# FaceCloak: An Architecture for User Privacy on Social Networking Sites

Wanying Luo, Qi Xie, Urs Hengartner
Cheriton School of Computer Science,
University of Waterloo, Waterloo, ON, Canada
{w8luo, q7xie, uhengart}@cs.uwaterloo.ca

*Abstract*—**Social networking sites, such as MySpace, Facebook and Flickr, are gaining more and more popularity among Internet users. As users are enjoying this new style of networking, privacy concerns are also attracting increasing public attention due to reports about privacy breaches on social networking sites. We propose FaceCloak, an architecture that protects user privacy on a social networking site by shielding a user's personal information from the site and from other users that were not explicitly authorized by the user. At the same time, FaceCloak seamlessly maintains usability of the site's services. FaceCloak achieves these goals by providing fake information to the social networking site and by storing sensitive information in encrypted form on a separate server. We implemented our solution as a Firefox browser extension for the Facebook platform. Our experiments show that our solution successfully conceals a user's personal information, while allowing the user and her friends to explore Facebook pages and services as usual.**

## I. Introduction

The advent and fast adoption of Web 2.0 technologies has dramatically changed the Internet and has enabled people to build social networks online regardless of their geographical locations. Popular social networking sites, such as Facebook, allow users to explore other users with similar interests, share personal information with friends, showcase photos, etc. However, the ease of socializing online also raises privacy concerns, sometimes resulting in severe consequences. For example, 13 crew members were dismissed by Virgin Atlantic due to their inappropriate posts to Facebook [1]. A teacher in Wisconsin was suspended after she posted a picture of herself with a gun to Facebook [2].

Social networking sites generally allow a user to post sensitive personal information, such as relationship status, sexual orientation, political affiliation, and various personal interests. Although the public is regularly warned about the risks associated with social networking sites, a large portion of the population is still unaware of the potential privacy threats or even chooses to expose personal information despite these risks. According to a survey conducted at Carnegie Mellon University, the university's users of Facebook provide an astonishing amount of information: 90.8% of the profiles contain an image, 87.8% of the users reveal their birth date, 39.9% list a phone number (including 28.8% of profiles that contain a cellphone number), and 50.8% list their current residence [3]. Besides the wishful thinking that the Internet is a family of well-behaved users, another factor that contributes to people's disregard of privacy risks is the trust in the protection measures and good intentions of a social networking site. Unfortunately, these sites are no more secure than any other website in terms of defending themselves against malicious attackers. The two biggest players in online social networking, Facebook and MySpace, were both found to be prone to cross-site scripting attacks enabling attackers to steal user credentials [4], [5]. Moreover, social networking sites are vulnerable to insider attacks, such as Facebook employees seeing or even modifying any user's personal information [6]. Finally, a social networking site might make a user's profile available to third parties for advertising purposes. In its privacy policy [7], Facebook states that information provided in this way will not identify the user; however, guaranteeing this property in practice is hard (see, e.g., Sweeney [8]).

After having studied various existing solutions, we believe that the type of privacy protection technologies that can effectively circumvent the threats raised by user unawareness and server-side vulnerabilities is a client-side architecture that automates the process of privacy protection. We make the following contributions:

- We present FaceCloak, an architecture that enforces user privacy on social networking sites by shielding a user's personal information from the site and from other users that were not explicitly authorized by the user. At the same time, the services and the user interface provided by the site continue to function as before.
- We introduce a novel scheme that allows users to customize what information should be shielded from the social networking site. More specifically, users are given the option to express what information they intend to put a guard on, and any information can be left unprotected if they truly desire so. For example, existing users can leave their names and some profile information unencrypted, so that old friends can still get in touch with them.
- We evaluate the design behind FaceCloak by applying it to the Facebook platform, which recently became the largest social networking site [9]. Whereas our design is applicable to other social networking sites as well, we decided to focus on Facebook for simplicity.

The rest of the paper is organized as follows: In Section II, we survey related work that addresses privacy protection on social networking sites. We explain our design principles and security assumptions in Section III. The architecture of FaceCloak is described in Section IV, which is followed by the security analysis in Section V. Section VI presents the details of our prototype implementation of FaceCloak.

## II. Related Work

Several new systems and architectures for privacy protection on social networking sites have been proposed.

flyByNight [10] is a Facebook application designed to protect the privacy of messages exchanged between Facebook users. It adopts public key encryption algorithms to encrypt a user's message before sending it via Facebook to the server hosting the application. A user's private key is encrypted with a password and also stored on the flyByNight server. All cryptographic operations are performed in a user's browser with JavaScript code that is downloaded from the flyByNight server via Facebook. In this scheme, both Facebook and the flyByNight server need to be trusted not to provide the user's browser with malicious JavaScript code that leaks messages or private keys. Even if this trust assumption held, the use of encryption remains problematic because it could cause suspicion on the side of Facebook and may even cause user accounts to be suspended. Moreover, flyByNight is a Facebook application, so its fate is entirely at the discretion of Facebook. In the worst case, Facebook could remove the application since it prevents Facebook from learning users' information and from using this information for advertising and other purposes. FaceCloak is not a Facebook application, and it is designed not to be at the mercy of Facebook. Moreover, FaceCloak leaves no traces of encryption on a user's Facebook pages, so it is less likely to attract the attention of Facebook.

NOYB (short for "None Of Your Business") [11] is another system targeted at protecting user privacy on Facebook using "encryption" in a novel way. Instead of applying traditional encryption schemes, which leave clear traits of ciphertext, NOYB divides a user's private information into atoms and replaces each atom with the corresponding atom from a randomly selected other user. For example, user A's profile ($\mathrm{name}_\mathrm{A}$, $\mathrm{gender}_\mathrm{A}$, $\mathrm{age}_\mathrm{A}$, $\mathrm{addr}_\mathrm{A}$) is broken into the three pieces ($\mathrm{name}_\mathrm{A}$, $\mathrm{gender}_\mathrm{A}$), ($\mathrm{age}_\mathrm{A}$), and ($\mathrm{addr}_\mathrm{A}$), which are then substituted with ($\mathrm{name}_\mathrm{B}$, $\mathrm{gender}_\mathrm{B}$), ($\mathrm{age}_\mathrm{C}$), and ($\mathrm{addr}_\mathrm{D}$) from users B, C, and D, respectively. Only user A herself and A's Facebook friends have enough information to reverse this process to recover A's profile. Although NOYB employs encryption in this novel way to avoid the problems caused by traditional encryption schemes, it has two limitations: (1) NOYB protects only the privacy of user profiles. Since any piece of information posted by a user to Facebook applications can also be exploited to invade her privacy, a more general way is required. FaceCloak can protect the privacy of both a user's profile and the data posted to a Facebook application. (2) The number of users that use NOYB impacts its effectiveness. The larger the number of users, the better the anonymity. The effectiveness of FaceCloak is not affected by the number of its users. (3) NOYB does not allow old friends to get in touch unless they have enough information to recover the profile information of their friends. FaceCloak supports incremental deployment and allows old friends to get in touch.

Social networking APIs let third parties access sensitive user information stored on a social networking site. This API makes it possible to greatly enhance the services offered by a site (e.g., Facebook applications), but it also poses privacy risks. Felt et al. [12] studied the 150 most popular Facebook applications and found that almost all of them were unnecessarily given wider access to private user data than needed. Felt et al. designed a privacy-by-proxy approach to improve social networking APIs such that third-party applications are prevented from accessing real user data while the functionality and availability of the applications are preserved. Compared to our approach, the privacy-by-proxy design deals only with the privacy risks posed by Facebook applications, but assumes that Facebook itself is trustworthy. In cases where this assumption does not hold, the privacy-by-proxy design is rendered useless. In contrast, FaceCloak does not assume that Facebook is trustworthy and thus greatly enhances user privacy.

## III. Assumptions and Goals

Privacy protection on social networking sites is a difficult research problem. To our best knowledge, there are no widely accepted protection schemes. Our goal is to make our solution immediately usable and have it cover more privacy risks than previous research. In this section, we examine the threat model and the design principles that underlie our solution.

### A. Threat Model

We consider two types of threat: The social networking site itself and sensitive information seekers.

- **Social networking site**. In Section I, we have already outlined several ways in which a social networking site, typically not deliberately, might reveal a user's personal information to parties not authorized by the user. Moreover, an attacker could break into the social networking site and gain access to any user's personal information, or the provider of the site might be forced by the government or a court to disclose personal information. In our threat model, we assume that an attacker can launch any sort of attack against the social networking site and gain access to any personal information that a user has stored on the site. Therefore, the social networking site must be considered a potential threat for user privacy and should not have access to a user's personal information.
- **Sensitive information seeker**. A sensitive information seeker tries to invade the privacy of the users of a social networking site by exploring the site's pages to gather sensitive user information. In particular, for users who fail to limit access to only their friends, information seekers can easily browse their profiles, their blogs (e.g., Facebook Notes), or their bulletin boards (e.g., Facebook Wall). As observed by other researchers [13], the default privacy settings of social networking sites are often quite lax. For example, Facebook by default grants anyone in a user's networks or communities access to the user's

profile. Many users are unaware of these default privacy settings, so they often end up not restricting access to only their friends. Even if a user prevents sensitive information seekers from accessing her profile but lets them access her blog or her bulletin board, a sensitive information seeker might still be able to derive profile information from these applications. For example, the message "Happy 16th Birthday!!" posted by a friend to a user's bulletin board and the posting date reveal the user's birth date.

We assume that users' computers are not compromised. In particular, we rely on the integrity of users' web browsers, since our solution is implemented as a browser extension.

### B. Design Principles

The design of FaceCloak is based on four key principles:

1) **Preservation of normal browsing experience**. An important property for a usable privacy protection solution is to refrain from interfering with users' normal browsing activities. More specifically, the solution should function automatically most of the time and require little user interaction. Constantly interrupting users for input or actions will distract them. Our solution automatically applies data encryption/decryption, page manipulation, etc and requires no user intervention.
2) **No server-side changes**. Providers of social networking sites value monetary profits as their primary goal, just as any other business, and user privacy protection is more often than not put on the back burner. There generally is no incentive for these providers to introduce changes to their system architecture for the purpose of privacy protection, unless those changes have monetary gains or are legally required. Therefore, a widely deployable privacy protection mechanism should not rely on server-side cooperation or changes. Our solution requires no such modifications and cooperation.
3) **Self-containment and minimal user configuration**. Users of social networking sites have technical skill levels ranging from almost zero to highly experienced. To make a privacy protection solution usable to all the users regardless of their skills, the solution should be self-contained, not rely on users to install additional software, and require minimal configuration. We implemented FaceCloak as a Firefox browser extension, which can be installed in the same way as any other Firefox browser extension, and it requires no configuration.
4) **Incremental deployment**. FaceCloak users should not be stopped from getting in touch with old friends. To achieve incremental deployment, FaceCloak must ensure compatibility between the ones using it and the ones that are not relying on it.

## IV. FaceCloak

FaceCloak carries out privacy protection in three phases: the **setup phase**, the **encryption phase** and the **decryption phase**. Figure 1 gives an overview of the three phases. When a user of a social networking site sets up FaceCloak in her browser,

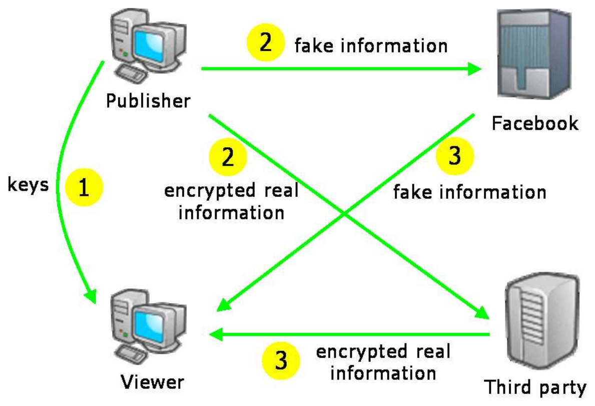


Fig. 1: Architecture of FaceCloak

FaceCloak generates several keys and distributes a subset of these keys to the user's friends (see Section IV-A). An optional task for the setup phase is setting up a third party server, which is used to store and retrieve the user's encrypted personal information. In the encryption phase, FaceCloak guides a content publisher (i.e., a user who posts information to her or her friend's account on the social networking site) to encrypt the posted information and send it to the third party server over a connection protected by TLS (see Section IV-B). This step guarantees that any information that the content publisher chooses to hide from the site and from users not authorized by the account owner will not get sent to the social networking site. Instead, fake information will be transmitted to the site. The decryption phase occurs when a content viewer (i.e., the owner of an account or an authorized friend of the owner) wants to look at information posted to the account (see Section IV-C). The content viewer decrypts the real information retrieved from the third party server over a connection protected by TLS and uses it to replace the fake information obtained from the social networking site. This phase ensures that services offered by the site continue to function properly despite the intervention of FaceCloak.

Both existing and new users of a social networking site can benefit from FaceCloak. Existing users, who have already sent their profile information to the site, can use FaceCloak to protect future messages that they or their friends post, such as articles on their blog or messages on their bulletin board. For new users, FaceCloak protects their information starting with the account registration, so these users have the additional benefit of ensuring the privacy of the information stored in their profile, such as their real name, birth date, and gender. It is up to a new user to leave some of this information unprotected to make it easier for old friends to find her profile.

### A. Setup Phase

When a user installs FaceCloak, it generates three keys: a master key, a personal index key, and an access key. A copy of the master key and the personal index key are distributed by the user to her friends, whereas the access key is stored locally on the user's computer and never distributed. To prevent the

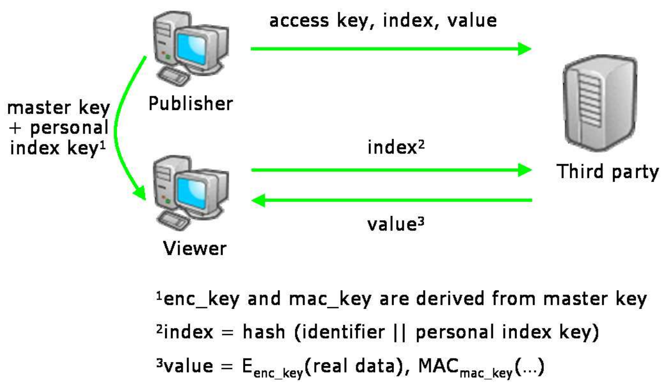

access key, index, value
Publisher
master key + personal index key[1]
index[2]
Third party
value[3]
Viewer
[1]enc_key and mac_key are derived from master key
[2]index = hash (identifier || personal index key)
[3]value = $E_{enc\_key}$(real data), $MAC_{mac\_key}$(...)

Fig. 2: Interaction with the third party server

social networking site from learning the master and personal index keys, the user should use out-of-band mechanisms, such as e-mail, to distribute them. However, users do not need to manually send e-mails to their friends, as FaceCloak provides two useful tools to automate this task (see Section VI-A).

An account's master key is used by a content publisher to derive a symmetric encryption key for encrypting information that is posted to the account and a MAC key for protecting the integrity of this information. A content viewer who is in the possession of the master key can decrypt the encrypted information and check its integrity.

An account's personal index key is used for storing encrypted personal information that is posted to the account. A third party server stores index-value pairs, where a value consists of the encrypted personal information and its MAC (see Figure 2). Both the content publisher posting the information and a content viewer authorized to look at the information can compute the index for this information. The index is the cryptographic hash of 1) the personal index key of the account to which the information was posted and 2) an identifier that depends on the type of posted information. Namely, for information in a user's profile, the identifier corresponds to the user's fake name, as generated by FaceCloak (see Section VI-A). In other words, a user's real profile is stored in encrypted form on the third party server using a single index-value pair. For messages posted to a blog or bulletin board, the entire fake message that is generated by FaceCloak and posted to the social networking site serves as the identifier in the index computation. That is, each real message is stored in encrypted form on the third party server using a separate index-value pair. Our algorithm for generating fake messages makes it highly unlikely that the same fake message is generated multiple times for the same account (see Section VI-A), which would result in an index collision.

Note that a content publisher uses the master key and the personal index key that she generated for her own account only when she adds information to her profile or her applications (e.g., her blog or her bulletin board). If she adds messages to somebody else's applications, she will use the master key and personal index key associated with this person's account. If she is not in the possession of these keys, FaceCloak will refuse to post a message.

Finally, the access key is required when a content publisher stores new data or tries to update existing data on the third party server. It is not required when a content viewer tries to retrieve data. A content publisher always uses the access key generated for her own account, regardless to which account she posts a message. (As mentioned before, access keys are not distributed in the first place.) The purpose of the access key is to prevent denial-of-service attacks. Suppose an account's personal index key leaks to an attacker. Without the protection of the access key, the attacker might be able to compute valid indices for information posted to the account and can replace the encrypted information stored on the third party server with arbitrary data. Consequently, neither the account owner nor her friends will be able to view the real data. As a countermeasure to this attack, the third party server requires that overwrites of existing data are accompanied by the access key that was used to store the existing data. Our design implies that the owner of a bulletin board cannot modify the real content of postings made by other people to her bulletin board. However, she can still remove the corresponding fake content, which will result in the real content becoming inaccessible, because its index can no longer be computed.

The owner of an account with a social networking site can choose the third party server that should be used for storing encrypted personal information that is posted to her account. The owner trusts the third party server not to collude with the social networking site (even if they colluded, they could not learn the owner's personal information), to store encrypted data reliably, and to provide correct data on request. A publicly accessible third party server is maintained by the authors of this paper. However, any group of FaceCloak users may set up their own server, and we provide server-side PHP scripts and MySQL scripts to ease the task.

## *B. Encryption Phase*

Assuming a content publisher has installed FaceCloak in her browser, she enters texts in HTML forms on a social networking site just as she normally does, except that she prepends the entered text with a special marker pre-defined by FaceCloak ("@@" in the current implementation). For other form elements, such as dropdown menus or radio buttons, the FaceCloak automatically adds another set of the same inputs prefixed with the special marker. Since a design principle is incremental deployability, users can leave any field unencrypted (by not prefixing it with the special marker) for purposes such as disclosing real names to get in touch with old friends.

When the content publisher submits the form to the social networking site, FaceCloak intercepts the submitted information. Fields that start with the special marker get replaced by appropriate fake text. The fake text cannot simply be random combinations of letters that do not compose valid words, or random valid words that convey nonsense in the specific context, because junk data can be noticed by the social networking site. To avoid this problem, FaceCloak judges the intended content of the field first, and then generates fake text using various techniques, including consulting its own dictionaries or Wikipedia, which we discuss in more details in Section VI-A. After the real text has been replaced with fake text, the form is submitted to the social networking site.

Next, the real data needs to be encrypted and sent to the third party server. FaceCloak determines the account's master key and personal index key, and derives the encryption and MAC keys and the index, as discussed in Section IV-A. It then encrypts the real information and computes its MAC. Finally, the index and the value, consisting of the encrypted data and its MAC, are sent, together with the content publisher's access key, to the third party server.

### C. Decryption Phase

FaceCloak, as installed in a content viewer's browser, knows which of the viewer's friends also use FaceCloak, because it has the friends' master and personal index keys. However, after downloading a friend's page, FaceCloak does not know which of the downloaded information is real or fake. Obviously, we cannot tag fake information and store tags with the social networking site, since this would allow the site to identify FaceCloak users. Having FaceCloak store tags locally on users' computers does not scale. Instead, for each piece of information that could be fake (i.e., a profile or an article), FaceCloak computes an index, as explained in Section IV-A, and tries to download the corresponding value from the third party server. If there is a value, FaceCloak checks the integrity of the received ciphertext, decrypts it, and updates the downloaded page by substituting the real data for the fake data. Otherwise, the data is left unchanged. We use the same procedure when downloading the content viewer's own pages.

When a content viewer without the master key and the personal index key of an account owner browses the account's pages, she will receive fake information and will not be able to perform the above procedure to check for fake information. Therefore, the page will show fake information to anyone who has not received the necessary keys from the account owner.

## V. Security Analysis

This section analyzes the security of FaceCloak by considering several attacks against our architecture.

### A. Social Networking Site

Our solution protects against all the threats arising from the social networking site itself, as described in Section III-A. A user's sensitive information that the user chooses to protect using FaceCloak is stored on a third party server beyond the reach of the social networking site. The information stored on the site's server is fake, which bears no significant meaning whatsoever. However, we do rely on the user to judge what information is sensitive and should be protected according to her own standard, and FaceCloak never forces the user to protect her information. Another threat that our solution deals with is the social networking site trying to identify users of FaceCloak and then taking punitive actions against them, such as suspending their accounts. Namely, the fake information generated by FaceCloak looks real (see Section VI-A). Moreover, by including a user's personal index key, which is not known to the social networking site, in the index of information stored with the third party server, we make it impossible for the site to query publicly known third party servers about the existence of encrypted information posted to the user's account.

### B. Third Party Server

The third party server, which is responsible for storing encrypted data, can also become the target of attackers, or the server itself could turn malicious. In these cases, the attacker can view all the ciphertexts, but not their plaintexts. The attacker cannot figure out the account owner of the encrypted information, either. The attacker can modify ciphertexts, but with the help of the MAC, FaceCloak will detect these modifications. In a replay attack, the attacker can replace the current value of an index-value pair with a previous (valid) value. FaceCloak currently does not detect replay attacks. However, FaceCloak users might get suspicious when, for example, their profile all of a sudden reverts to stale information. The users can then notify the provider of the third party server of the breakin and the original content can be restored from a backup. If the provider of the third party server turned malicious, the user should switch to a new server. In the worst case, the user could lose all information stored with the old server. However, the same threat also exists in a world without FaceCloak, where users have to rely on the social networking site not to turn malicious. Finally, we note that none of the attacks that involve the third party server put users' privacy at risk.

### C. Collusion

Even a social networking site that colludes with a third party server cannot decrypt a user's personal information stored on the third party server. However, a social networking site that colludes with a third party server might be able to detect which users are using FaceCloak by launching timing attacks. More specifically, since FaceCloak sends the encrypted data to the third party server immediately after it submits the fake data to the social networking site, if both the site and the third party server log the time when they receive data from users, the site can compare its own log to the log of the third party server. When the site detects a FaceCloak user, the site can suspend the user's account, which is a nuisance. One way to mitigate the impact of collusion is to distribute a user's encrypted data across multiple servers, which we leave for future work. Also, different users can use different third party servers, which makes collusion harder.

### D. Web Browser

The most disastrous threat to FaceCloak is the compromise of a user's web browser, because FaceCloak is implemented as a browser extension. Once the attacker takes control of the browser, she can disable or uninstall the extension. The attacker may also extract the victim's keys from the extension, which will give him access to the victim's personal information stored on the third party server. The best security measures to mitigate this threat are to educate users to constantly patch browser vulnerabilities and install anti-virus software.

### *E. Leak of Keys*

The security of FaceCloak largely depends on the trustworthiness of friends. A user has no control over the master and personal index keys once she sends them to a friend. In particular, the friend can send the keys to another person who is not a friend of the victim. This person will be able to access the victim's personal information stored on the third party server. The person will not necessarily be able to access the victim's data through browsing the social networking site because the victim might have restricted access to her pages to only her friends. In general, there is no technical solution that can reliably prevent a friend from maliciously distributing information that the friend has had access to. One way to limit the damage resulting from key leakage is key revocation. Our implementation does not yet support such a mechanism.

## VI. Implementation and Experiments

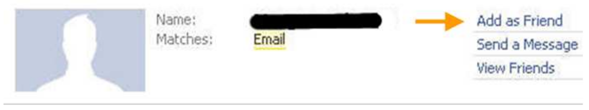
In this section, we elaborate on the implementation of FaceCloak and present experiments that we conducted by interacting with a real social networking site and its services. We also discuss limitations of the current implementation.

### *A. Firefox Browser Extension*

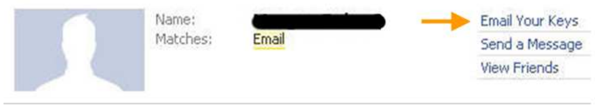
Our current implementation of FaceCloak works only for Facebook. We chose Facebook because it has become the largest networking site [9]. However, we do not expect much difficulty in porting FaceCloak to other sites. We are currently preparing a public release of our implementation.

Since a large portion of the tasks performed by FaceCloak involves HTML manipulation, we implemented FaceCloak as a Firefox browser extension. We use a JavaScript implementation of AES [14] to carry out data encryption/decryption and a JavaScript implementation of SHA-1 [15] to compute indices of encrypted data. All keys have a length of 128 bits. To facilitate users to distribute keys to their friends by e-mail, FaceCloak provides two useful tools. The first one is an e-mail list manager, through which the user can add her friends' e-mail addresses to her contact list, as managed by FaceCloak. The user then clicks the "Email" button, which will bring up her default e-mail client with the subject of the e-mail, the recipient's e-mail address, and the pre-composed content (i.e., the personal index and master keys and some explanatory information) all filled in. The second tool is dynamically generated on the Facebook page for adding a friend. After having sent a friend request, the extension replaces the hyperlink text "Add as Friend" with hyperlink text "Email Your Keys", which the user can click to bring up her e-mail client with all parts already filled in. This feature is shown in Figure 3.

The browser extension includes a female English first name dictionary, a male English first name dictionary, and an English surname dictionary, which are used to generate fake names



(a) The page before the friend request is sent



(b) The page after the friend request is sent

Fig. 3: Facility for e-mailing keys

when users register Facebook accounts. Since Facebook account registration also requires the user to enter her birth date, a fake birth date is generated such that the resulting age is between 15 and 65. If the user decides to hide her gender, the extension chooses the fake gender randomly. The fake first name is chosen to conform to the gender. Besides the registration information, the extension can also protect all other profile information, such as relationship status, and political and religious views. FaceCloak does not generate fake information for this additional profile information, since Facebook does not require this profile information. When somebody looks at the user's profile, FaceCloak retrieves the real information from the third party server and adds additional HTML code to the profile to display it. We chose not to replace optional profile information with fake information because finding fake information that looks real can be hard. Also, for some profile information (e.g., an address or a phone number), it is ethically questionable to replace it with fake information that turns out to be the real information for somebody else.

Furthermore, to prevent information leakage caused by Facebook applications, the extension currently also supports privacy protection for the Facebook Wall and Facebook Notes applications. Generating fake text for these two applications is challenging, since the content is not limited and could be about anything. For instance, a user can publish a poem, a science article, or a diary in her Facebook Notes. The extension makes use of the "Random article" feature of Wikipedia to generate fake text in this case. More specifically, when the user finishes composing a note and is ready to publish it, the extension retrieves a random article from Wikipedia and parses the content to strip off HTML tags. The resulting text is used as the fake note that is submitted to Facebook, where the user is given a chance to review the fake note before its submission.

### *B. Third Party Server*

We set up a third party server, with a MySQL database for storing encrypted data at the University of Waterloo. FaceCloak makes use of asynchronous JavaScript to send HTTP requests to the server for the purpose of storing, updating, and retrieving data. Based on the value of a parameter in the HTTP request, the server performs either a "get" or "put" action. The "put" action refers to storing or updating a piece of encrypted data and its MAC on the server, while the "get" action involves retrieving the encrypted data using the corresponding index. The server responds to a request with an XML string, which contains an error code, the error message in case the execution failed, and the requested data in case of a "get" action.

### C. Experiments

To verify that the extension successfully conceals user information and provides acceptable performance, we conducted the following experiment. Suppose John Doe uses our extension to register a Facebook account. Before his registration information is submitted to Facebook, our extension replaces it with fake information as shown in Figure 4. In order not to violate the Terms of Use (TOU) of Facebook (see below), we modified the extension to change the fake information to the real information of one of the authors of this paper, namely, Wanying Luo. We omit the real birth date for privacy reasons.

After his account is created, John Doe logs in as Wanying Luo. Figure 5a shows the page after he logs in. Although fake information was sent to Facebook, the extension successfully replaced it with the real information retrieved from the third party server. A feature of the extension is that any information (except the fake name due to technical issues) that FaceCloak replaces or inserts is displayed in red, which makes it easier to spot problems. For example, the birth date is shown in red (see Figure 5a, while the gender is shown in the default color since John Doe chooses to reveal his real gender. To verify that the real information is invisible to unauthorized Facebook users, another author of this paper logged in to Facebook with his own account and added Wanying Luo as his friend. Suppose John Doe does not trust this user and refuses to send his keys to this user. As displayed in Figure 5b, this unauthorized user can only view John Doe's fake information. During the experiment, we also observed that the fake information on most parts of a page is already replaced with the real information by the time the page is loaded, thus, users will not be able to tell that page manipulation is taking place. However, on some parts of the page, the replacement occurred about one second after the page is loaded, which makes users see the fake information first. This phenomenon is due to the fact that asynchronous JavaScript is heavily used by Facebook for rendering the page. By the time the basic structure of a Facebook page is loaded, it does not necessarily have the entire page fully rendered, and some parts may still be fetching contents. Therefore, the extension has to wait until the fake information appears on the page, and then starts to replace it.

This experiment actually uses our extension in the opposite way than a user is supposed to use it, because we used the real information of one of the authors to replace the information of an imaginary person, John Doe. The purpose is simply to avoid breaking Facebook's TOU [16], which make the user agree to submit accurate profile information. Nevertheless, if a user is concerned about the privacy of her name and birth date, she does have to choose between protecting this information and breaking the TOU. For example, the Information and Privacy Commissioner of Ontario registered a Facebook account under a fake name [17]. If a user is not willing to break the TOU, she can still use our extension to protect profile information other than her name and birth date and postings to Facebook Wall and Notes. Here, the user would register with Facebook under her real name and birth date. In this scenario, our extension gives the user the flexibility to acknowledge add-friends requests from not very close friends, in order not to annoy them. However, she would give her keys and hence access to her complete profile only to close friends.

### D. Limitations

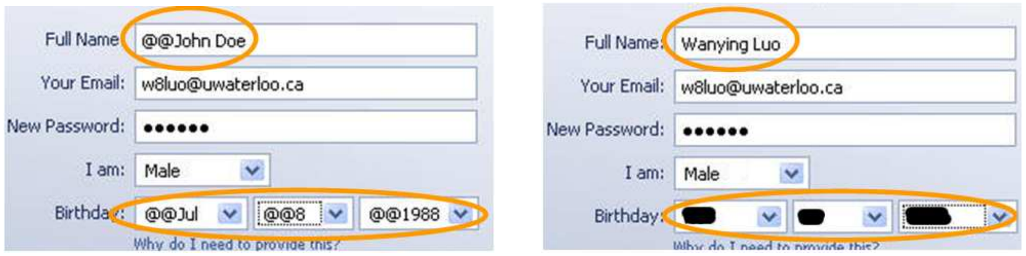
For simplicity, the current implementation of our extension assumes that a Firefox profile is used by only one user, so it associates this user's Facebook account with her Firefox profile, and all the files related to this user's Facebook account (such as her keys, the list of her friends' e-mails, etc) are stored under the directory of her Firefox profile. Therefore, the extension will not function properly if a Firefox profile is shared among multiple Facebook users.

The retrieval of the real information requires keys. Therefore, a user could not use FaceCloak on computers without her keys. The keys are stored in the extension, so a user may move these keys between computers using a USB stick.

Our implementation of FaceCloak currently supports only Facebook. However, the architecture is also applicable to other social networking sites, and we do not foresee any technical difficulties in porting our current implementation other sites. In fact, implementing FaceCloak for other sites might actually be easier than our Facebook implementation. Though not confirmed by Facebook, we read from other sources that Facebook engineers may have used very obscure ways to render their pages, which makes replacing part of the content difficult. Our experiences confirm this observation, and the extension required some tweaking.

## VII. Conclusions and Future Work

Online social networks have acquired enormous popularity and become an essential part of many people's daily life. Along with the convenient ways of socializing with others online comes the threat to privacy. While the research community has devoted intensive efforts to looking for effective means of protecting user privacy, incidents resulting from privacy breaches on social networking sites continue to happen. We have developed a solution to provide privacy protection for users of social networking sites. The solution takes a strong privacy stand that not only should the sensitive information of a user be protected from unauthorized users, it should also be shielded from the social networking site. Furthermore, the solution allows a user to be selective of which information she wants to safeguard and which to leave as it was, based on her own judgment of the value of privacy. Compared to previous approaches, this feature dramatically improves the usability and deployability of our solution. In addition, we implemented our design of FaceCloak as a Firefox browser extension and showed its practicality. Although the implementation is currently useful only for Facebook users, the architecture is general enough to be applied to other social networking sites.



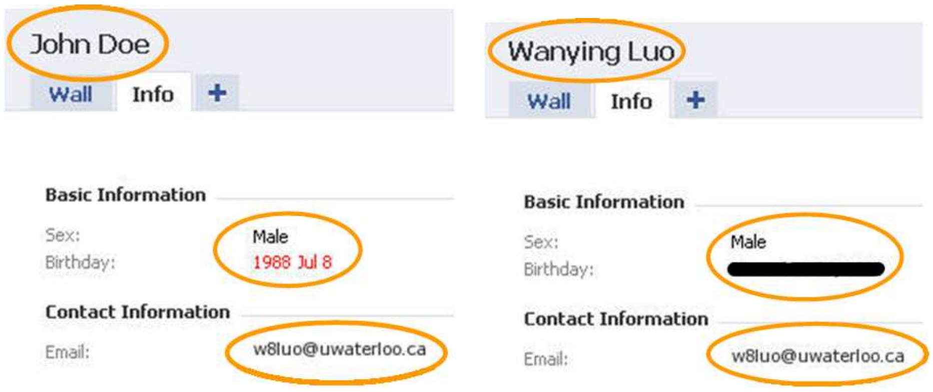
Full Name: @@John Doe
Your Email: w8luo@uwaterloo.ca
New Password: ●●●●●●
I am: Male
Birthday: @@Jul @@8 @@1988
Why do I need to provide this?

(a) Page before clicking submit

Full Name: Wanying Luo
Your Email: w8luo@uwaterloo.ca
New Password: ●●●●●●
I am: Male
Birthday: [redacted] [redacted] [redacted]
Why do I need to provide this?

(b) Page after clicking submit

Fig. 4: Facebook account registration



John Doe
Wall Info +
Basic Information
Sex: Male
Birthday: 1988 Jul 8
Contact Information
Email: w8luo@uwaterloo.ca

(a) View of account owner

Wanying Luo
Wall Info +
Basic Information
Sex: Male
Birthday: [redacted]
Contact Information
Email: w8luo@uwaterloo.ca

(b) View of unauthorized members

Fig. 5: A Facebook account under protection

Several directions of future work are possible: (1) To allow better damage control in case keys are leaked, we should incorporate a key revocation mechanism. (2) The extension currently makes use of the "Random article" feature of Wikipedia to generate fake data for Facebook Notes and Wall. This strategy may give away which users are using our extension by checking their Notes and Wall postings. (3) The extension supports only the protection of textual information. We would like to explore the protection of pictures and videos.

## REFERENCES

[1] BBC News, "Crew sacked over Facebook posts," http://news.bbc.co.uk/2/hi/uk_news/7703129.stm, October 2008, accessed April 2009.

[2] Switched, "Teacher Suspended for Gun Pictures on Facebook," http://www.switched.com/2009/02/06/teacher-suspended-for-facebook-gun-pictures/, February 2009, accessed April 2009.

[3] R. Gross, A. Acquisti, and J. H. Heinz, "Information Revelation and Privacy in Online Social Networks," in *Proc. of 4th ACM Workshop on Privacy in the Electronic Society (WPES'05)*, November 2005, pp. 71–80.

[4] darknet.org.uk, "Another 0-day MySpace XSS Exploit," http://www.darknet.org.uk/2007/02/another-0-day-myspace-xss-exploit/, February 2007, accessed April 2009.

[5] CyberInsecure.com, "New Cross-Site Scripting Vulnerability Found On Facebook," http://cyberinsecure.com/new-cross-site-scripting-vulnerability-found-on-facebook/, May 2008, accessed April 2009.

[6] VALLEYWAG, "How Facebook employees break into your profile," http://valleywag.gawker.com/tech/your-privacy-is-an-illusion/how-facebook-employees-break-into-your\-profile-319630.php, November 2007, accessed April 2009.

[7] Facebook, "Privacy Policy," http://www.facebook.com/policy.php, November 2008, accessed April 2009.

[8] L. Sweeney, "k-anonymity: a model for protecting privacy," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[9] TechCrunch, "Facebook Now Nearly Twice The Size Of MySpace Worldwide," http://www.techcrunch.com/2009/01/22/facebook-now-nearly-twice-the-size-of-myspace-worldwide, January 2009, accessed April 2009.

[10] M. M. Lucas and N. Borisov, "flyByNight: Mitigating the Privacy Risks of Social Networking," in *Proc. of 7th ACM Workshop on Privacy in the Electronic Society (WPES 2008)*, October 2008, pp. 1–8.

[11] S. Guha, K. Tang, and P. Francis, "NOYB: Privacy in Online Social Networks," in *Proc. of 1st Workshop on Online Social Networks (WOSN 2008)*, August 2008, pp. 49–54.

[12] A. Felt and D. Evans, "Privacy Protection for Social Networking Platforms," in *Proc. of Web 2.0 Security and Privacy (W2SP 2009)*, May 2008.

[13] A. Acquisti and R. Gross, "Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook," in *Proc. of 6th Workshop on Privacy Enhancing Technologies (PET 2006)*, June 2006, pp. 36–58.

[14] H. Hanewinkel, "AES (Rijndael) Encryption Test in JavaScript," http://www.hanewin.net/encrypt/aes/aes.htm, 2005, accessed April 2009.

[15] Movable Type Scripts, "SHA-1 Cryptographic Hash Algorithm," http://www.movable-type.co.uk/scripts/sha1.html, 2005, accessed April 2009.

[16] Facebook, "Terms of Use," http://www.facebook.com/terms.php, September 2008, accessed April 2009.

[17] CBC, "Search Engine with Jesse Brown," http://www.cbc.ca/searchengine/blog/2009/03/podcast_25_cctvs_biometrics_an.html, March 2009, accessed April 2009.