Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 4, April 2014, pg.664 – 670

RESEARCH ARTICLE

Social Authentication and Untrusted Clouds for Secure Location Sharing

Miss. Priyanka K. Shinde[1], Prof. Nitin R. Chopde[2]

[1]Department of Computer Science &Engineering, G. H. Raisoni College of Engineering, Amravati, India

[1] shinde_priyanka.ghrcemamecse@raisoni.net

Abstract— ***Recently, many location-sharing services (LSSs) have emerged that share data collected using mobile devices. However, research has shown that many users are uncomfortable with LSS operators managing their location histories and that the ease with which contextual data can be shared with unintended audiences can lead to regrets that sometimes outweigh the benefits of these systems. In an effort to address these issues, we have developed SLS: a secure location sharing system that combines location-limited channels, multi-channel key establishment, and untrusted cloud storage to hide user locations from LSS operators while also limiting unintended audience sharing. In addition to describing the key agreement and location-sharing protocols used by SLS, we discuss an iOS implementation of SLS that enables location sharing at tuneable granularity through an intuitive policy interface on the users' mobile device.***

Keywords— ***Key Management; Location Tracking; Presence Systems; Privacy; Security***

I. INTRODUCTION

Present location and message sharing system asks to input the personal information which fails to protect the privacy of information. The recent explosion in mobile computing and social networking has led to deployment of a wide range of location-sharing systems both stand-alone in nature (e.g., Google Latitude , FourSquare, or Glympse) as well as integrated with other social networking platforms (e.g., Facebook places)These types of systems allow a user to share her geographic location with her social contacts either as a first-class data object or as support for other content (e.g., attaching one's location to restaurant review). This sharing can be done in a near seamless manner, particularly when the LSS is embedded within a larger social platform. Despite their popularity, LSSs are not without their own security and privacy problems. By their very design, these systems have the implicit shortcoming that sharing one's location with social contacts requires sharing this location with the LSS operator as well. This can lead to undesirable profiling of users by third parties, or increase users' exposure risk in the event of an LSS compromise. In addition, it is has been shown that social networks in general [17] and LSSs in particular [14] can sometimes lead to situations in which users experience regrets after (over)sharing information. This is often the result of the so-called unintended audience problem, in which data is shared with individuals other than those with whom the subject intended to share. This may manifest as a result of a location being automatically attached to content posted on a social network, accidental sharing of a location with a user's entire set of contacts instead of a restricted subset, or posting a location that contradicts other statements made by the user [13]. The latter problem is symptomatic of both LSS and access control complexity. For instance, it is well-known that users' social networks have many more contacts than they interact with on a day to-day basis: a 2011 poll of 1,954 British citizens found that the average person had 476 Facebook friends,

but only 152 contacts in their cellular phone. Furthermore, research studies have shown that users frequently make mistakes when authoring even basic access control policies in commodity systems [4,9]. As such, it is clear that accidents and misconfigurations can lead to over sharing in large social networks. On the other hand, the problem of required sharing with LSS operators is one of economic incentives: the ability to study user habits and carry out targeted advertising provides revenue for operators of the systems.

An interesting observation, however, is that current generation Smartphone's are capable to explicitly manage a user's published location history and Furthermore store rich information about a users' close contacts (e.g., email addresses, phone numbers) and have access to multiple channels of communication (e.g., WiFi, 3G/4G, Bluetooth, SMS). As a result, it is possible to develop robust key exchange protocols that allow location data to be selectively encrypted prior to upload, thereby preventing snooping attack by the SaaS provider and limiting incidences of over-sharing.

In this paper, we describe Secure Location Sharing (SLS) that allows users to set up secur location sharing with selected contacts by pairing devices in one of two ways. Users who happen to be physically co-located can use location-limited visual channels to pair devices (similar to [12]). Users who are located apart from one another can instead leverage multiple communication channels (e.g., email and SMS) along with contextual question/answer protocols. Keys established during this pairing process are then used to aid in securely sharing a user's location at a tunable granularity (e.g., GPS). In this paper in section 2, we discuss related work, and In Section 3 we present our framework and implementation for secure location sharing. We present our conclusions in Section 5.

II. RELATED WORK

Google Latitude, Foursquare, Facebook Places, and Glympse are examples of LSSs that operate by having users upload their location data to the service, such that others (i.e., consumers) can access the location data. Our work deviates from prior work by preventing the LSS from viewing a user's location data, and by ensuring that a user has full control over who she chooses to share her location data with and how she intends for her location to be consumed. The protection and secrecy of a user's data contained in the cloud is the focus of Data Locker, which is a collection of tools that enable a provider to encrypt data prior to uploading the data to the cloud. Our model does this precisely with location data, however, our model is not tied to any specific cloud entity. Moreover, we do not generically encrypt location data: policy dictates the precision with which data is presented to the consumers, and how it is protected in the cloud. Instead of protecting data, X-Pire! [2] Attempts to decay data (ostensibly images, but the technique could be applied to location data) by associating a key to an image; when the key expires the X-pire! aware server refuses to serve the data. Similarly, Vanish [8] decays data by altering links to the data stored within a DHT. Although our model does not address the decaying of data, it could benefit from techniques like these in the future.

Several papers present advanced key management protocols that make use of smart phone technology [9,10]. McCune et al. describe the protocol, Seeing-is-Believing (SiB) [10], in which the camera in users' smart phones capture 2D barcodes—these 2D barcodes are used as commitments for exchanging public keys. Our key management protocol relies heavily on the concepts and ideas introduced presented in this work. SafeSlinger is a protocol and framework designed to exchange public keys between smart phones; this is precisely one of the tasks that our key management protocol is designed to accomplish; our work diverges from by using the location-limited channel between pairing smart phones to fully exchange asymmetric keys, and (ii) by leveraging what we refer to as a file-store deposit to assist in symmetric key management. Accelerometer data from two smart phones is used in [9] and to aid in authentication for secure pairing. Mayrhofer et al. [9] employ a strategy of shaking two phone simultaneously to generate a movement limited channel,. Again, our work differs from pairing protocols based on movement limited channels, by operating over location-limited and multichannel communication channels.

Multichannel security protocols, as surveyed in [16], are ways to mitigate against MitM attacks by using multiple communication channels, e.g., radio, visual and 1-bit on/off or toggle buttons, during authentication. The idea is that a malicious eavesdropper cannot eavesdrop on all channels. We use an instantiation of this idea in the variant of our pairing protocol based on historical, multiple open-lines of communication.

LSS have significant privacy issues, and in fact the primary issue was exposed in [5]. In this study, it was shown that those users are uncomfortable with a service controlling access to their location data. Techniques have been introduced to mitigate users' privacy issue concerns, e.g., data can be diffused, or aggregated, but all of these reduce the utility of the data. This is especially troubling if the providers' intentions are for their data to be consumed at a high precision by a specific user, or one or more groups of users. A secondary issue that arose in the study is that many users are uncomfortable knowing that anyone can see their location information. The combination of symmetric and asymmetric cryptography can help address both of

these concerns: i.e., providers encrypt their location data prior to uploading it to a LSS, and then must distribute the decryption key(s) to enable retrieval by authorized consumers. However, this key management process can be a heavy burden. McCune et al. and Farb et al. [13] observe that smart phones are fast becoming ubiquitous and are exceptionally portable and, as such, are usually available during vis-a-vis interactions. This makes Smartphone's an ideal platform for bootstrapping the exchange of cryptographic keys through the location limited channels that can exist between two parties. We further observe that current smart phones possess the technology to perform key management efficiently and fully over location-limited channels, but only lack the protocols and framework to achieve this. SafeSlinger [13] is architected to rely on Internet connectivity from a server to aid in the key exchange protocol (i.e., SafeSlinger only uses the location-limited channel between the pairing smart phones for initializing the key exchange, and then for confirmation). This has the obvious disadvantages of requiring that the server be available at all times, and the exchange occurs in an environment that possesses network connectivity. But both of these requirements are unnecessary to exchange asymmetric keys in a close, vis-a-vis setting that leverages location-limited channels, while using current smart phone technology.

III. **SYSTEM DESIGN**

The SLS system was designed with two main goals in mind enabling tunable and private location sharing with limited contacts, and limiting end-user location over exposure. We now overview our system architecture and describe the threat model within which we expect SLS to be used.

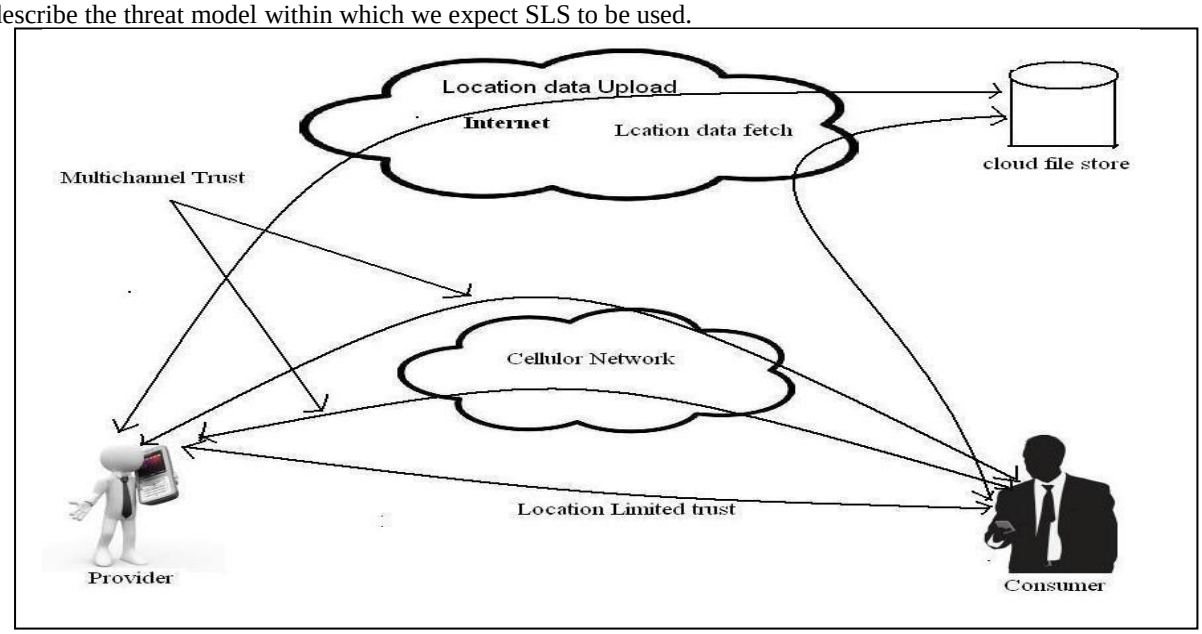

Figure 1: SLS Architecture Overview

Figure 1 presents a high-level view of the SLS system. Users in the system can be divided into two classes: providers and consumers. Providers share their location with others, while consumers retrieve the locations of others; a user can act as both a provider and a consumer. We assume that all users have smartphones, as well as (perhaps self-signed) asymmetric key pairs. Providers' smartphones must be able to detect their current location, e.g., via GPS or cellular/WiFi localization. Secure location sharing is enabled by shared, symmetric keys. The sharing of these symmetric keys is facilitated by asymmetric keys exchanged during a device pairing protocol. SLS provides two pairing protocols to exchange asymmetric keys: one based upon in-person communication over location-limited channels, and another that leverages multi-channel communication for situations in which in-person exchange is not possible. To pair devices using location-limited channels, users' smartphones must have the ability to read and decode QR codes. To pair devices using multiple, historical open-lines of communication, users' smartphones must have both network access (e.g., via WiFi), as well as the ability to send and receive SMS messages. Although the multi-channel based pairing protocol requires that principals have previously communicated, there is no such restriction within the location-limited pairing protocol. Encrypted location data is shared through the use of a SaaS service contracted by the provider. Although one

can currently find SaaS services that are free, our model assumes an associated cost to use the service. We require that the SaaS service allow any user to download data posted to the provider's account.

IV. PROPOSED WORK

References In SLS, a provider's smartphones is responsible for capturing her location data using, e.g., WiFi/cellular localization or GPS. Each location sample collected by SLS is represented as a four-tuple containing a location coordinate, an estimate of the provider's speed of travel, the providers bearing/heading, and a timestamp indicating when the sample was collected. In total, each location sample collected by SLS requires approximately 200 bytes to store. Given that a provider may wish to share her location at multiple granularities, the sample collected by SLS is generalized to each desired granularity. These (perhaps generalized) provider locations are shared with consumers via an (untrusted) SaaS service contracted by the provider. As such, location data must be cryptographically protected prior to upload. To accomplish this, the provider generates one symmetric key for each granularity level at which her data is to be shared, and then CBC encrypts each sample prior to upload. Encrypted location samples are thus unreadable to the SaaS service, with whom the user is under no obligation to share her location(unlike in a traditional LSS). Further, providers also have complete control over the amount of information shared: they may post only a single "current" location (e.g., by overwriting a single location sample), or instead maintain a history of location samples In SLS, we refer to these two operational modes as update and history, respectively. Finally, consumers are under no obligation to create accounts with an LSS, as all data is pulled from SaaS providers by SLS using HTTP GET requests made to world-readable URLs.

Of course, the reliance of SLS on symmetric keys to protect provider location data raises two issues. First, it must be possible for providers and consumers to securely authenticate one another and exchange the cryptographic material needed to retrieve location data at the desired level of granularity. Second, it must be both possible and efficient for the provider to alter the list of consumers with whom she shares information and the granularity at which this information is shared.

A. Location-Limited Pairing

In-person interactions are an ideal setting for device pairing and key exchange. These interactions present the users engaging in the pairing protocol with an opportunity to physically identify the owner of the device with which they are attempting to establish a secure channel, as well as enable the use of local channels to exchange data. The combination of human to- human authentication and device-to-device communication that is difficult to intercept results in demonstrative identification [4] of the participants in location-limited device pairing protocols. SLS utilizes the traits of visual, location-limited channels, and extends the concepts presented in SiB [20] when pairing devices to aid in symmetric key management.

Figure 2 illustrates our location-limited device pairing protocol, which we call Communionable Trust (CT). This protocol makes use of human-to-human audio and visual communication, as well as device-to-device visual communication using on-board cameras and Quick Response (QR) codes. The first step of this protocol is the real-world identification and authentication of the humans who wish to pair devices to facilitate location sharing via SLS. After the human participants have agreed to pair devices, the remainder of the protocol focuses on the exchange of public key information between provider and consumer, and exchanging metadata that enables the sharing of both symmetric keys and location data.

In Step 2 of the protocol, the consumer generates a QR code that contains her public key (K_C) as well as a device identity token (ID_C) used to associate her device with her real-world identity, as managed by the provider's smartphone. Figure 3 shows a QR code containing a 1024-bit RSA public key and it's associated identity token.2 The provider scans this code with his phone, and recovers K_C and ID_C. He then generates a random challenge nonce, N_P , encrypts N_P using K_C, and generates a QR code containing the resulting ciphertext (Step 3). The consumer scans this QR code, decrypts the resulting ciphertext, and verbally communicates then once value to the provider (Step 4). After verifying this exchange, the provider associates K_C and ID_C with the consumer's contact information in his smartphone. This process is then mirrored in Steps 5, 6, and 7 of the communicable trust protocol, which provides the consumer with the providers public key (K_P) and identity token (ID_P). In the final step of the Pairing Phase, the consumer QR-encodes their file-store deposit—a description of an out-of-band channel over which the consumer wishes to be notified of the provider's SaaS store—and presents it for the provider to scan. This message is sent unencrypted due to the location-limited nature of the visual channel used by this protocol.

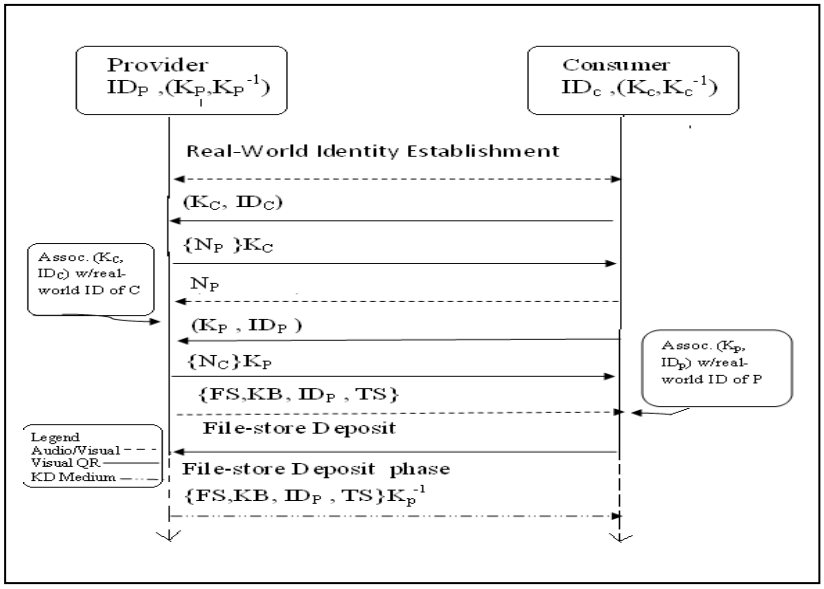

Figure 2: Communionable Trust Protocol

The key bundle URI, KB, provides the consumer with a pointer to an encrypted key bundle stored on the provider's SaaS service. This key bundle is a (key, version, signature) three-tuple that is encrypted using the consumer's public key (K_C). The key field of this tuple contains the current symmetric key corresponding to the precision level with which the consumer is permitted to access the provider's location, the version field indicates the version of this key, and the signature field is a hash of the key and version fields encrypted with provider's private key (K_p^{-1}). Key versions are used to facilitate location retrieval as keys change in response to changes in provider access controls. The level of indirection added by the key bundle—as opposed to directly transferring keys as part of the CT protocol—eliminates the need for direct communication between the provider and consumer upon every policy change. After recovering their key bundle, the consumer can easily retrieve provider locations from the file store URI, FS, and decrypt this data.

B. Multi-channel Pairing

It is unreasonable to assume that users of SLS will always have the ability to physically co-locate during the device pairing process. As such, we also describe a pairing protocol that can be used by individuals who are not within close proximity. As such protocols can be vulnerable to MitM attacks, we make use of multiple historical, open-lines of communication associated with principals on their smart phones (e.g., email address, phone number, or instant messaging account). In this context, historical refers to pre-existing contacts, and open-lines of communication implies that the principals have communicated with the preexisting contact over those multiple channels. This combination of properties gives providers (resp. consumers) higher assurance that the identity of the consumer (resp. provider) is correct, since existing contact information is used to bootstrap the communication process and an active attacker would need to control multiple communication channels to subvert the protocol. Our Historical Communication Channels (HCC) protocol is described in Figure 3

HCC is initiated in the first step of the Pairing Phase by the consumer, who sends their public key (K_C) and device identity token (ID_C) used to associate her device with her real-world identity. (which is established through previous contact as managed by the provider's smartphone). This message is sent to the provider over an existing communication channel (e.g., a known email address), signified using a solid line in Figure 3. Upon receiving K_C and ID_C, the provider generates a random challenge nonce N_P , encrypts N_P using K_C, and sends {N_P }K_C to the consumer via a different historical, open-line of communication that the provider has previously associated with the consumer (e.g., via SMS), which is denoted by a dashed line in Figure 3. The consumer decrypts the ciphertext and returns N_P back to the provider over HCC Secondary (Step 3). At this point in the protocol, the provider is confident that the consumer has access to the private key associated with the public key received in Step 1.

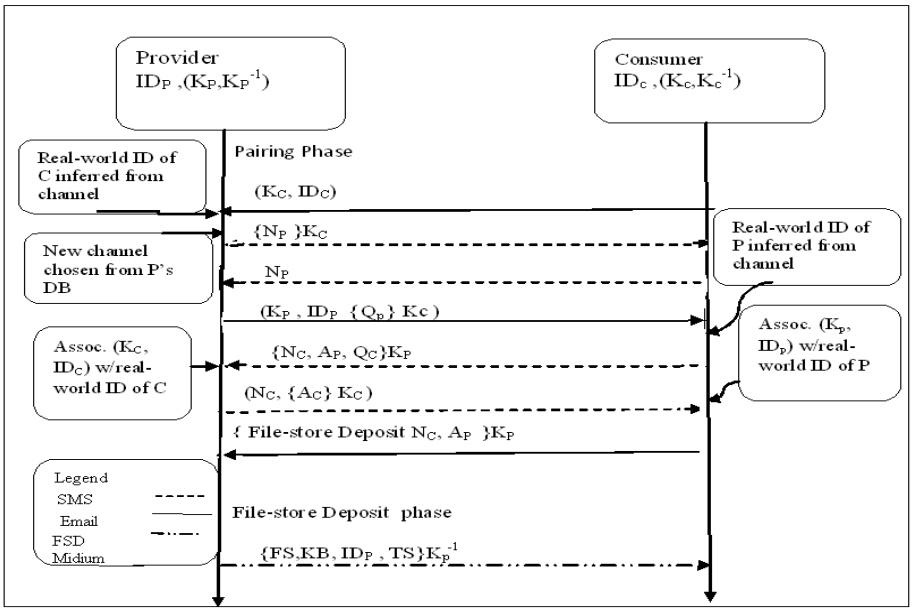

Figure 3: HCC Protocol

However, the provider does not yet have a high level of confidence that the consumer is indeed whom the provider believes they are (e.g., someone other than the consumer could have stolen the consumer's smart phone). Hence, the provider generates a secret question (Q_P) that, within reason, only she and the consumer should know the answer to; e.g., "Who was the away team at the last hockey game that we attended together?". Q_P is encrypted with K_C and is sent to the consumer along with K_P and ID_P over the primary channel (Step 4). The consumer generates (i) the answer (A_P) to Q_P , (ii) her own random nonce N_C, and (iii) her own secret question (Q_C). All three are encrypted with K_P and sent to the provider via HCC Secondary (Step 5). If, after decrypting the resulting ciphertext, A_P is correct the provider sends N_C and her answer (A_C) encrypted with K_C via HCC Secondary (Step 6). After verifying A_C, the consumer encrypts their file-store deposit (File-Store Deposit), N_P and N_C with K_P and sends them via HCC Primary (Step 7). Finally, similar to the CT protocol, the provider assigns the consumer's precision level and the File-store Deposit Phase begins. We note that the HCC protocol should be terminated if either (i) a principal receives a secret question via SLS without first initiating or receiving a public key from the same principal over a different channel, or (ii) the response to a participant's secret question is incorrect.

C. Policy Control

After learning the consumer's public key and file-store deposit (either through CT or HCC), the provider must associate the consumer with the precision level at which they are authorized to view the provider's data. All consumers that are assigned the same precision level by a provider are considered to be in the same group and, thus, all have access to a single symmetric key protecting location disclosures made at this precision level. As a result, the symmetric key associated with a particular precision level may need to be updated as the group of users who have access to that precision level changes over time. To provide the highest level of security for the provider's location data—i.e., preserving forward- and backward-secrecy—these shared symmetric keys should be changed whenever a consumer is added to a precision group or removed from a precision group. The former case ensures that new consumers cannot access old data, while the latter ensures that former consumers cannot access new data. Altering the symmetric key for a particular precision level requires creating a new symmetric key, encrypting key bundles for each user authorized at this precision level, and depositing the bundles on the provider's file store. After asynchronously retrieving these new key bundles, authorized consumers can again access the providers data. While shared symmetric key update is non-trivial, our evaluation shows that the overheads associated with this process in practice are minimal. We note that it is not necessary for the provider to re-pair their device with consumers via CT or HCC, as the asymmetric keys use for key management are not affected by a consumer's change in precision level.

We recognize that our LSS model prevents the enforcement of certain policies found in existing LSSs; e.g., policies that enable location sharing only when two parties are within a certain physical proximity, or policies that place access count limits on individual users. LSSs that can implement proximity-based policies are able to do so because they have access to the location data of all their users, and can thus determine the distance between two users. Since our goal is to prevent the LSS from acquiring this information, this type of policy cannot easily be enforced in SLS. Enforcing constraints on access frequency is also enabled via LSS intervention, which is contrary to our assumed sharing model.

V. CONCLUSION

Location-sharing systems (LSSs) have high utility, but recent research has shown that many users are wary of sharing detailed trace information with LSS operators and the large social networks with which LSSs are often integrated make it all too easy to accidentally share location data with unintended audiences. In this seminar, we make solution to the above problems by developing SLS, a framework for private location sharing that combines the use of social authentication protocols based upon location-limited or multi-channel protocols, to facilitate secure data storage and location sharing. In particular, our device pairing protocols leverage the smaller social networks managed by user smartphones to provide a high degree of assurance in the identities of the individuals with which sharing is to occur. The asymmetric keys exchanged during this process can then be used to distribute shared symmetric keys that protect a location provider's sensitive location information from unauthorized viewers, including the cloud service used to host the data.

ACKNOWLEDGMENT

Authors like to thanks all the people who directly and indirectly help me for this paper. This work is supported by Department of computer science and engineering, G. H. Raisoni College of engineering, Amravati, India.

REFERENCES

[1] Andrew K. Adamsyz and Adam J. Lee "Combining Social Authentication and Untrusted Clouds for Private Location Sharing" *SACMAT*'13, June 12–14, 2013.

[2] J. Backes, M. Backes, M. Dürmuth, S. Gerling, and S. Lorenz."X-pire! - a digital expiration date for images insocial networks." CoRR, abs/1112.2649, 2011.

[3] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. "Talking To Strange Authentication in Ad-Hoc WirelessNetworks." *NDSS*, 2002.

[4] L. Bauer, L.F. Cranor, R. W. Reeder, M. K. Reiter, and K. Vaniea. "Real life challenges in access-control management. "In CHI 2009: Conference on Human Factors in Computing Systems, pages 899–908, April 2009.

[5] J. T. Biehl, E. G. Rieffel, and A. J. Lee. "When privacy and utility are in harmony: towards better design of presence technologies." Personal and Ubiquitous Computing, 17(3):503–518,2013.

[6] Y. Cai and T. Xu."Design, analysis, and implementation of a large-scale real-time location based information sharing system. Proceeding of the 6th international conference on Mobile systems applications and services MobiSys 08", page 106, 2008.

[7] D. Dolev and A. C. Yao."On the security of public key protocols."In Proceedings of the 22nd Annual Symposium on Foundations of Computer Science, SFCS '81, pages 350–357, Washington, DC, USA, 1981. IEEE Computer Society.

[8] S.Egelman, A. Oates, and S. Krishnamurthi."mitigating repeated access control errors on facebook." In CHI 2011: Conference on Human Factors in Computing Systems, pages 2295– 2304, 2011.

[9] R. Geambasu, T. Kohno, A. Levy, and H. M. Levy. "Vanish: Increasing data privacy with self-destructing data. In Proc. Of the 18th USENIX Security Symposium, "2009

[10] R. Mayrhofer and H. Gellersen." Shake Well Before Use: Intuitive and Secure Pairing Mobile Devices." IEEE Transactions on Mobile Computing, 8(6):792–806, June 2009

[11] J. McCune, A. Perrig, and M. Reiter." Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication" In IEEE Symposium on Security and Privacy, pages 110–124. IEEE, 2005.

[12] B. Palanisamy and L. Liu. "Mobimix: Protecting location privacy with mix-zones over road networks. Data "Engineering, International Conference on, 0:494–505, 2011.

[13] S. Patil, G. Norcie, A. Kapadia, and A. J. Lee. Reasons rewards," regrets: Privacy considerations in location sharing as an interactive practice." In Symposium on Usable Privacy and Security (SOUPS), July 2011

[14] R. W. Reeder, L. Bauer, L. F. Cranor, M. K. Reiter, K. Bacon, K. How, and H. Strong"Expandable grids for visualizing and authoring computer security policies." In CHI Conference on Human Factors in Computing Systems, pages 1473–1482, 2008.

[15] R. Schlegel, A. Kapadia, and A. J. Lee. "Eyeing your exposure: quantifying and controlling information sharing for improved privacy." In Proceedings of the Seventh Symposium on Usable Privacy and Security, SOUPS '11, 2011

[16] Y. Wang, S. Komanduri, P. Leon, G. Norcie, A. Acquisti, and L. Cranor. A qualitative study of regrets on facebook "In Symposium on Usable Privacy and Security (SOUPS), 2011.

[17] F. L. Wong and F. Stajano. Related Work in Multichannel Security Protocols. IEEE Pervasive Computing,31–39,2007.

