neighbor's proximity metric. The first approach makes use of a universal hash function for efficiency and is based on the features of an additive homomorphic encryption scheme. The second approach makes effective use of the idea of anonymous message routing to safeguard the anonymity of the source while still providing accurate friend recommendations. We also discuss the specifics of putting the two approaches into action and conduct an empirical comparison of their efficiency and accuracy. Depending on the underlying needs, users or the network provider may select between the two suggested protocols, which provide a trade-off among security, accuracy, and efficiency.

**Proposed System**

Here, we provide a group-oriented convergence cryptosystem-based private OSN architecture. In this design, collaborative apps are a common means for OSNs to share data.

**Community and Member Category**

We begin by defining community, the central idea of our approach, before describing the scheme's architecture. A community is an informal group of people who share an interest, according to the conventional wisdom about social networks. Collaborative web apps manage and organize a community in our own OSN. If you want to be able to make content and see material in a community, you have to

join that community. Examples of popular Facebook apps include the Wall, a user-and friend-driven forum where posts and comments are displayed; Facebook Photos, which allows users to add comments and tags to photos and share them with friends; and Flickr, a photo-sharing and management app that lets Flickr members comment on individual photos.

To build the aforementioned OSN collaboration apps, we provide a generic model. Here is the system architecture of our model. The web-based apps and the storage of published data are the responsibility of a third party under this architecture. There are a few things it offers to consumers as well, such a web browser service. However, for a private OSN, we are not requiring this third party to be reputable. Our methodology works well in preexisting social network settings like Facebook, Flickr, and Myspace, as well as in cloud computing environments.

## Our Model and Architecture

One possible location for our private OSN model is on preexisting social network platforms like Facebook, Orkut, etc., where programmers can typically build "applications" to expand the kinds of data that can be saved, processed, and shared via these platforms' user interfaces. Our architecture's application data flow is shown in Fig. 2. All interactions between application providers and end users are handled by an OSN Platform API in this paradigm. Everyone from kernel members to unauthorized users may access OSN platforms and utilize URLs to contact application providers. Application developers register the addresses of their servers with the platforms, and the platforms evaluate the data sent with these requests and send it on to the servers over the Internet. Based on user input that the platform understands, the application server executes specified actions, which may include database transactions. After that, the application server sends an HTML page with platform-specific markup, scripts included, to the platform. Users get the interpreted output page after the platform replaces platform-specific markup with standard HTML and JavaScript. The client's browser is utilized to decode the output page using an ActiveX-based cryptography module.

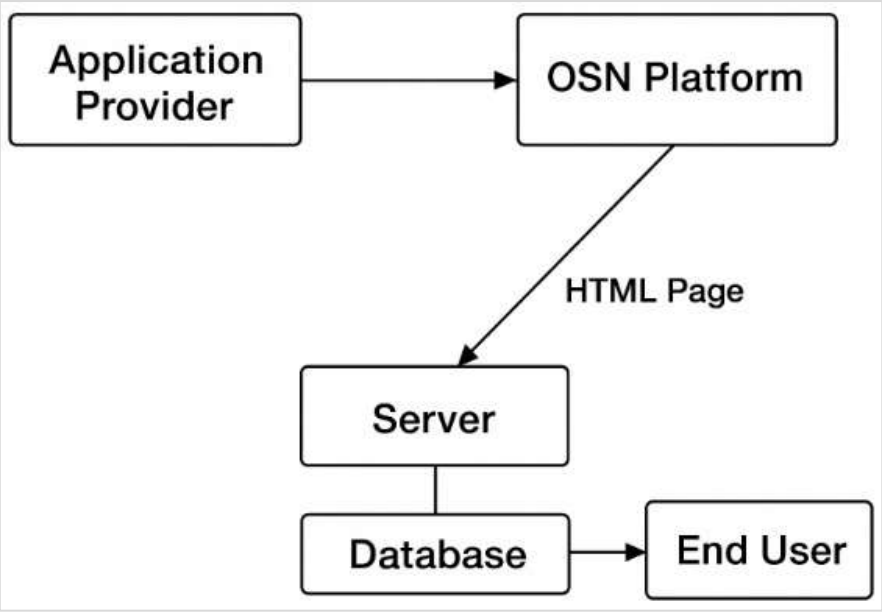


**Figure 1: The Application Data Flow for Our Architecture**

The resource publisher uses encryption and key management on our GCC scheme to impose access control in this architecture. Drawing on the application dataflow mentioned earlier, the following process for publishing and accessing resources described:

- A social network's User Register algorithm allows users to choose a favorite label, create a private key, and register their label on an OSN platform.
- A user can form a community with a group of trusted friends on an OSN platform when she wants to share resources with others. At last, every member receives a community key that can be used for accessing, managing, and maintaining the resources in this community.
- If a user wants to join a community, her friends who have the key can use the Delegate Permission algorithm to give her an APK.
- If a user wants to post a message or resource to the community, she chooses the key, uses her private key to encrypt the resource, and then sends the encrypted data to the storage server.
- To finish, when a user wants to post a resource to the community, she chooses the community key, invokes the Upload Resource algorithm to encrypt the resource, and finally transmits the encrypted data to the storage server.
- If a single user in the community manages to decrypt data stored on the server, she may use her private key and APK to access the original post or resource using the Download-Resource technique.

Our explanation suggests that a subset of the kernel is responsible for client-side access control and key management enforcement. So that the community may be built in an independent and collaborative manner, without the participation of a system manager, our design does not require that the system manager be trusted to operate a private OSN. Our architecture must meet several critical performance and security criteria, including independence, autonomy, cooperation, authentication, and revocation, in order to allow key management-based access control without a system manager.

## DESIGN AND IMPLEMENTATION OF A SECURE COMMUNITY KEY MANAGEMENT MODEL

Here we lay out our plan for managing community keys using the aforementioned architecture. In order to create this plan, our research takes into account the following issues: What is the kernel group's concept of a community? How is the community keys created and distributed by the authorized members? What is the process for members to provide permissions for accessing a community? How can an unreliable third party, like the OSN platform, verify the identities of the community's kernel members?

Considering these issues, we provide the following community key management scheme: Community management in OSN primarily utilizes three algorithms—Build community, Delegate Permission, and Revocation—to construct communities and grant or revoke access permissions independently of OSN. Two algorithms—Upload Resource and Download Resource—are used for the creation, request, update, and deletion of resources. Every user in OSN is assigned a unique private key by the User Register algorithm, and OSN is guaranteed not to know this key. The community member list (CML) is also kept and enforced by the community. By using a cryptographic revocation mechanism, kernel members may remove themselves from access to resources or modify their CML. Furthermore, our model's storage services enable two data storage and retrieval actions: upload and download. These operations are made possible by our GCC scheme's encryption and decryption procedures. In a nutshell, the algorithms outlined here may provide various members flexible and rapid access to data and resources based on their rights.

### User Register

The first step is for the system manager to call $q\!et(\kappa)$ in order to create a public global parameter $p$. By executing $Regib(id)$, each user $ui$ in OSN may produce his private key $ui.id$ and choose a preferred label $ui.sk$. Following the user's submission, the label is registered by the management.

**Algorithm 1 User Register ($\kappa$):**

1: manager: $p \leftarrow Setu(\kappa)$;
2: $ui$: choose a favorite $ui.id$;
3: $ui$:$ui.sk \leftarrow Registe(ui.id)$;
4: $ui \rightarrow$manager: $ui.id$;

Keep in mind that the manager doesn't know the user's private key and only needs to run $Se(\kappa)$ once.

**Build Community**

A group of verified individuals may create a community with the help of the Build Community feature. Instead of a single user defining a community, our model has several users working together to create it. In addition, the community key is not set by a single user or the system management, but rather acquired via the convergence of information of these members.

For a group of reliable individuals $\mathcal{S} = \{u1, \cdots, um\}$, The following instructions are for anybody in $\mathcal{S}$, known as the dealer, to construct the community key $gk$: Someone randomly generates the dealer's hand $g \in \mathbb{G}$ on behalf of this group and gives it out to everyone in $\mathcal{S}$; everyone in $\mathcal{S}$ gives back a temporary public key $ui.pk$. (As his private key's commitment) in relation to $g$ for $i \in [1,]$; thereafter, the dealer creates convergence data $\Sigma$. The community key $gk$ in terms of $CKey(u.sk, \Sigma)$ without the manager's assistance, from all temporary public keys $\{u1.pk, \cdots, um.pk\}$.

**Algorithm 2 Build Community ($dealer, \mathcal{S}$):**

1: dealer: $g \leftarrow Rando(p)$; // to generate a random integer.
2: dealer $\rightarrow \mathcal{S}$: distribute $g$to all members in $\mathcal{S}$;
3: $ui$:$ui.pk \leftarrow EGSetu(ui.sk, g)$;
4: $ui \rightarrow$dealer: $ui.pk$;
5: dealer: $\Sigma \leftarrow$Converge($S$=$\{u1.pk, \cdots, um.pk\}$);
6: dealer: $gk \leftarrow$CKeyGen($u.sk$, $\Sigma$);
7: dealer $\rightarrow \mathcal{S}$: distribute $gk$ to all members in $\mathcal{S}$;

**Delegate Permission**

One way for a community member to provide access to her friends is via the permission delegation procedure. Delegate Permission is an algorithm that allows members to grant other users the "read" privilege of a community. This technique can only be used to delegate access rights by kernel members and full authorized members. This is to prevent unlimited delegation. This process involves two stages: 2) the member's friends get the access permission $km$ safely from the member and 1) the permission is produced based on the user's label. We construct a safe channel using ElGamal encryption.

**Algorithm 3 Delegate Permission ($ui, uj$):**

1: $ui$:$g \leftarrow$obtain from $gk$;
2: $ui \rightarrow uj$:$g$;
3: $uj$:$uj.pk \leftarrow EGSetu(uj.sk, g)$;
4: $ui \leftarrow uj$:$uj.pk$;
5: $ui$:$uj.pm \leftarrow P\ ermissio(ui.sk, gk, uj.id)$;
6: $ui$:$c \leftarrow EGEncryp(uj.pk, uj.pm)$;
7: $ui \rightarrow uj$:$c$;
8: $uj$:$uj.pm \leftarrow EGDecryp(uj.sk, c)$;

Before her buddy $uj$ may be granted access, member $ui$ must obtain the generator $g$ in the community key $gk$. In response to $g, uj$ generates a temporary ElGamal public key according to $EGSe(uj.sk, g)$ and sends it back to $ui$. Next, using his private key, the data received from $uj$, and the community key $bk$, $ui$ determines $bb$'s access authorization. Following that, $ui$ encrypts the authorization using $uj$'s temporary public key and then transmits it to $uj$. At last, $uj$ uses her private key to decode the encrypted text and retrieves the authorization to access the protected content.

If the member $ui$ wishes to delegate the "write" right to her friend, she only needs to transmit the community key besides the permission $pm$. That is, $ui$ merely replaces the line 6 by $c \leftarrow EGEncr(uj.pk, mj.pm||gk)$, where // denotes the concatenation operation for two strings.

**Upload Resource**

In order to publish a message to the community, a kernel member or a fully authorized member uses the Upload Resource function. In order to execute this procedure, the member must possess a valid community key $gk$, since encryption has been established. Authorized members cannot post messages because of this. Furthermore, we provide an effective authentication technique - $CAbhenbid(A,B)$- for members' identify verification. Illegal users are unable to send cipher texts that are invalid thanks to this method.

**Algorithm 4 Upload Resource ($ui, F$):**

1: $u \leftrightarrow SN : b \leftarrow F\ Athenticate(ui, S\ N\ P)$;
2: if $b$is true then
3: $ui: C \leftarrow Encryp(ui.sk, ui.pm, gk, F)$;
4: $ui \rightarrow SN : C$;
5: $SN\ P \rightarrow C\ S$ : upload($C$);
6: end if

For example, let's pretend that $ui$ is interested in publishing $F$ for a certain community G. At the outset, the $ui$ checks her authorization status by interacting with the SNP. She may encrypt the message and send the cipher text to SNP once the $ui$ passes the authentication process. The last step is for the SNP to send the encrypted text to an SSP.

**Download Resource**

Messages in a private OSN may be accessed by members using the Download Resource feature. This function is run on the end user's cryptography module to increase performance. $ui$ Any member of a given community may decode encrypted materials retrieved from the social network platform and storage server according to the algorithm by utilizing the user's private key $sk$ and the access permission $pm\ Decr(ui.sk, ui.pm, C)$. Because of this, the storage provider of encrypted data may be accessed by any authorized member of a private OSN.

**Algorithm 5 Access Resources ($ui, C$):**
1: $ui: F \leftarrow Decryp(ui.sk, ui.pm, C)$
2: $ui: b \leftarrow CV\ erif\ (F, C)$
3: if $b$is true then
4: $ui$: Message is intact and output $F$
5: end if

The GCC method offers an effective verification algorithm for message integrity checks $CV\ erify$ for the encrypted message by the use of the cryptographic Hash algorithm. Therefore, after decrypting the cipher text, the member may check whether the message is still intact. If the outcome of this procedure is $true$, the message may be sent back to the web browser.

**Revocation**

All permitted members may have a specified group of members $\mathcal{R}$ excluded using the Revocation function. A private OSN's security may be efficiently maintained over the long term using the revocation process, which prevents the exposure of privacy. Here is how we may implement the revocation using the revocation algorithm in the GCC scheme: Using the user's public label to acquire a collection of revoked members $\mathcal{R}$, either the kernel or a fully authorized member may call the $Revocat\ (ui.sk,\ gk,\mathcal{R},\ F)$ in order to encrypt the message $F$ using the community key and the private key. Even after such a revocation, the authorized user may still access all group resources.

**Algorithm 6 Revocation ($ui$,$\mathcal{R}$, $F$)**

1: $ui$:$C$←Revocation($ui.sk,\ gk\ ,\ \mathcal{R},\ F$)

She only has to add the authorized member to the community's revoked members list (RML) and make this RML public if kernel members want to permanently remove their authorization. Encrypting the message before posting it to the community is as easy as using this RML as the set $\mathcal{R}$.

Take note that the GCC approach rigorously limits the number of revoked users to fewer than the group's kernel users. By using a random key pair during the generation of the community key, we may simply increase the number of revoked users, thereby enhancing the revocation capacity.

**RESULTS OF THE STUDY**

**Implementation of The GCC Scheme**

To prove that our technique might work, we used an experimental GCC cryptosystem. The system was built using the standard C++ language in the QT environment, which allows for deployment on several platforms. The three parts that make up this system are the browser software, the private social network platform, and the cryptographic module. To deal with integers of any precision, we used the GNU multiple precision arithmetic library (GMP) in the cryptography module. The next step was to build a finite field mathematics library that would allow pairing-based and elliptic curve cryptosystems to operate in a run-time environment, according to Stanford's PBC liberty. Furthermore, a Group-Oriented Convergence Cryptosystem library was created to implement several GCC algorithms. This library is built on the finite field arithmetic library. Last but not least, a lightweight private social network platform and the GCC algorithms collaborated to provide browsers services for key-label management, authentication, and encryption.

**Application for a Blog Management**

We provide a blog administration system that allows users to manage who may access their data directly, without relying on any third party. Blog entries, comments, and photographs may all be edited and published using this system. Data entered into this system may be either "public," which is accessible to everybody, or "protected," which is viewable only to the people in the user-defined community. A server stores all of the blog posts.

Before adding new information to her blog, a user must choose which pieces of data to make public and which to keep private. Before encrypting her protected data using her keys and the community key, she chooses which members of the community will have access to it. The server receives both encrypted and public data. The server provides user *A*'s with data whenever an individual in the system visits her blog. She sees the public data straightaway, but the protected data has a default page that makes it useless to the visitor. Before she can see the whole material, she has to verify her authorization to enter the community by looking at the header of the encrypted text. If she is a legitimate user, she can decipher the encrypted message and see everything within; otherwise, she has no idea what the protected information is.

## CONCLUSION

A move toward privacy-preserving designs that safeguard user data from unwanted access and abuse is important due to the exponential growth of online social networks. Although OSNs are great for connecting people and sharing information, they may be dangerous for users' privacy, especially in highly centralized settings. The importance of effective key management in allowing safe and privacy-aware OSNs has been emphasized in this work. The success of cryptographic techniques like attribute-based encryption and decentralized trust models depends on key management systems that are both scalable and easy to use, while these methods do provide some promising answers. Supporting dynamic social structures, keeping information secret both forwards and backwards, and ensuring that only authorized individuals may access data are all benefits of efficient key management. Nevertheless, performance limitations and user experience concerns must be considered throughout the implementation of these protocols. Blockchain integration, AI-driven automation, and post-quantum resilience are a few examples of the novel solutions that need to be explored in the future to meet the increasing need for secure digital communication while still being easy to use and efficient. By moving forward with these initiatives, we can provide the groundwork for social media platforms of the future that respect users' right to privacy without sacrificing the interactivity and accessibility that characterize today's online interactions.

## REFERENCES

1. Y. Gao, Y. Li, Y. Sun, Z. Cai, L. Ma, M. Pustišek and S. Hu, "IEEE Access Special Section: Privacy Preservation for Large-Scale User Data in Social Networks," *IEEE Access*, vol. 10, pp. 4374–4379, 2022.
2. N. Voloch and N. Gal-Oz, "A Trust Based Privacy Providing Model for Online Social Networks," *Online Social Networks and Media*, vol. 24, Art. no. 100138, 2021.
3. G. Guo, Y. Zhu, Y. Ruyun, W. Chu, and D. Ma, "A privacy-preserving framework with self-governance and permission delegation in online social networks," *IEEE Access*, vol. PP, no. 99, pp. 1–1, 2020.
4. L. Zhang, L. Li, E. Medwedeff, H. Huang, X. Fu, and R. Wang, "Privacy protection of social networks based on classified attribute encryption," *Security and Communication Networks*, vol. 2019, Article ID 9108759, pp. 1–14, 2019.
5. R. G. Pensa, G. D. Blasi, and L. Bioglio, "Network-aware privacy risk estimation in online social networks," *Social Network Analysis and Mining*, vol. 9, no. 1, pp. 1–15, 2019.
6. P. van Schaik, J. Jansen, J. A. Onibokun, J. Camp, and P. Kusev, "Security and privacy in online social networking: Risk perceptions and precautionary behaviour," *Computers in Human Behavior*, vol. 78, pp. 283–297, 2018.
7. M. Siddula, L. Li, and D. Li, "An empirical study on the privacy preservation of online social networks," *IEEE Access*, vol. PP, no. 99, pp. 1–1, 2018.

8. W. Wei, S. Liu, W. Li, and D. Du, "Fractal intelligent privacy protection in online social network using attribute-based encryption schemes," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 3, pp. 736–747, 2018.
9. X. Zhang, Q. Zhou, C. Gu, and L. Han, "The location privacy preserving of social network based on RCCAM access control," *IETE Technical Review*, vol. 35, no. sup1, pp. 68–75, 2018.
10. Z. Wang, Z. Ma, S. Luo, and H. Gao, "Enhanced instant message security and privacy protection scheme for mobile social network systems," *IEEE Access*, vol. 6, pp. 13706–13715, 2018.
11. A. De Salve, R. Di Pietro, P. Mori, and L. Ricci, "A logical key hierarchy based approach to preserve content privacy in decentralized online social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 5, pp. 494–508, 2017.
12. S. K. Jagannathan and N. Maheswari, "A survey on privacy-preserving techniques for social network data," *Asian Journal of Pharmaceutical and Clinical Research*, vol. 10, no. 13, pp. 1–12, 2017.
13. E. Luo, Q. Liu, and G. Wang, "Hierarchical multi-authority and attribute-based encryption friend discovery scheme in mobile social networks," *IEEE Communications Letters*, vol. 20, no. 9, pp. 1772–1775, 2016.
14. J. Pang and Y. Zhang, "A new access control scheme for Facebook-style social networks," *Computers & Security*, vol. 54, pp. 44–59, 2015.
15. B. K. Samanthula, L. Cen, W. Jiang, and L. Si, "Privacy-preserving and efficient friend recommendation in online social networks," *Transactions on Data Privacy*, vol. 8, no. 2, pp. 141–171, 2015.
16. L. Schwittmann, M. Wander, C. Boelmann, and T. Weis, "Privacy preservation in decentralized online social networks," *IEEE Internet Computing*, vol. 18, no. 2, pp. 16–23, 2014.
17. H. Hu, G. Ahn, and J. Jorgensen, "Multiparty access control for online social networks: Model and mechanisms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 7, pp. 1614–1627, 2013.
18. M. Shehab, A. C. Squicciarini, G. Ahn, and I. Kokkinou, "Access control for online social networks third party applications," *Computers & Security*, vol. 31, no. 8, pp. 897–911, 2012.
19. P. Mittal, C. Papamanthou, and D. Song, "Preserving link privacy in social network based systems," *CoRR*, 2012.
20. P. Mittal, M. Caesar, and N. Borisov, "X-vine: Secure and pseudonymous routing using social networks," *arXiv preprint*, arXiv:1109.0971, 2011.
21. S. Jahid, P. Mittal, and N. Borisov, "Easier: Encryption-based access control in social networks with efficient revocation," in *Proc. of the 6th ACM Symposium on Information, Computer and Communications Security*, 2011, pp. 411–415.
22. J. Sun, X. Zhu, and Y. Fang, "A privacy-preserving scheme for online social networks with efficient revocation," in *Proceedings - IEEE INFOCOM*, vol. 1, no. 1, pp. 2516–2524, 2010.
23. S. Bhagat, G. Cormode, B. Krishnamurthy, and D. Srivastava, "Class-based graph anonymization for social network data," *PVLDB*, vol. 2, no. 1, pp. 766–777, 2009.
24. B. Krishnamurthy and C. E. Wills, "Characterizing privacy in online social networks," in *WOSN 2008*, 2008.
25. M. M. Lucas and N. Borisov, "Fly by night: Mitigating the privacy risks of social networking," in *Workshop on Privacy in the Electronic Society (WPES)*, 2008, pp. 1–8.