A Secure Channel Using Social Messaging for Distributed Low-Entropy Steganography

School of Computer Science and Mathematics

Kingston University

Kingston upon Thames, London, United Kingdom

Eckhard Pfluegel (corresponding author, E.Pfluegel@kingston.ac.uk), Charles A. Clarke, Joakim G. Randulff, Dimitris Tsaptsinos, James Orwell

Abstract

Free, confidential and uncensored communication between two individuals is an essential requirement for the modern, digital age. Social communication and interaction between users of Online Social Networks (OSNs) enjoys certain security properties, but true end-to-end security cannot be achieved. In this paper, we devise a secure channel between two OSN Friends using social messaging. Confidentiality is achieved through undetectable communication based on distributed low-entropy steganography, thus implementing a security control for message content vis-à-vis an untrusted OSN provider. A prototype solution, in the form of an Android mobile app, is implemented. This uses two channels of social messaging (Twitter and Google+) to demonstrate the feasibility of the proposed approach. The source code is publicly available, thus making it suitable for use by individuals (and organisations) as a cost-free, secure communication tool.

1. Introduction

Free confidential and uncensored communication between two individuals is an essential requirement for the modern, digital age. Advanced and essentially free communication and information sharing functionality is nowadays readily provided by Online Social Networks (OSNs) and used by billions of individuals every day. However, true end-to-end security cannot be hoped for from this form of communication channel as most OSN terms of use and security practices actually pose confidentiality threats, effectively classifying the OSN provider as untrusted. Using formal security terminology, the OSN provider is referred to as honest-but-curious.

Being concerned about the security of their critical information assets, OSN users will face a range of adversaries, posing as potential threat actors, including the OSN provider. In order to improve this situation, existing OSN Security research has primarily focused on devising controls, ensuring that OSN users can rely on the security goals of anonymity and confidentiality when managing their User Profiles (UPs) and User Generated Content (UGC). The appearance of social messaging applications, which are extremely popular especially in the form of mobile clients, built as front-ends to OSN platforms, introduces a new dimension of communication and the attached security issues are directly related to many aspects of OSN Security.

At this point, it is helpful to introduce two recent proposals for OSN security controls that have been suggested in the literature, which influential for our research. (These are covered in more detail, in the literature review in Section 2). They were devised with confidentiality and anonymity in mind, and also taking into account that all secret communication that is routed through a centralised OSN needs to be *undetectable*, even if routinely filtered through statistical or semantic analysis – c.f. the terminology of a socially constrained channel in Section 1.1).

Firstly, confidential communication techniques for untrusted OSNs have been proposed [3], and in particular using low-entropy steganography. This implements a security control against threats to UGC vis-à-vis an untrusted OSN provider.

Secondly, Virtual Private Social Networks (VPSNs) have been introduced [8] and further developed [9] as a powerful conceptual framework for leveraging existing OSN infrastructure to provide UP anonymity, while still providing the full functionality of the OSN. The VPSN architecture fits traditional, web-based OSN platforms.

The research in this paper extends [3] by using multiple OSNs, and moving focus away from traditional OSN platforms to mobile architectures. The VPSN approach could be seen as complementary to our solution, especially our previous work [5, 6]. This aspect is further discussed in the conclusion section of this paper.

1.1 Outline of Proposed Method

The aim is for two OSN users, Alice and Bob, to exchange a secure message by using social messaging. This main requirement is that the content of the message is not able to be deciphered by any third party, including the OSN provider (e.g. Facebook, Twitter, Google+), which could hypothetically take the

role of the 'man-in-the-middle' in an attack of the same name. An additional requirement is that the message is undetected, in the sense that it is difficult for any third party to identify all and only those elements of communication necessary to decrypt the message. To summarise, Alice and Bob need end-to-end security, which is not normally implemented in these scenarios.

The standard approach to implement message security is based on encryption. Regardless of whether symmetric or asymmetric encryption is used, key exchange is a critical task that needs to take place 'out-of-band' before the actual encryption process. When attempting to apply this to our scenario, Alice could send a secret key to Bob using social messaging, and she could then transmit an encrypted message using a different social messaging channel. This would be secure under the assumption that two different OSN providers do not co-operate. The problem with this approach is that the presence of a ciphertext in the social messaging channel will stand out from "normal" plain text conversations, using natural language. This could potentially raise suspicion and would draw attention to Alice and Bob in the presence of traffic analysis.

Thus, under these circumstances the, social messaging can be regarded as a *socially constrained channel* in which messages containing deviations from natural language patterns will be noticed and hence need to be avoided. The OSN provider may eavesdrop on all communication, and in particular any "unusual" content such as encryption (if tolerated by the social network) or artificial messages used for steganography will also raise suspicion.

The previously proposed solution [3] uses the socially constrained channel as a side-channel in order to send an inconspicuous cover message text *t*, which to any third party would be regarded as a reasonable and genuine communication. A second channel can be characterised as socially *un*constrained. This can be used to transmit the actual ciphertext, obtained by encrypting the plaintext message with a symmetric key, which is obtained by using *t* as a seed, e.g. by applying a secure hash function. This protocol is in effect using information hiding (referred to as low-entropy steganography). For this to work, Alice and Bob need access to each other's OSN account names, and the cryptographic parameters used such as the specific cipher and hash algorithm.

It is worth noting that, both VPSN [8] and undetectable communication architectures [3] are instances of the two-channel communication genus – the first communication channel being a socially constrained OSN channel, the second socially unconstrained channel a TCP/IP connection to an XMPP server [8], or an external file server such as Dropbox [3].

Below, these approaches are generalised, to propose a protocol making use of one 'regular' (socially unconstrained) channel and *n* socially constrained channels. This proposal has a number of benefits that are explored in Section 3.

1.2 Research Contributions

The contribution of this paper is twofold, comprising a theoretical result and its practical evaluation through the implementation of a prototype solution.

The first contribution is the proposal for a secure channel, achieving confidentiality through undetectable communication based on distributed low-entropy steganography using social messaging.

The second contribution is the development and release of a prototype implementation, in the form of an Android mobile app, using on communication using multi-channel mobile instant messaging through Twitter and Gmail.

1.3 Paper Organisation

This paper is organised as follows: in Section 2, the relevant previous work is reviewed, and build on these ideas to shape the research question central to this paper. In Section 3, the proposed system architecture is presented, followed by a description and evaluation of our implementation in Section 4. Some concluding remarks are provided in Section 6.

2. Previous Work

The proposed approach builds on several key papers [1, 3, 8, 9] in the field. In this section, after a general discussion of OSN security issues, the main results of these papers are reviewed, and an indication is given of the direction of the follow-up research [5], which precurses the work described in this paper.

There are a wide range of security and privacy threats to OSN-related assets, as illustrated in the this section. From the point of view of OSN providers, they need to protect user data assets against attacks from both external attackers and insiders (other OSN users). Examples of OSN-internal (user) attacks are:

- False account registrations – providing wrong personal information (such as name, town, gender).
- Identity masquerading – malicious theft and consequent misuse of another OSN user's identity to commit a crime.
- Use of their platforms for illegal purposes – the use of advanced OSN communication functionalities for discussing, planning or executing criminal activities.
- Malware – malicious software is infiltrated into the OSN system, for potential downloading by other users.

Typical OSN-external attacks are:

- Web application attacks – the OSN's web interface is targeted by typical attacks such as Denial of Service (DoS), Buffer Overflow, Cross Site Scripting (XSS) or SQL Injection.
- Data Breach – user-related information is disclosed to an external party, compromising the system.
- Account Compromising ("hacking") – user account information or any other user data is disclosed to unauthorised parties.

We can assume that the OSN provider provides integrity, both in terms of data and origin integrity. However, as already mentioned, other OSN users might betray their peers through various attacks. Even more crucially, the OSN Provider has access to all UGC and user account data and is able to intercept and potentially modify all messages sent between users.

The main OSN-provider threats to UGC are as follows:

- Data Exploitation – an OSN host may impose the right to use UGC for commercial or marketing purposes, without the need to consult, or compensate the user.
- Data Censorship – an OSN host may impose the right to modify or remove UGC for reasons of censorship or violation of terms and conditions.
- Data Sanitisation – OSN hosts may sanitise user data prior to publication, in order to protect themselves and other users from malware.

Finally, digital surveillance is reaching a worrying scale, and the Internet Service Provider (ISP) needs to be added to the list of adversaries. The ISP adversary is as capable as the OSN Provider adversary, but additionally, he can monitor the Internet traffic between any OSN users.

A number of cryptographic techniques have been suggested in the literature to address the security goals of confidentiality, integrity and availability in the presence of various adversary types. Encryption was suggested in early work [1, 10, 12, 14, 15, 16, 19]. Steganography, proposed in [2, 3, 13], was recognised as being a more suitable technique due to its hiding properties.

Security controls should be transparent to the OSN host and users, and should not impair the performance of the OSN. These additional requirements lead to the idea of implementing security based on devising a novel OSN system architectures.

Decentralisation prevents UGC to be stored and controlled on a centralised server [21, 4] but this has proven to be difficult to implement.

The work described in the remainder of this section is based on two-channel communication architectures (or more generally, n channels as employed by this paper), which is a promising approach, combining desirable security and usability features with ease of implementation.

However, in all of these approaches, the potential threat of traffic analysis is a problem. Traffic analysis refers to a process of message or data intercepting, from which pattern based information (e.g. trends in communication times, data size etc.) can be deduced, even if the messages or data are encrypted.

2.1 UP Anonymity

In [8], the authors conceptualise the notion of a VPSN, in which the entities of a traditional Virtual Private Network (VPN), are mapped to elements of an OSN. The OSN is deemed analogous to a public untrusted infrastructure and the users are analogous to network devices. The objective is to implement a means of secure communication between users, via the OSN, that delivers the security goals of a VPN, hence the term VPSN.

A VPSN enjoys a great number of properties, and an ideal implementation would result in a system offering the full feature set of the underlying OSN, as well as additional security. The essential VPSN characteristics as introduced in [8] can be summarised as follows:

- A VPSN is virtual, since it utilises the functionality of an existing OSN and is by nature an OSN itself. In particular, a VPSN inherits security mechanisms from the OSN. These include authenticated access and authenticated HTTP connections between users. However, being a private entity, user profile information can be hidden from any non-intended audience (crucially, this includes the OSN provider).
- A VPSN is hidden to users that are not part of it, as well as the OSN provider. In addition, it is transparent to its users and hence provides the desirable combination of security and usability.
- A VPSN solution should allow members to connect to multiple VPSNs [or OSNs???]. The impact of a VPSN on the OSN's performance should be negligible and VPSN users should still have access to the full functionality of the OSN.

VPSN research addresses a central problem of how to make the public facing elements of a Facebook user profile, private to all but intended friendship groups. The solution relies on users creating Facebook accounts, with fake user profile credentials. By default, the publicly accessible profile information displays the fake credentials. True profile values are revealed to legitimate VPSN members via privately shared XML lookup tables that render dynamically when the user profile is visited. This idea is implemented as a Firefox plugin called FaceVPSN as further detailed in [9].

Virtual Private Social Networks were primarily introduced with user profile privacy and anonymity in mind. This was achieved by restricting the availability of profile data. A benefit of traditional VPNs when operating in so-called tunnelling mode is that they hide the endpoints of the communication link from eavesdroppers. It can be noted that user profile anonymity in a VPSN could be seen as an analogy of the existence of hidden endpoints of a VPN tunnel between nodes in the network.

To our knowledge, the only VPSN implementation described in the literature is given in [9], where the authors describe FaceVPSN, implemented as a Firefox web browser plug-in. An alternative VPSN architecture was given in [5], although the contribution remained conceptual.

2.2 UGC Confidentiality

The idea of [1] and later [3] is to disguise UGC through a specific form of steganography, which makes it "socially indistinguishable" and hence difficult to detect (under certain assumptions). The paper gives a formal definition of communication models, enabling undetectable communication in OSN, based on two schemes: a high-entropy model using traditional steganography techniques for cover modification, selection or synthesis and the low-entropy model, based on specialised techniques for the usage in OSNs.

The first scheme adopts a conventional theme of hiding a secret payload within a cover object, which is then simply transmitted as carrier object by the sender, via the OSN to a recipient. This could involve a

secret key, in order to enhance the security of the scheme, which needs to be exchanged out-of-band before the message can be understood by the recipient.

The second scheme is described as an information sharing scheme that is provably undetectable within the OSN. In this scheme, it is assumed that a prospective cover-object has no capacity for embedding a secret as a payload (e.g. if an OSN such as Twitter applies a constraint that limits the capacity of the cover-object). Therefore, rather than containing a payload, the cover-object (which may be a keyword), represents the location of where a secret can be accessed. Prior to exchanging information via this scheme, a sender and receiver pre-agreed keywords that are mapped to online locations that are hidden by way of a URL shortening service. In this approach, the pre-shared keywords that are mapped to a shortened URL, represent a stego-key. Conceptually, when a recipient receives a particular keyword in a message posted to the OSN by the sender, they match the keyword to the pre-agreed location, which they visit to access the secret.

The general concept of how the scheme works is as follows:

1. The sender visits a storage location to deposit a secret. For each recipient a separate folder is created, in which a separate copy of the secret is stored.
2. The sender uploads the storage URL and keyword mapping index to the URL mapping service, and subsequently posts an appropriate keyword to the OSN.
3. A participating recipient accesses the innocuous keyword, derives the mapping index and visits the URL mapping service to derive the URL of the storage location for the secret.
4. The recipient visits the storage location to access the secret.

The example presented is based on the use of Facebook as the OSN, TinyURL as the URL shortening service and Dropbox as the storage server. It is assumed that links between all nodes are secured via HTTPS and that encryption keys and stego-keys are exchanged via secure OOB channels. The secret is ultimately protected by the use of encryption.

2.3 Distributed High-Entropy Steganography Approach

It is seen that both reviewed approaches use a two-channel communication architecture – the first communication channel being the OSN channel, the second channel a TCP/IP connection to a XMPP server in the case of FaceVPSN, and an external file server such as Dropbox in [3].

Revisiting the notion of socially constrained and unconstrained communication channels, we identify the OSN channel as the socially constrained, and the second channel as the socially unconstrained channel. The OSN provider may eavesdrop on all communication, and in particular any "unusual" content such as encryption (if tolerated by the social network) or artificial messages used for steganography will raise suspicion.

An alternative architecture based on a distributed communication protocol using *n* channels has been proposed in [5]. This approach combines steganography with another fundamental cryptographic technique: secret sharing. The idea of secret sharing is to divide given data (the secret *s*) into *n* parts (the shares) in such a way that knowing at least *m* shares allows for reconstructing *s*. In an ideal secret

sharing scheme, knowledge of less than *m* shares will not reveal any information on *s*. A secret sharing scheme with parameters *m* and *n* satisfying the aforementioned properties is also called a (*m*, *n*) threshold scheme. A popular scheme is based on polynomial interpolation, introduced by Shamir [18].

The approach of [5] effectively yields a high-entropy steganographic technique with additional security and robustness due to the nature of the secret sharing scheme. The approach can be described as follows. First, an (*m*, *n*) secret sharing scheme is applied to the plaintext message, splitting it into *n* shares. High-entropy steganography is used to hide the individual shares in a suitably crafted carrier-medium, thus creating a stego-medium. Each stego-medium (which to the OSN takes the form of generic and ordinary UGC) is then sent on the individual OSN channel.

The recipient can access the original message by retrieving a subset of *m* messages, extracting the shares from the stego-medium, and combining them in order to reconstruct the message.

Apart from the need to carry out steganalysis in order to detect individual secret payloads, the correct combination of shares would have to be identified using brute-force searching across different users and different messages per user. In addition, the particular secret sharing scheme would have to be known in order to reconstruct the secret message m from the shares. These tasks seem intractable when considering the potential number of OSNs, users, messages, steganographic and secret sharing scheme combinations, which informally justifies the claim of additional security.

3. Proposed Architecture

The key contribution of this paper is given in this section. It consists of a cryptographic protocol and an OSN system architecture.

The proposed protocol uses a suitably adapted secret sharing scheme with random shares, inspired by the scheme presented in [22]. The provided high-level description hides the complex details that would be required for the specification of a robust real-world security protocol. As already explained in Section 2, this is a low-entropy steganography approach as introduced in [3], with improved payload capacity compared to [5].

The system uses a distributed architecture (see below for a more detailed discussion) inspired by [5], in turn improving on the two-channel approach of [3] by achieving additional security and redundancy.

In this solution, undetectable communication is achieved by using *n* socially constrained and one additional, socially unconstrained channel.

In order for the proposed scheme to be operational and secure, some additional assumptions are required, as follows. Alice and Bob have accounts with *n* OSNs, on each of which they are mutual friends. They also have (joint) access to an external file server. The OSN authentication mechanism is trusted. This is a reasonable assumption, as a compromise would threaten trust in the OSN. The individual OSNs do not co-operate, i.e. they respect the confidentiality of user data with regards to external entities. We again deem this expectation to be reasonable.

In order to send a secret message m, Alice and Bob proceed as follows – it is assumed that both Alice and Bob have previously exchanged a secret key K_{AB} and suitable parameters g and p where p is a large prime number and g a primitive root modulo p. A secure hash function H is also required. The integers k and n are as in the previous section. Figure 1 provides a diagrammatic overview.

1. Alice chooses n short texts $t_1,..,t_n$. She then sends t_i to Bob, using OSN_i.
2. She also chooses a random polynomial $f(x)$ of degree k, satisfying $f(0) \neq 0$. The session key is then $K = H(K_{AB}, f(0))$.
3. Furthermore, she computes $h_i = H(t_i)$ and $g_i = g^{hi} \bmod p$. She finally sends $c = E_K(m)$, $[g_1,.., g_n]$ and $[f(h_1),.., f(h_n)]$ using the socially unconstrained channel.

Message reconstruction then works as follows:

1. Bob selects k out of the n OSNs and reads his messages, hence obtains a set of k short texts t_{ij} $(j=1,..,k)$. He computes the corresponding h_{ij} and g_{ij}.
2. Listening on the socially unconstrained communications channel, he receives c and the additional data. He uses this to identify the corresponding $f(h_{ij})$.
3. Using polynomial interpolation, he constructs $K = H(K_{AB}, f(0))$. Finally, he obtains the original message by decrypting c.

We observe that for $k=n=1$, we will obtain the undetectable communication scheme of [3]. If $n>= k>1$, two additional benefits arise from our scheme:

- The value $n-k$ measures the robustness against the unavailability of OSNs. In particular, this implies a resistance against an account closure attack of a leveraged OSN, potentially in response to suspicious user activity.
- The value k is a threshold value for the number of OSNs that would have to cooperate in order to combine individual shares of the short text t. Any number of OSNs less than this value can provably

not gain any information about *t*, due to the properties of the secret sharing scheme.

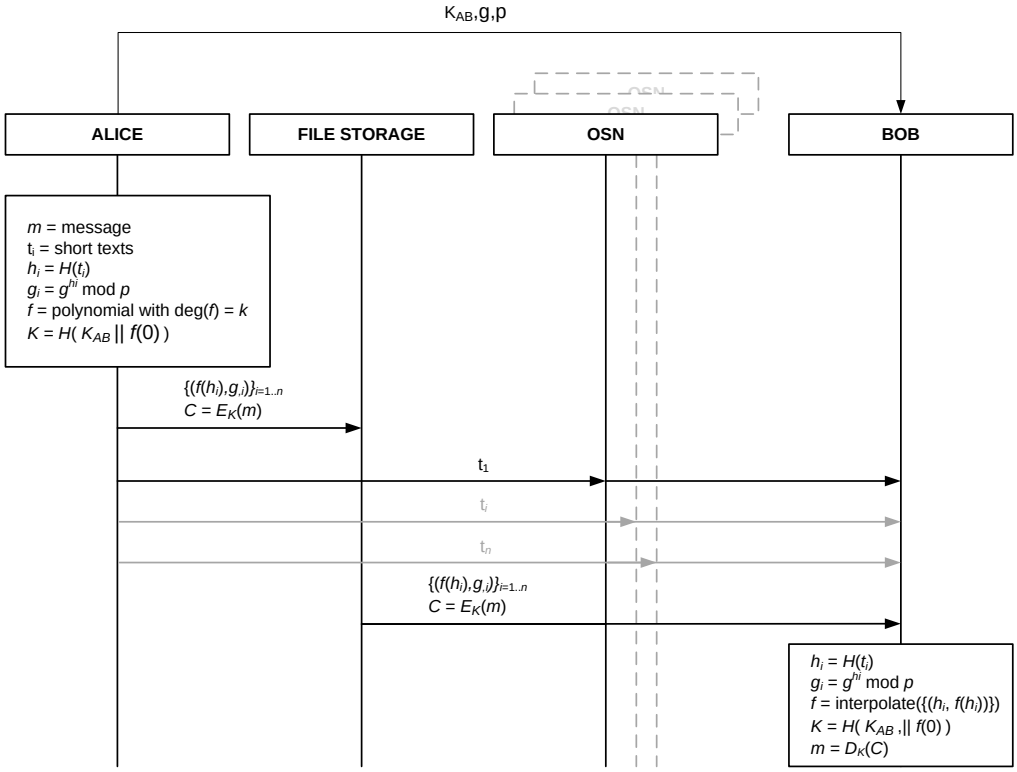

Figure 1 – Distributed architecture for undetectable communication

4. Implementation

In this section, the demonstration proof-of-concept is described.

The system is realised as an Android mobile app, implemented in the Java programming language. Running within the Android operating system, the application uses the application programming interfaces (API) of the OSNs in order to access the needed communication functionality. Implementing the solution in the Android operating system (OS) is a choice that improves security, in part due to the fact that Android is an Open Source development platform. Because of Google's focus on privacy, the Android OS includes several security controls related to protecting applications and user data, and the fact that the OS itself is built on the well-known Linux kernel is one of its strongest security features. The openness of Android also makes it easier to develop applications, and by utilising the built-in permissions feature, the users of the application can themselves examine what type of data the system will be able to access.

Currently, the implementation uses Twitter and Gmail, but it would be possible to extend the number of available channels by adding additional APIs. Using the APIs of these OSNs is a big advantage in terms of security, as it helps a developer implement potentially complicated authentication functionality with fairly simple code. All the logic related to network protocols, HTTPS and authentication has already been

designed, implemented and soundly tested by the OSNs themselves. An open source library is used to implement the secret sharing functionality, which is based on Shamir's Secret Sharing scheme. This library has been reviewed by cryptographers, who have evaluated it as a secure solution. The mobile application is implemented as an Open Source project and the source code is publicly available on the GitHub website [17].

The basic data flow of the application is as follows:

1. Prior to exchanging secret messages, both users need to have an account with Gmail and Twitter, and they also need to know each other's Twitter username and Gmail address. Furthermore, they need to follow each other on Twitter, which is due to the way Twitter is built, where users can only privately message someone who they already follow.
2. The user logs into his Gmail and Twitter account using his own credentials. He then writes a message string, and specifies the Twitter username and Gmail address of the recipient. The system then applies secret sharing to the message string, thus creating two different shares. The shares are then augmented by a unique suffix, which consists of an application-specific identifier and the globally unique ID (GUID) of the Android device. This suffix is added in order to make it possible for the application to identify shares when it is looking for incoming messages. These augmented shares are each distributed using the OSN channels, with one share being sent as an email, and the other being sent as a private Twitter message.
3. The recipient, who is logged in with his own Gmail and Twitter credentials, then tells his application to look for a message. The application accesses the Gmail and Twitter inbox of the user, and looks for the augmented shares, by looking for the application-wide keyword that was added to the share suffix before sending. After identifying the potentially multiple shares the application uses the Android GUID to identify shares coming from the same sender. By then comparing the date and time of receiving a message the application can identify shares that belong together. When they are found, the application can join the shares to recreate the original message string from the other user.

The following figures depict the Graphical User Interface of the mobile app, offering the user a choice between sending and receiving a message. Sending a secret message only requires entering the text and recipient addresses, if the user has previously configured his Gmail and Twitter account details. The secret message is received by the two communication channels and automatically reassembled to obtain the plaintext message.

The application has been evaluated in terms of its computational overhead and no significant delays occur when sending messages of a reasonably small size compared to using a standard messaging app. Currently, the message size in this tool is limited to a small number of characters, due to limitations imposed by the Twitter API – this is unfortunately less than 140, the number of characters in a Twitter tweet. The downtime of global OSNs such as Gmail and Twitter is typically limited to a couple of hours per year, and there are thus no real danger of availability issues for this application. This solution is currently only using two communication channels, and the security could be improved substantially by using other OSNs as well. Using a third channel simply for redundancy reasons could be a valuable

improvement.

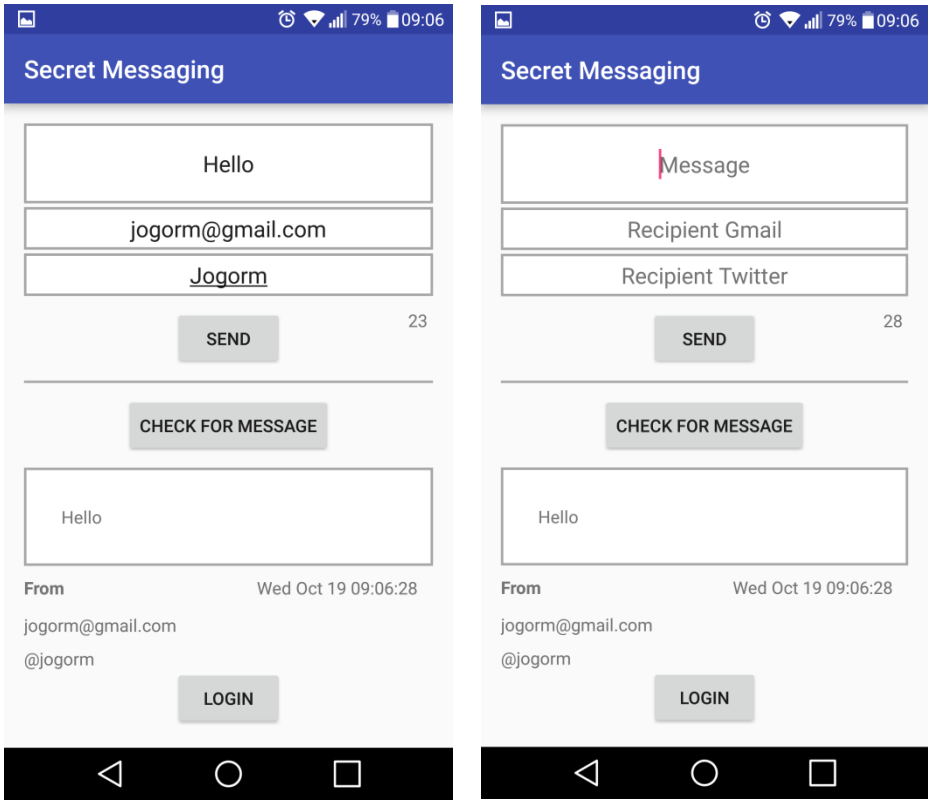

Figure 2 – Sending and Receiving a Message

5. Conclusion

This paper describes the design of a secure channel, based on social messaging, secret sharing and the use of multiple communication channels for message transmission. It also describes the implementation of an social messaging Android app using the Twitter and Gmail APIs as a proof-of-concept solution, thus moving focus away from traditional OSN platforms to mobile architectures.

It would be relatively straightforward to enhance this implementation to yield a fully usable, robust system with richer functionality. For example, a further enhancement could be provided by the automatic suggestion of cover messages that fit well with the online personae and message history. Indeed, plausible message histories could themselves be automatically generated to better camouflage the cover message.

To date, OSN security architectures and systems do not seem to have attracted a lot of attention for commercial exploitation, despite their readiness for robust implementations and the resulting security benefits. A range of issues might be underlying reasons:

Implementing a truly robust and user-friendly solution would require a substantial effort. The existing research prototype implementations demonstrate proof of concept but are lacking of features that a widely accepted and adopted tool would need.

Some of the systems, depending on itheir architecture, may violate the terms and conditions of the leveraged OSNs. For example, FaceVPSN achieves UP anonymity by using fake user profiles. In [11], Facebook report that the total percentage of "fake" or duplicate user accounts that violate their terms and conditions, is estimated to be 7% (116 million users out of a total 1.65 billion monthly active users). Whilst it might be impractical for an OSN to detect such fake accounts, it might be difficult for a commercial VPSN solution to become viable and remain legally unchallenged.

There has been a change in use patterns, with users increasingly relying on mobile devices and the use of mobile apps for social networking, in particular mobile instant messaging apps are one of the most heavily used mobile apps [20]. Clearly, there is a need for more research in this area, in particular in the design of mobile-based security solutions.

We argue that OSN security architectures are a powerful concept — yet to be discovered for future use by individuals and organisations alike, and that this work is a step in this direction.

References

[1] Beato, F., Kohlweiss, M., & Wouters, K. (2011). Scramble! your social network data. In Privacy Enhancing Technologies (PETS) (pp. 211–225). https://doi.org/10.1007/978-3-642-22263-4_12

[2] Filipe Beato, Iulia Ion, Srdjan Čapkun, Bart Preneel, and Marc Langheinrich. 2013. For some eyes only: protecting online information sharing. In Proceedings of the third ACM conference on Data and application security and privacy (CODASPY '13). ACM, New York, NY, USA, 1-12. DOI=http://dx.doi.org/10.1145/2435349.2435351

[3] Beato, Filipe, Emiliano De Cristofaro, and Kasper B. Rasmussen. "Undetectable communication: The online social networks case." Privacy, Security and Trust (PST), 2014 Twelfth Annual International Conference on. IEEE, 2014.

[4] Buchegger, S., & Schi, D. (2009). PeerSoN : P2P Social Networking — Early Experiences and Insights. In Proceedings of the Second ACM EuroSys Workshop on Social Network Systems (pp. 46–52). ACM. https://doi.org/10.1145/1578002.1578010

[5] C. A. Clarke, E. Pfluegel and D. Tsaptsinos, "Confidential Communication Techniques for Virtual Private Social Networks", DCABES 2013: The 12th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, Kingston University, Sep 2-4 2013.

[6] C. A. Clarke, E. Pfluegel and D. Tsaptsinos, "Enhanced Virtual Private Social Networks: Implementing User Content Confidentiality" in ICITST 2013: The 8th International Conference for Internet Technology and Secured Transactions, London, UK, Dec 9-12 2013.

[7] Clarke, Charles A., Eckhard Pfluegel, and Dimitris Tsaptsinos. "Multi-channel overlay protocols: Implementing ad-hoc message authentication in social media platforms." Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), 2015 International Conference on. IEEE, 2015.

[8] Conti, Mauro, Arbnor Hasani, and Bruno Crispo. "Virtual private social networks." Proceedings of the first ACM conference on Data and application security and privacy. ACM, 2011.

[9] Conti, Mauro, Arbnor Hasani, and Bruno Crispo. "Virtual private social networks and a facebook implementation." ACM Transactions on the Web (TWEB) 7.3 (2013): 14.

[10] De Cristofaro, E., Soriente, C., Tsudik, G., & Williams, A. (2012). Hummingbird: Privacy at the time of Twitter. In Proceedings - IEEE Symposium on Security and Privacy. https://doi.org/10.1109/SP.2012.26

[11] Facebook Inc., “Facebook - Financials - SEC Filings,” fb.com, 2016. [Online]. Available: https://investor.fb.com/financials/sec-filings-details/default.aspx?FilingId=11342580. [Accessed: 20-October-2016].

[12] Feldman, A., & Blankstein, A. (2012). Social networking with frientegrity: privacy and integrity with an untrusted provider. In Proceedings of the 21st USENIX conference on Security symposium (Vol. 12, pp. 31–31). Retrieved from https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final67.pdf

[13] Ion, I., Beato, F., Capkun, S., Preneel, B., & Langheinrich, M. (2013). For some eyes only: Protecting online information sharing. In CODASPY 2013 - Proceedings of the 3rd ACM Conference on Data and Application Security and Privacy (pp. 1–12). https://doi.org/10.1145/2435349.2435351

[14] Lucas, M., & Borisov, N. (2008). FlyByNight: Mitigating the Privacy Risks of Social Networking. In Proceedings of the Seventh ACM Workshop on Privacy in the Electronic Society (pp. 1–8). https://doi.org/10.1145/1456403.1456405

[15] Luo, W., Xie, Q., & Hengartner, U. (2009). FaceCloak: An Architecture for User Privacy on Social Networking Sites. In International Conference on Computational Science and Engineering (Vol. 3, pp. 26–33). https://doi.org/10.1109/CSE.2009.387

[16] Malik, S., & Sardana, A. (2011). Secure Vault: A privacy preserving reliable architecture for Secure Social Networking. In Proceedings of the 2011 7th International Conference on Information Assurance and Security, IAS 2011 (pp. 116–121). https://doi.org/10.1109/ISIAS.2011.6122805

[17] "Secretmessaging Open Source Project", github, https://github.com/jogorm/Secretmessaging [Accessed: 20-October-2016].

[18] Shamir, A. (1979). How to share a secret. Commun. ACM, 22(11), 612–613. https://doi.org/10.1145/359168.359176

[19] Sorniotti, A., & Molva, R. (2010). Secret interest groups (SIGs) in social networks with an implementation on Facebook. In Sac 2010 (pp. 621–628). https://doi.org/http://doi.acm.org/10.1145/1774088.1774219

[20] Statista Inc., “Mobile messaging users worldwide 2014-2019,” statista.com, 2016. [Online]. Available: https://www.statista.com/statistics/483255/number-of-mobile-messaging-users-worldwide/. [Accessed: 20-October-2016].

[21] Yeung, C. A., Liccardi, I., Lu, K., Seneviratne, O., & Berners-Lee, T. (2009). Decentralization : The Future of Online Social Networking. In W3C Workshop on the Future of Social Networking Position Papers (Vol. 2, pp. 1–5).

[22] Zhao, J., Zhang, J., & Zhao, R. (2007). A practical verifiable multi-secret sharing scheme. Computer Standards and Interfaces, 29(1), 138–141. https://doi.org/10.1016/j.csi.2006.02.004