# Undetectable Communication: The Online Social Networks Case

Filipe Beato
KU Leuven
ESAT/COSIC and iMinds
filipe.beato@esat.kuleuven.be

Emiliano De Cristofaro
University College London
Dept. of Computer Science
e.decristofaro@ucl.ac.uk

Kasper B. Rasmussen
University of Oxford
Dept. of Computer Science
kasper.rasmussen@cs.ox.ac.uk

***Abstract*—Online Social Networks (OSNs) provide users with an easy way to share content, communicate, and update others about their activities. They also play an increasingly fundamental role in coordinating and amplifying grassroots movements, as demonstrated by recent uprisings in, e.g., Egypt, Tunisia, and Turkey. At the same time, OSNs have become primary targets of tracking, profiling, as well as censorship and surveillance. In this paper, we explore the notion of *undetectable communication* in OSNs and introduce formal definitions, alongside system and adversarial models, that complement better understood notions of anonymity and confidentiality. We present a novel scheme for secure covert information sharing that, to the best of our knowledge, is the first to achieve undetectable communication in OSNs. We demonstrate, via an open-source prototype, that additional costs are tolerably low.**

## I. Introduction

In the past few years, Online Social Networks (OSNs), such as Facebook, Twitter, Google+, Linkedin, have taken the world by storm, boasting users in the hundreds of millions. OSNs offer fast and reliable diffusion of information as well as seamless coordination of social activities. At the same time, however, they create treasure troves of sensitive information, as an entity in control of a user's data can infer her interests, whereabouts, social circles, or even political and sexual orientation.

OSN providers allow users to select which users (or groups) can access their profile, however, this process relies not only on the diligence of the users but also on the trustworthiness of the providers in enforcing access control and protecting stored content. While the risk of negative publicity and lawsuits incentives providers to safeguard user information, end-user license agreements often include clauses allowing them to mine profiles or sell information to third-party services [1]. Moreover, the concentration of personal information also exacerbates the dangers of data leaks and insider attacks.

OSNs have also become primary targets of government surveillance and Internet censorship, and not only in countries regarded as being ruled by oppressive governments. For instance, the number of subpoenas issued by U.S. law enforcement agencies on OSN data has increased steadily over the past few years [2] as also reported by the documents leaked by Edward Snowden. Several providers, e.g., Twitter, have the ability to censor content on a country basis in order to comply with governments' requests to remove or block certain content. In general, many countries worldwide are reported to block, selectively filter, or perform censorship on OSNs [3], [4].

These worrisome issues motivate the need for effective techniques to protect user privacy in OSNs. While decentralized architectures have often been advocated as a privacy-respecting alternative to social networking, e.g., [5]–[8], they often hinder reliability and real-time availability or require users to buy cloud storage for their data. On the contrary, centralized OSNs support high-availability content dissemination to a large number of non-tech-savvy users. Arguably, centralized online social networks are here to stay and actively being used by millions of people around the world, thus we focus on technologies that can be deployed atop existing OSNs.

Internet users can protect themselves from surveillance using anonymous communications (e.g., through Tor [9]), so that actions that they perform online cannot be connected with their offline identities. However, modern OSNs require users to create and maintain a profile, thus, only pseudonymity—rather than anonymity—is actually feasible with respect to the OSN provider. Moreover, an adversary having access to profiles (e.g., the provider or a government agency through a subpoena) would be able to obtain user's social interactions, content, and connections, and thus, most likely, the user's true identity.

In order to hide sensitive social content from the potentially prying eyes of the OSN provider and/or surveying entities, the security community has proposed a few solutions: some rely on encryption [10]–[14], and some on obfuscation [15]. However, besides often violating the terms of service, posting encrypted data actually draws even more attention on a user targeted by, e.g., an oppressive government.

Using the above discussion as main motivation, this paper formalizes the concept of undetectable communication in OSNs, whereby unauthorized entities are unable to detect the existence of secret messages posted and exchanged by OSN users. While steganographic techniques have been proposed to achieve general-purpose statistical undetectability, little work has focused on the specific constraints introduced by OSNs. Several approaches aim at undetectability by assuming a setting where a cover object, e.g., an image, has enough entropy to embed a secret. However, not all OSNs fit into this setting, and many providers process published images by applying compression, resizing, or removing metadata, thus, image-based steganographic techniques are moot in the OSN setting. After defining two different system models, based on the amount of entropy available in the cover object (high vs low), we introduce concrete attacker models and present an information sharing scheme in OSNs with provable undetectability.

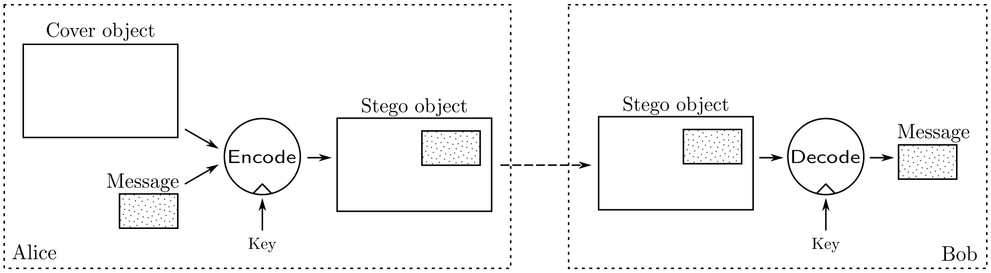

**Fig. 1:** The high-entropy model. It involves three objects: a *cover object*, a *message*, and a *stego-object*. While the message is embedded in the stego-object, the adversary should not be able to determine this without access to the key.

**Contributions.** This paper makes several contributions. First, it formalizes the notion of undetectable communication in OSNs, taking into account the limitations of modern OSNs. We then propose a protocol that provably achieves undetectability in OSNs. In the process, we identify a number of open challenges that call for further research. Finally, we build and evaluate an open-source prototype.

**Paper Organization.** The rest of this paper is organized as follows: next section introduces system and adversarial models. Section III formalizes steganographic security in OSNs, while Section IV presents our low-entropy information sharing scheme. In Section V, we discuss possible side-channel attacks on information hiding schemes in general. Then, we describe our prototype implementation in Section VI. In Section VII, we review related work, and Section VIII concludes the paper.

## II. System Model

This section introduces the system and adversarial models used throughout the rest of the paper.

**Undetectability in OSN.** Let Alice be an OSN user willing to send a secret message $m$ to another OSN user, Bob. We assume that Alice uses the OSN infrastructure and, optionally, some auxiliary out-of-band channel. Alice and Bob wish to protect the confidentiality of $m$ and also hide its existence from the adversaries defined below. We also assume that Alice and Bob share a symmetric private key, $k$.

**Adversaries.** We consider as adversaries any entity attempting to break the undetectability and/or the confidentiality of the secret message $m$ sent by Alice to Bob. In practice, there may be a few different adversarial entities, including the social network provider as well as a passive adversary monitoring Alice's and Bob's connection to the Internet.

As many social network providers rely on user data for targeted advertisement, data mining, marketing and sentimental analysis, financial and commercial interests often lead them to restrict the use of encryption mechanisms. This restriction motivates the need for undetectability in case users wish to share encrypted content. Restrictions on the use of encryption is also crucial in the presence of a surveilling government attempting to systematically monitor its citizens; besides partially (or totally) monitoring users' traffic, governments can often obtain social networking data from OSN providers, e.g., through a subpoena or even warrantless wiretapping (and, in extreme cases, coerce citizens to surrender encryption keys).

We distinguish between two adversaries, based on the amount of information they have access to.

- ***Online Social Network:*** This adversary may eavesdrop on all communication and access data routed to, or stored at, the OSN provider. This includes all data that has been posted to the OSN in the past, along with relationships, explicit or inferred, with other users of the same OSN.
- ***Internet Provider:*** This adversary has all the powers of the OSN adversary and also the ability to monitor the network traffic of any user. It cannot break secure encryption schemes, e.g., it cannot see the content of a transmission protected by HTTPS, but it can block the content and will learn the identities of communicating parties.

The second adversary is strictly more powerful than the OSN adversary. This implies that, if a scheme is not secure in the presence of the OSN adversary, then it cannot be secure against the Internet Provider adversary. Conversely, if a scheme is secure in the presence of the Internet Provider adversary, then it will also be secure against the OSN.

## III. Steganographic Models in OSNs

This section defines the concept of steganography in Online Social Networks (OSNs). We consider two possible models for undetectable communication. The first is a *high-entropy model* that captures the traditional notion of steganography, where a message is embedded inside a "normal looking" object (cover object), e.g., an image or a music file. The second is a *low-entropy model* which models the case where the cover object does not have enough entropy to contain the message.

### A. High-Entropy Model

Our first model, which we denote as the *high-entropy* model, mirrors the traditional steganography setting where a message is embedded into a cover object using an embedding function Encode specific to the cover object. This process may, or may not require a key. The resulting stego-object is self-contained, i.e., nothing besides the stego-object (and possibly a key) is required to extract the message. Definition 1 formalizes the notion of high-entropy stego-systems, and Figure 1 illustrates the model.

*Definition 1 (High-entropy stego-system):* A high-entropy stego-system $\mathbb{S}_h$ consists of the following efficient algorithms. $\mathsf{Setup}(\cdot)$ is a probabilistic algorithm that takes as input, a security parameter $1^\kappa$, and returns a key $k \in \mathcal{K}$. $\mathsf{Encode}(\cdot,\cdot,\cdot)$ is a probabilistic algorithm that takes as input a key $k$, a cover

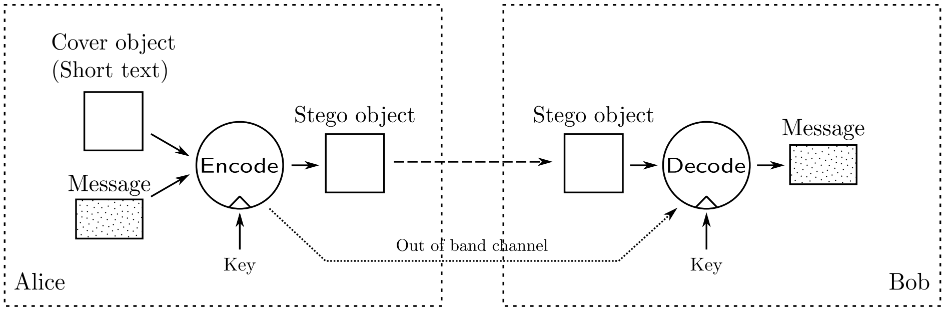

**Fig. 2:** The low-entropy model. The stego-object, e.g., in the form of a *short text*, is public and is used as input to Decode (possibly along with a key) in order to recover the secret message.

object $c \in \mathcal{C}$, and a (secret) message $m \in \{0,1\}^l$, and returns a stego-object $o \in \mathcal{O}$. $\mathsf{Decode}(\cdot,\cdot)$ is a deterministic algorithm that takes as input a key $k$ and a stego-object $o$, and returns the embedded message $m$. There must exist a polynomial $p(|c|)$ such that:

$$\forall m, |m| < p(|c|): \quad \mathsf{Decode}(k, \mathsf{Encode}(k,c,m)) = m$$

Security in the high-entropy model pertains to the unfeasibility of an attacker to distinguish between a cover object and a stego-object. Security definitions are presented in Section III-C.

Naturally, the cover object must have enough entropy to contain the message. For example, if the cover object is a 2MB image and the message is a short 100-byte text, the image could be modified in such a way that the 100 bytes of text could be embedded, without noticeably altering the image [16], [17]. On the other hand, if the cover object does not have enough entropy to hide the message (e.g., a large image cannot be embedded in a short text), then another approach has to be used. We present such an approach below as the low-entropy model.

### B. Low-Entropy Model

The *low-entropy* model is used if the cover object does not have enough entropy to contain the message. Without loss of generality, we consider the cover object as a short text, e.g., some text that could seamlessly be published on a social network such as, Twitter or Facebook. The low-entropy model is illustrated in Figure 2.

Observe that the cover object (e.g., some short text) is chosen to *represent* the secret message, rather than have the message encoded in it. The process of linking the stego-object to the message may, or may not, require a key. The secret message itself must be sent to the recipient(s) using an out-of-band channel, since, by definition, the stego-object cannot contain it. Definition 2 formalizes the notion of low-entropy stego-system.

*Definition 2 (Low-entropy stego-system):* A low-entropy stego-system $\mathbb{S}_l$ consists of the following efficient algorithms. $\mathsf{Setup}(\cdot)$ is a probabilistic algorithm that, takes as input a security parameter $1^\kappa$, and returns a key $k \in \mathcal{K}$. $\mathsf{Encode}(\cdot,\cdot,\cdot)$ is a probabilistic algorithm that takes as input a key $k$, a cover object $c \in \mathcal{C}^*$, and a secret message $m \in \{0,1\}^l$, and returns a stego-object $o \in \mathcal{O}^*$ and a secret message $m'$ (which may be identical to $m$). $\mathsf{Decode}(\cdot,\cdot,\cdot)$ is a deterministic algorithm that takes as input a key $k$, a stego-object $o$, and a secret $m'$, and returns the message $m$. It must hold that:

$$\forall m: \quad (o, m') = \mathsf{Encode}(k,c,m); \mathsf{Decode}(k,o,m') = m$$

When Bob receives the stego-object, he will use it to determine the nature of the out-of-band channel and eventually get the secret message. For instance, consider the following example scenario:

> Alice and Bob have agreed on two different physical sites for a dead-drop (an envelope with confidential information). They have also agreed on three keywords that Alice will use when the information is ready to be picked up, “Hello” for the first location, “Good morning” for the second location and “Good day” for abort pickup.

Prearranged keywords and locations represent the *key* in the low-entropy model, while the dead-drop – the out-of-band channel. We anticipate that our novel covert information sharing, aiming to achieve steganography in OSNs and presented in Section IV, will follow the low-entropy model.

### C. Security Definition

We now formalize the notion of steganographic security in OSNs. We aim to provide a generic definition that applies to both high- and low-entropy stego-systems, thus, we start by defining the general notation for a stego-system.

*Definition 3 (Stego-system):* A stego-system $\mathbb{S}$ is either a high-entropy stego-system $\mathbb{S}_h$ or a low-entropy stego-system $\mathbb{S}_l$.

To simplify the notation used in the rest of this section, we let $o(m)$ denote a cover object that simply encodes the message $m$. (Actually, in a high-entropy stego-system $o(m) = \mathsf{Encode}(k,c,m)$, whereas, in a low-entropy stego-system $(o(m), m') = \mathsf{Encode}(k,c,m)$.)

Before presenting our notion of security of a stego-system $\mathcal{S}$, we introduce, in Game 1, the following game played between a challenger and an adversary.

*Game 1 ($\mathsf{IND\text{-}STEGO}_{\mathsf{Adv},\mathsf{Ch},\mathbb{S}}(\kappa)$):* The game between an adversary Adv and a challenger Ch proceeds as follows:

1) Adv is given access to an oracle that returns cover objects $\{c_1, \ldots, c_q\}$, which are taken from the set appropriate

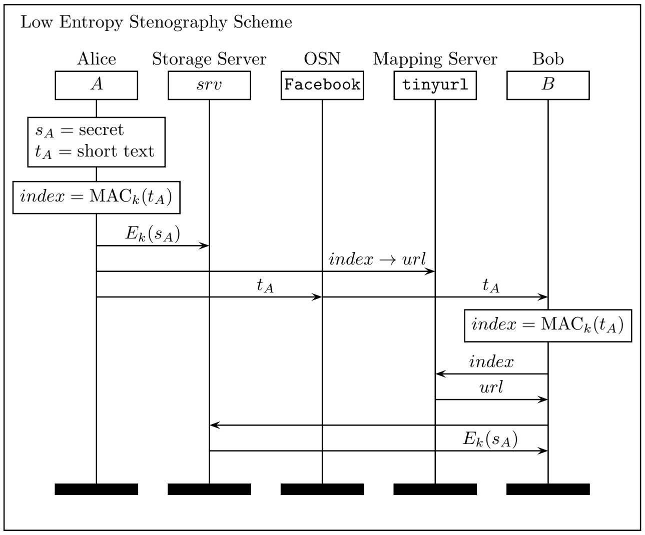

**Fig. 3:** Low entropy information sharing scheme. Alice derives an index from the MAC of a small cover text. She then uses a mapping service, e.g., tinyurl or a TOR hidden service, to create a mapping from that index to the URL that contains the secret data. Alice posts the cover text on Facebook where Bob reads it. Later, Bob can retrieve the secret by first deriving the index from the cover text and then obtaining the URL for the secret message from the mapping server. All connections to the storage- and mapping server are assumed to be encrypted, e.g., using HTTPS.

for the type of stego-system, i.e., $\mathcal{C}$ for a high-entropy stego-system and $\mathcal{C}^*$ for a low-entropy stego-system.

2) Adv outputs a message $m$ and Ch returns either $c' = o(m)$ or a random stego-object $c'$ with probability $\frac{1}{2}$.
3) Eventually, Adv outputs a bit $b$, where $b = 1$ if Adv believes that $c' = o(m)$ and $b = 0$ otherwise. The game outputs 1 iff $(c' = o(m) \wedge b = 1) \vee (c' \neq o(m) \wedge b = 0)$, i.e., if Adv could successfully guess the type of object returned by Ch.

We now define IND-STEGO security using Game 1 above:

*Definition 4 (*IND-STEGO *security):* A stego-system $\mathbb{S}$ is IND-STEGO-secure if there exists a negligible function negl such that, for any probabilistic polynomial time adversary Adv, it holds that:

$$\Pr\big(\mathsf{IND\text{-}STEGO}_{\mathsf{Adv},\mathsf{Ch},\mathbb{S}}(\kappa) = 1\big) \leq \frac{1}{2} + \mathsf{negl}(\kappa)$$

## IV. Covert Information Sharing Scheme

In this section, we describe our low-entropy stenography scheme. The scheme conforms to the low-entropy model defined in Section III-C.

Our goal is to allow Alice to communicate some secret information to Bob, via a social network, e.g., Facebook. Alice is assumed to be unable to post the information directly to Facebook for fear of being arrested by the local authorities. To simplify the description we first describe the protocol with a single receiver (Bob), then we generalize to multiple receivers and groups. The scheme is illustrated in Figure 3.

Alice and Bob share a key $k$, that is used to derive the encryption key $k_{ENC}$ and the MAC-key $k_{MAC}$. Given some secret information that Alice would like to communicate $s_A$, Alice will first pick a short text $t_A$, independent of the secret, that will be published on Facebook. The short text can be any arbitrary string that does not invoke suspicion. Alice creates the encryption key $k_{ENC} = H(k \parallel 0)$ and the MAC-key $k_{MAC} = H(k \parallel 1)$, using a collision resistant hash function $H(\cdot)$. She then uploads the secret $s_A$, optionally encrypted $E_{k_{ENC}}(s_A)$, to the storage service, e.g., Dropbox. At the same time, Alice uploads to the mapping service (e.g., TinyURL or a specific Tor Hidden Service) the $url$ to the storage service along with a mapping index, $index = MAC_{k_{MAC}}(t_A)$ (this corresponds to Encode in our model). The mapping service links the $index$ to the $url$, and allows flexibility for the choice of the storage server. Note that while it is optional to encrypt the actual secret, it is a requirement that the communication with the storage server is over a secure connection, e.g., HTTPS. In addition, if the storage server provider supports the option to setup accounts, Alice can also create a temporary username and password and set an account as $(urs, pwd) = MAC_{k_{MAC}}(t_A \parallel index)$. These steps can all be done days before Alice actually intends to transmit the secret to Bob, if needed.

When Alice wishes to send the secret information to Bob, she publishes her chosen message $t_A$ on Facebook. This will look to Facebook (and anyone else) as an innocent message that does not carry any additional information. Bob will MAC the text using the key shared with Alice to derive the MAC-key and obtain the $index$ that points to the $url$ of the storage service by using the mapping service, such that $index = \mathrm{MAC}_k(t_A)$ (this corresponds to Decode in our model). Bob can then connect to $url$ and retrieve the (possible encrypted) secret $s_A$. Again the communication with the storage server and the mapping service must be over a secure connection.

If Alice wants to communicate the secret to multiple receivers, she simply creates multiple accounts, one for each receiver, on the storage server. Each receiver can then independently retrieve a copy of the secret. For revocation, Alice simply deletes the account corresponding to the key she wishes to revoke, and users of that key can no longer access the secret. Users in possession of the revoked key cannot even tell that the message $t_A$ posted to Facebook corresponds to a secret, since the storage server will not recognize the temporary $(usr, pwd)$ generated using the revoked MAC-key.

### A. Use of the OSN Infrastructure

Our covert information sharing scheme assumes that users share a key, hence, at some point they must have established a secure, and possibly authenticated, channel. Therefore, one could question why these users would later communicate using the social network infrastructure rather than this secure channel. In some cases, direct communication might be the best option but there are several reasons, beyond convenience, why one might want to use a low-entropy steganographic approach instead. First, and foremost, the direct secure channel might not be available all the time. Alice and Bob could have exchanged USB sticks with each others' cryptographic keys at some point in the past, but the information they want to communicate is only available now. Another reason could be that the secure channel is very low bandwidth and cannot be used to transfer the entire secret message.

Given that Alice and Bob share a key, they could also choose to communicate directly, e.g., via an encrypted email attachment, rather than relying on the OSN. Again there are plenty of scenarios where this would be the preferred way but if Alice and Bob are trying to conceal the fact that they are transferring a potentially large and secret message, our protocol is ideal.

In addition, by using a mapping service our scheme allows flexibility with respect to the choice of the storage server per shared secret. Thereby, if a motivated adversary blocks this service then the user can always switch to a more privacy-friendly server.

### *B. Security Analysis*

As mentioned earlier, our proposed scheme is an example of a low-entropy stego-system. We now prove that it is $\mathsf{IND\text{-}STEGO}$-secure, by measuring the advantage an adversary $\mathsf{Adv}$ has in wining the $\mathsf{IND\text{-}STEGO}$-game. We consider two different adversaries (defined in Section II): the Online social network adversary and the Internet Provider adversary.

**Online Social Network.** Without loss of generality we will use Facebook as an example of a social network adversary. Facebook has the ability to read any message posted by any user as well as monitor user behavior.

Even though Facebook has full access to the short text $t_A$, there is no way to check if $t_A$ corresponds to any secret information, since $t_A$ is specifically chosen independently of the secret. The short text need not have any specific structure or be about any specific topic. In fact, $t_A$ could be a text that Alice would have posted anyway and therefore it is indistinguishable from any other message in $\mathcal{C}^*$, in fact Facebook does not even know which server a potential secret is stored on, since this information is part of the key. This means that Facebook can only win the $\mathsf{IND\text{-}STEGO}$-game with probability $1/2$, and the scheme is thus $\mathsf{IND\text{-}STEGO}$-secure under a social network adversary.

**Internet Provider.** The Internet Provider adversary is assumed to have all the powers of the social media plus the ability to monitor all connections of specific users in their jurisdiction.

First, we observe that if none of the parties involved in the protocol (other than Facebook) are under observation the analysis is identical to the social media scenario, and the scheme is $\mathsf{IND\text{-}STEGO}$-secure.

If *at least one other party* is under observation, we need a more careful analysis. Without loss of generality we assume that Alice is the one under observation. The adversary ($\mathsf{Adv}$) will see that Alice connected to the storage server, i.e, $\mathsf{Adv}$ will know $srv$, which is part of the key. $\mathsf{Adv}$ can try to use this knowledge to get an advantage in the $\mathsf{IND\text{-}STEGO}$-game.

The $\mathsf{IND\text{-}STEGO}$-game proceeds according to Game 1 as follows. (1) $\mathsf{Adv}$ has access to all the users previous messages (as well as any arbitrary message). (2) $\mathsf{Adv}$ submits a secret message $m_{\mathsf{Adv}}$ to $\mathsf{Ch}$ and $\mathsf{Ch}$ must now return a stego-object. This involves computing the MAC of an independently chosen short text $t_{\mathsf{Ch}}$, to obtain a username and password, then upload a secret (either $m_{\mathsf{Adv}}$ or random data, chosen with probability $1/2$) to the storage server $srv$, and make the secret accessible using the newly created username and password. After that the stego-object $t_{\mathsf{Ch}}$ is returned to $\mathsf{Adv}$. (3) $\mathsf{Adv}$ must now guess if the secret on the storage server is $m_{\mathsf{Adv}}$ or not.

Having $A$ under observation, $\mathsf{Adv}$ knows the location of the storage server $srv$, but assuming the connection between $A$ and $srv$ is secure, $\mathsf{Adv}$ learns nothing about the data exchanged between $A$ and $srv$. $\mathsf{Adv}$ learns nothing by supplying an incorrect username and password to the storage server, and $\mathsf{Adv}$ cannot create the username and password without knowledge of the MAC-key. Guessing the username and password corresponds to guessing the output of the MAC, which can only be done with probability $1/2^n$, where $n$ is the number of bits of the MAC output. Since $1/2^n$ is negligible the protocol is $\mathsf{IND\text{-}STEGO}$-secure.

## V. Side Channel Attacks

Our system model in Section III, captures the notion of undetectable communication. The security definition in Game 1 is based on the difficulty of distinguishing a stego-object from a covert object. This distinction is, however, subject to various side channel attacks in any practical scheme. In this section, we discuss a number of such side channel attacks that could affect the security of a practical protocol. First, we introduce the concept of social indistinguishability and highlight its importance for undetectability in OSNs and, later, discuss traffic analysis.

### *A. Social Indistinguishability*

The nature of *undetectable* communication requires more than just confidentiality. It requires that no one is able to identify the cover messages as suspicious. We call this notion *Social Indistinguishability*. It is very difficult to quantify exactly what social indistinguishability means. For example, even if the cover message is completely unrelated to the secret topic, it can still be suspicious if it is unusual for an individual to express themselves in a certain way. consider the following example: if Alice is usually interested in football but has never expressed any interest in politics, an adversary might have a good reason to suspect that a message containing political comments is a cover object for a secret message. However, close to elections, it might be perfectly normal for Alice to comment on political figures, even though she is not normally very politically active. (And similar behavior may exist for any major event, TV show, news story, Internet meme, etc.)

We choose to model the notion of social indistinguishability in terms of the constraints a communication system places on the cover message. For a naive stego-system where the secret message is derived from the first letter of every word, the cover message must use words that start with that specific letter in order to convey the secret message. This will make it hard to choose a cover message that appears innocent, i.e., socially indistinguishable.

Our low entropy stenography scheme can use any cover text without any constraints. This means that the user is free to express himself in the exact way that he chooses, and that is appropriate to the context of the message. This freedom comes from the fact that the cover message itself does not actually contain the secret message, rather, it acts as an index to where the secret can be found. Since the cover message is known

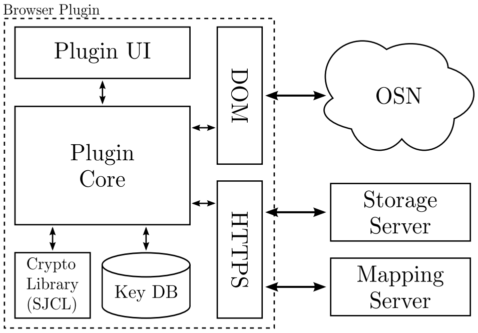

**Fig. 4:** Plugin block diagram. The user interacts with the plugin in three ways. The key $k$, the secret message $m$ and the cover text $t_A$ is communicated to the user via a dedicated *Plugin UI*. The plugin communicates with the OSN, e.g., Facebook, using the Document Object Model (DOM) in the browser, and communication with the storage- and mapping server is done using HTTPS.

to Alice when she stores the secret message, she can change the storage location (path in a URL) to fit the cover message, rather than the other way around.

### B. *Traffic Analysis*

Our security analysis has, thus far, set aside the issue of traffic analysis [18], although it can ostensibly help the adversary in the scenario where the storage server is under observation. Consider the following example: as the storage server is under observation, the adversary notices that Alice uploads 1,564 bytes of data. Later, Bob connects to the same server and downloads exactly 1,564 bytes of data. Even without considering their interaction on Facebook, it seems likely that the adversary can guess that there was a transfer of information between Alice and Bob.

Traffic analysis constitutes a traditional obstacle to privacy, e.g., for confidentiality [19], [20] and anonymity [21], as well as censorship resistance [22]. To cope with it, a few solutions have been proposed both in the general Internet setting [23] and in OSNs [24]. We readily acknowledge that the security of our proposed scheme holds assuming traffic analysis resistance and leave, as part of future work, a thorough study of traffic analysis issues and countermeasures in the context of our covert information sharing scheme.

## VI. Prototype Implementation

To demonstrate the viability of our proposals, we implemented a proof-of-concept prototype of the covert information sharing scheme proposed in Section IV.[1]

In the description of the implementation, we distinguish between server- and client-side components, as depicted in Figure 4. The former is used to realize the out-of-band storage service, whereas, the latter runs as a browser extension on the user environment.

**Server-side.** In our prototype, the server-side corresponds to a simple PHP back-end server and a MySQL database. It supports post and get actions:

- *Post:* The storage server returns an $url$ location when receives, from the user, the tuple $(s, usr, pwd)$, i.e., the secret $s$ (optionally encrypted), along with the username and password generated according to our scheme in Section IV, and stores such tuple in its database.
- *Get:* On input $(usr, pwd)$, the storage server returns $s$ to the user.

**Client-side.** Our covert information sharing scheme is designed to work with existing OSNs, such as, Facebook, Twitter, Google+. Therefore, users will interface with the system via the regular OSN web site. Each operation in our scheme, such as Encode and Decode, is implemented as a web transaction, and users perform them from their web browser. There is no need to perform any operation outside the browser: our covert information sharing scheme only involves simple symmetric-key operations that can be executed, e.g., in Javascript using the Stanford Javascript Crypto Library (SJCL) [25]. We use AES-CMAC [26] for the MAC implementation and AES-CCM [27] for the authenticated symmetric encryption, as both are already available in SJCL.

However, we need a mechanism to seamlessly implement the interaction between the user and the storage server, i.e., without requiring the user to run other software other than their browser or to leave the OSN website. To this end, we built a Firefox Extension (FE) that, installed on the user's device, is used to post and read secret messages, alongside with the TinyURL website for the mapping service. Specifically, the Encode and Decode operations are as follows:

- Encode *(Post):* The user, Alice, selects a text area on the OSN website. The FE launches a dialog where the user inserts the secret message $s_A$ and the short text $t_A$. In addition, the FE publishes $t_A$ in the selected text area, and produces $(usr, pwd) = \mathrm{MAC}_k(t_A)$. Subsequently, FE uploads the tuple $(c_A, usr, pwd)$ automatically into the server that returns a $url$, where $c_A = E_{k_{ENC}}(s_A)$. At the same time, the FE uploads the tuple $(index, url)$ to the TinyURL server, such that $index = \mathrm{MAC}_{H(k\|1)}(t_A)$.
- Decode *(Get):* FE parses the messages on the OSN, and, for each message from Alice, produces $index = \mathrm{MAC}_{k_{MAC}}(t_A)$. The $index$ is used to query the TinyURL service for the valid $url$. Then, the FE submits the tuple $(usr, pwd)$ to $url$, that outputs $c_A$ if there is a match, and $\perp$ otherwise. If $c_A$ exists and the decryption result is $s_A$, then the FE replaces, transparently, $t_A$ with the secret message $s_A$.

The current prototype is compatible with Firefox 14+, but it could be easily ported to other browsers extensions, e.g., to Chrome, as it is written in simple Javascript. In terms of performance, as it is resumed to the MAC and AES implementations, that takes about $2ms$, and to the communication latency existent between the client- and server-side, the prototype presents limited complexity. Thus, while it only supports desktop browsers at the moment, it is perfectly suitable for resource-constrained devices, such as smartphones. This is crucial considering that a significant portion of users access OSNs via their mobile devices (e.g., almost 60% of Facebook users in October 2012 [28]).

[1] Source of our implementations is freely available upon request.

## VII. Related Work

This section reviews related work: we start with privacy-enhancing technologies for OSN communication, then, overview techniques for undetectability focusing on OSNs.

### A. *Privacy-Enhancing Technologies in OSNs*

A few efforts, e.g., [29]–[31], analyzed a multitude of privacy issues in Facebook, Twitter, MySpace, and other OSNs. Then, several cryptographic protocols have been proposed to improve confidentiality, access control, or anonymity in centralized OSNs. The work in [32] applies the concept of "virtual private networks" to OSNs, i.e., it establishes a confidential channel between friends in order to share sensitive data. Then, FaceCloak [11], and FSEO [33], lets Facebook users encrypt sensitive data and store it on third party servers, while posting a short *fake* text. Only authorized users holding decryption keys can produce the mapping index, to extract and decrypt sensitive data on the server. However, FaceCloak uses random sentences from Wikipedia to represent the fake text, thus, according to our definitions in V-A, it does not achieve social indistinguishability. In fact, FaceCloak [11] employs fake data not to achieve undetectability, rather, to circumvent restrictions on the use of encryption. FSEO [33], on the other hand, allows publishers to choose their *fake* text. Further, Scramble! [12] uses broadcast encryption for improved access control on Facebook, i.e., allowing each user to specify the recipients of shared information, similar to the concept of *circles* in Google+, and to hide content from the provider, while Hummingbird [13] presents a variant of Twitter that provably guarantees confidentiality of tweet contents, hashtags, and follower interests.

Decentralized OSNs have also been advocated for privacy-aware social networking. For example, [6] provides a cryptographic API that achieves improved access control, anonymity and confidentiality. Whereas, our approach focuses on privacy for centralized OSNs, a social-network model that supports high availability and real-time content dissemination. In fact, decentralized architectures might hinder *real-time* availability of information or require users to buy cloud storage for their data [7]. Furthermore, we are interested in privacy-enhancing technologies that can be plugged in on top of social networks that are used today by hundreds of millions of people. In the context of OSNs, the decision to switch to an other service is collective by nature, thus, creating challenging obstacles to large-scale diffusion of new infrastructures, even if privacy-enhanced, for users for whom privacy is not a primary task. In other words, users may not be motivated to switch, unless the majority of their friends will switch as well.

### B. *Undetectability in OSNs*

Undetectability constitutes the main objective of steganography, i.e., the practice of transferring hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message. Steganography has been intensely studied in the last several years: we now review relevant models and applications to OSN settings.

Several efforts, e.g., [34]–[36], have focused on information theoretical definitions for steganography. Cachin [34] proposes security definitions and adversarial games for steganography in the presence of a passive adversary. Work in [37] and [38] define theoretical frameworks, based on computational assumptions, following a model akin to indistinguishability in cryptography.

There is also a large number of steganographic techniques using images as cover objects. Many methods rely on the Least-Significant Bit (LSB) to avoid large visual changes in the cover object. Techniques like those proposed in [39]–[41] use the LSB to store different types of secrets. The F5 algorithm [42] follows the traditional steganography model but is resilient to visual and statistical attacks, previously highlighted in [43]. Also, some work on text steganography has been done based on cryptographic paradigms, e.g., [44], [45],

To the best of our knowledge, there is relatively little work analyzing the notion of steganography in the specific context of OSNs. In general, OSN providers process published images by applying compression, resizing, or removing metadata, making it very hard for well-known image-based steganographic approaches to work. Additionally, adversaries could use steganalysis techniques, e.g., [46], to detect images containing secret information. Further, NOYB [10] proposes a technique based on substitution ciphers, used to encrypt user data (specifically, personal details) and encode resulting ciphertexts to look like fake yet legitimate data. This is accomplished using uniformly distributed data from an external dictionary. Therefore, one could say that NOYB aims to (somewhat) limited undetectability in OSNs, although limited to personal details, whereas, our goal is to let users unnoticeably exchange any secret message on any OSN.

Castiglione et al. [47] propose using image filenames and tagging as cover objects to embed a secret in an Online Photo Service (OPS) and circumvent image manipulation issues. However, this approach requires a large number of images and an a-priori shared knowledge of pictures.

Castelluccia et al. [48] use caches of publicly available open DNS resolvers as an out-of-band channel to encode bits of an encryption key, employed to encipher messages exchanged, e.g., on OSNs. While they do not aim at undetectability, their intuition could be modified by mapping ciphertexts in to innocent-looking texts (e.g., using "MadLibs" techniques [49]), but it remains unclear how to unnoticeably communicate the list of resolvers where bits are stored. Also, the availability of information in DNS caches would be limited to entries' time-to-live (which, incidentally, is a design goal of [48]).

## VIII. Conclusion

Motivated by the limited effectiveness of privacy-enhancing technologies aiming at confidentiality and anonymity in Online Social Networks (OSNs), this paper presented a first-of-its-kind study of undetectability in OSNs. After formalizing system and adversarial models, we presented a novel scheme for secure covert information sharing in OSNs and demonstrated, via an open-source prototype, that incurred additional costs are tolerably low.

Although inherently limited by the centralized nature of modern OSN architectures, as well as by the power of global government-like adversaries, the attained degree of privacy constitutes an important step forward toward secure OSN communications. We also identified some important open challenges that call for further research. Items for future work include further analyzing social indistinguishability and traffic analysis issues, usability evaluations of large-scale deployments of covert information sharing schemes, as well as exploring the concept of plausible deniability in OSNs.

**Acknowledgments.** Filipe Beato is funded by the FCT PhD fellowship SFRH/BD/70311/2010. This work was also supported by the Research Council KU Leuven: GOA TENSE (GOA/11/007) and by the IWT SBO SPION project.

## References

[1] J. Geetter, E. Sussman, and C. Hine, "FTC: Collecting and Selling Data Mined from Social Media Sites Covered by FCRA," http://preview.tinyurl.com/a6rap4r, 2012.

[2] C. Matyszczyk, "If your account is subpoenaed, Facebook sends police, well, everything," http://preview.tinyurl.com/facebook-subpoena, 2012.

[3] Freedomhouse.org, "Freedom on the net 2012 – A global assessment of Internet and digital media," http://www.freedomhouse.org/sites/default/files/FOTN%202012%20FINAL.pdf, 2012.

[4] A. Chaabane, T. Chen, M. Cunche, E. De Cristofaro, A. Friedman, and M.-A. Kafaar, "Censorship in the Wild: Analyzing Internet Filtering in Syria," *arXiv preprint 1402.3401*, 2014.

[5] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," *Sigcomm Computer Communication Review*, vol. 39, no. 4, 2009.

[6] M. Backes, M. Maffei, and K. Pecina, "A security API for distributed social networks," in *NDSS*, 2011.

[7] A. Shakimov, H. Lim, R. Caceres, L. Cox, K. Li, D. Liu, and A. Varshavsky, "Vis-á-vis: Privacy-preserving online social networking via virtual individual servers," in *COMSNETS*, 2011.

[8] S. Nilizadeh, S. Jahid, P. Mittal, N. Borisov, and A. Kapadia, "Cachet: A Decentralized Architecture for Privacy Preserving Social Networking with Caching," in *CoNEXT*, 2012.

[9] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router," in *Usenix Security*, 2004.

[10] S. Guha, K. Tang, and P. Francis, "NOYB: Privacy in Online Social Networks," in *WONS*, 2008.

[11] W. Luo, Q. Xie, and U. Hengartner, "FaceCloak: An Architecture for User Privacy on Social Networking Sites," in *CSE*, 2009.

[12] F. Beato, M. Kohlweiss, and K. Wouters, "Scramble! Your Social Network Data," in *PETS*, 2011.

[13] E. De Cristofaro, C. Soriente, G. Tsudik, and A. Williams, "Hummingbird: Privacy at the time of twitter," in *S&P*, 2012.

[14] A. Feldman, A. Blankstein, M. Freedman, and E. Felten, "Social Networking with Frientegrity: Privacy and Integrity with an Untrusted Provider," in *Usenix Security*, 2012.

[15] A. Pashalidis, N. Mavrogiannopoulos, X. Ferrer, and B. Bermejo Olaizola, "For human eyes only: security and usability evaluation," in *WPES*, 2012.

[16] G. Cancelli and M. Barni, "MPSteg-color: A New Steganographic Technique for Color Images," in *IH*, 2007.

[17] R. Chandramouli and N. D. Memon, "Analysis of LSB based image steganography techniques," in *ICIP*, 2001.

[18] G. Danezis and R. Clayton, "Introducing traffic analysis," 2007.

[19] Q. Sun, D. Simon, Y. Wang, W. Russell, V. Padmanabhan, and L. Qiu, "Statistical identification of encrypted web browsing traffic," in *S&P*, 2002.

[20] G. Danezis, "Traffic Analysis of the HTTP Protocol over TLS," http://research.microsoft.com/en-us/um/people/gdane/papers/TLSanon.pdf, 2009.

[21] S. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *S&P*, 2005.

[22] L. Invernizzi, C. Kruegel, and G. Vigna, "Message in a Bottle: Sailing Past Censorship," in *HotPETs*, 2012.

[23] C. Wright, S. Coull, and F. Monrose, "Traffic morphing: An efficient defense against statistical traffic analysis," in *NDSS*, 2009.

[24] C. Diaz, C. Troncoso, and A. Serjantov, "On the impact of social network profiling on anonymity," in *PETS*, 2008.

[25] E. Stark, M. Hamburg, and D. Boneh, "Symmetric Cryptography in Javascript," in *ACSAC*, 2009.

[26] J. Song, R. Poovendran, and J. Lee, "The AES-CMAC-96 Algorithm and Its Use with IPsec," RFC 4494 (Proposed Standard), 2006.

[27] D. Whiting, R. Housley, and N. Ferguson, "Counter with CBC-MAC (CCM)," RFC 3610 (Proposed Standard), 2003.

[28] E. Protalinski, "600 million of Facebook's 1 billion users are mobile," http://preview.tinyurl.com/bp5mrcv, 2012.

[29] L. Fang and K. LeFevre, "Privacy wizards for social networking sites," in *WWW*, 2010.

[30] H. Mao, X. Shuai, and A. Kapadia, "Loose Tweets: An Analysis of Privacy Leaks on Twitter," in *WPES*, 2011.

[31] Y. Wang, S. Komanduri, P. Leon, G. Norcie, A. Acquisti, and L. Cranor, "I regretted the minute I pressed share: A qualitative study of regrets on Facebook," in *SOUPS*, 2011.

[32] M. Conti, A. Hasani, and B. Crispo, "Virtual private social networks," in *CODASPY*, 2011.

[33] F. Beato, I. Ion, S. Capkun, M. Langheinrich, and B. Preneel, "For Some Eyes Only: Protecting Online Information Sharing," in *CODASPY*, 2013.

[34] C. Cachin, "An Information-Theoretic Model for Steganography," in *IH*, 1998.

[35] T. Mittelholzer, "An Information-Theoretic Approach to Steganography and Watermarking," in *IH*, 1999.

[36] J. Zöllner, H. Federrath, H. Klimant, A. Pfitzmann, R. Piotraschke, A. Westfeld, G. Wicke, and G. Wolf, "Modeling the Security of Steganographic Systems," in *IH*, 1998.

[37] S. Katzenbeisser and F. A. Petitcolas, "Defining security in steganographic systems," 2002.

[38] N. J. Hopper, L. von Ahn, and J. Langford, "Provably secure steganography," *IEEE Transaction on Computers*, vol. 58, no. 5, 2009.

[39] E. Franz and A. Pfitzmann, "Steganography Secure against Cover-Stego-Attacks," in *IH*, 1999.

[40] J. Roque and J. Minguet, "SLSB: Improving the Steganographic Algorithm LSB," in *CIBSI*, 2009.

[41] N. N. EL-Emam, "Hiding a large amount of data with high security using steganography algorithm," in *Journal of Computer Science*, 2007.

[42] A. Westfeld, "F5—a steganographic algorithm: High capacity despite better steganalysis," in *IH*, 2001.

[43] A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems - Breaking the Steganographic Utilities EzStego," in *Jsteg, Steganos, and S-Tools-and Some Lessons Learned*, 2000.

[44] Y. Wang and P. Moulin, "Perfectly Secure Steganography: Capacity, Error Exponents, and Code Constructions," *IEEE Transactions on Information Theory*, vol. 54, no. 6, 2008.

[45] A. Kiayias, Y. Raekow, and A. Russell, "Efficient Steganography with Provable Security Guarantees," in *IH*, 2005.

[46] N. Provos and P. Honeyman, "Detecting Steganographic Content on the Internet," in *NDSS*, 2002.

[47] A. Castiglione, B. D'Alessio, and A. D. Santis, "Steganography and Secure Communication on Online Social Networks and Online Photo Sharing," in *BWCCA*, 2011.

[48] C. Castelluccia, E. De Cristofaro, A. Francillon, and M. Kaafar, "EphPub: Toward Robust Ephemeral Publishing," in *ICNP*, 2011.

[49] M. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, "Loud and clear: Human-verifiable authentication based on audio," in *ICDCS*, 2006.