Who Am I? Two-Four-Six-Oh-One!

Michael Bernstein
MIT CSAIL
32 Vassar Street
Cambridge, MA 02139
msbernst@mit.edu

Adam Marcus
MIT CSAIL
32 Vassar Street
Cambridge, MA 02139
marcua@csail.mit.edu

Rob Miller
MIT CSAIL
32 Vassar Street
Cambridge, MA 02139
rcm@mit.edu

David Karger
MIT CSAIL
32 Vassar Street
Cambridge, MA 02139
karger@mit.edu

CSCW 2010 Workshop on Revisiting Research Ethics in the FB Era

Abstract
We discuss the practice of retaining user IDs, screen names or e-mails in an online research system. While collection of this information is common practice and necessary for most applications' functioning, it is technically disallowed by policies on Personally Identifiable Information. Do we build onerous protections, or revisit our privacy policies?

Introduction
We have a long literary history of wrestling with referents and names. In Victor Hugo's classic novel *Les Misérables* and its derivative Broadway musical, the main character Jean Valjean struggles to strip away the undesirable epithet of his ex-prison number, 24601. Yet he cannot remove the taint of his past deeds, eventually admitting musically [5]:

> *Who am I?*
> *Can I conceal myself forever more*
> *Pretend I'm not the man I was before*
> *And must my name until I die*
> *Be no more than an alibi*
> *Must I lie?*
> *[…]*
> *Who am I?*
> *Two-four-six-oh-one!*

It is relatively unlikely that Jean Valjean would have exuded the same passion if he were instead singing about personally identifying information on Facebook:

First name	No session key required (application can query without the user logging in again)
Last name	
Timezone	
Birthday	
Sex	
Profile URL	
Picture	
Affiliations	
List of friends' IDs	
Education history	Requires a session key (application can only access within ~24 hours of user logging in)
Hometown	
Interests	
Movies	
Music	
Quotes	
Website	
Work history	
Interested in {men, women, both, neither}	Requires application authorization (user must click through an additional authorization screen)
Looking for {friendship, dating, a relationship, networking}	
Significant other's ID	
Religion	

Table 1. A sampling of personal information available to applications on Facebook, and the required authorization level.

Who am I?
I don't even know my Facebook ID
While commercial apps get it for free
And must I click through scary forms
To satisfy IRB norms?

But the issue is serious. Every expedited human subjects proposal that gets approved by MIT comes with boilerplate: "The above referenced protocol is considered exempt […] pursuant to Federal regulations, 45 CFR Part 46.101(b)(2). This part of the federal regulations requires that the information be recorded by investigators in such a manner that subjects cannot be identified, directly or through identifiers linked to the subjects. It is necessary that the information obtained not be such that if disclosed outside the research, it could reasonably place the subjects at risk of criminal or civil liability, or be damaging to the subjects' financial standing, employability, or reputation."

This requirement stipulates that the research system must keep personally identifiable information about the users separate from any logs or experimental results. The participants' real identities must not be tied to their actions, and the full participant list must be hidden. This requirement is easy to carry out in the laboratory. When a participant signs the release form, the researcher assigns a number and ties all results to that anonymous number. The information linking number to name is kept in a locked drawer.

The situation is much less straightforward on a social network. Here, the participant's real identity is inexorably tied to their interactions with the application. Consider building a Facebook application for a research project; we have deployed several such systems[1,2,3] [1]. When a user joins, Facebook passes that user's Facebook ID to the application. That ID is the key to a veritable treasure trove of information about the participant, much of which is important or necessary to make the application do anything interesting (Table 1). Users are conditioned to authorize the application when requested, or else they cannot use it. [2]

The Ethical Dilemma

The material in Table 1 clearly constitutes personally identifiable information. It is certainly enough to stalk an individual, or to tie information to the original user. Is prisoner 24601, Jean Valjean, safe from being found out on Facebook? Apparently not – try visiting http://www.facebook.com/profile.php?id=24601. "Jean Valjean" certainly didn't expect us to be able to learn much about her just by typing in a number.

Facebook ID numbers could be considered personally identifiable information because they are the key to this data. If so, they need to be stored "in such a manner that subjects cannot be identified." But, true anonymity may be impossible: Facebook users interact using their online identities, not anonymous avatars.

The Pragmatic Dilemma

In order to build a Facebook application, we need to store the Facebook ID somewhere accessible to the software. Otherwise, we can't query friend networks, send messages, or take advantage of other parts of the API. Research systems also have likely implemented

[1] Which HCI Researcher Are You? http://apps.facebook.com/which-hci-res-bbfifj
[2] Stat.us. http://apps.facebook.com/stat-us
[3] Sociapedia. http://apps.facebook.com/sociapedia

logging infrastructure. The information is stored in a database somewhere, password-protected. However, MIT's IRB does not regard password protection as offering the same privacy guarantees as locking the name-to-ID papers in a file cabinet.

The Hardline Solution

MIT advocates what we call The Hardline Solution: placing personally identifiable information in a separate database from the rest of the application. So, the information linking "user 1" to Facebook user ID 24601 is in a second database.

This solution is both difficult to program, and the security is incomplete. To make a Facebook application work – that is, to use the Facebook API – we need the user's Facebook ID. Common operations like database joins ("tell me the name of user #1") become very slow and cumbersome to execute when you are combining two separate databases. This is especially problematic when Facebook requires that all page requests return in under 10 seconds. Worse, the security is not equivalent to locking IDs in a separate drawer. Both databases will likely be on the same server, and running the same version of the software – and thus are open to the same exploits. Second, both passwords generally have to reside in plaintext form in the code somewhere for the software to perform the join. If one is obtained – e.g., if the machine is hacked – the other is as well.

The Argument for Allowing ID Storage

We might argue to allow ID storage by appealing to precedent. Facebook and other SNSes implement policies to maintain user privacy, and we can draw on these professional best practices to inform our own. Such systems give IDs when the user joins the application, and the IDs of one degree 'friends' of that user. Applications use these IDs to index into the data stores of the providers, like Facebook or Twitter.

This precedent protects users: it is in the best interest of an SNS to guarantee privacy for its users. When an application does evil, it is the SNS that catches the heat. For example, one man on Facebook was recently surprised to find an ad suggesting "hot singles are waiting for you!" next to a picture of his wife [3]. Facebook got caught in an ensuing privacy firestorm. But, Facebook was not at fault: an application had used its API access to populate the ad within its own canvas page. Such threats to their brand may lead to enforcement of Facebook's policies.

The second appeal to precedent involves IRB policy. IRBs are generally fine with researchers harvesting information that is considered "public" by the user of your system [4]. Thus, YouTube videos, delicious tags, tweets and the like, which have the expectation of public visibility, don't worry the IRB. Facebook and Twitter IDs are now arguably public knowledge. If a user knows your screen name, they can find your ID. Or, if they know your real name, they can search Facebook or Twitter to find basic information about your online persona.

There is one more reason we should consider IDs relatively secure even if the database is hacked. Users authorize particular *applications* to have access to information. If our database is hacked by another developer, that developer cannot use the IDs to activate the API like we can, because the developer cannot spoof our application's API secret key. The secret key is resident in our private source code.

Obscure the Mapping

Encrypt IDs

Encrypt all user IDs in the database, for example by hashing them against a secret key. When the application needs to use an ID, it decrypts it in runtime code. If the database is hacked, no PII is released (IDs are encrypted). If the application or machine is hacked, PII is in danger (secret key is in the code).

Password on Startup

The application administrator gives a password on startup. This password deterministically generates a secret key to encrypt all user IDs in the database. If the database is hacked, no PII is released. If the application or machine is hacked, no PII is released. However, automated management scripts are no longer usable (they would need passwords on startup).

Keep the Mapping Offsite

Third-Party ID Mapping

A separate service manages encryption of SNS user IDs for all research applications, analogous to today's locked drawer. When the SNS gives the application a user ID, the application calls the service, which returns an encrypted ID for that user. That encrypted ID can be safely stored in the database. Whenever the application wants to call the SNS's API, it first calls the service to return the unencrypted user ID. The researcher takes responsibility for never storing true IDs on the server. This service could be maintained a trusted third party (e.g., ACM).

Buy-in from SNSes

The service maintaining the ID mapping could be the SNS itself. The SNS, e.g., Facebook, would always return encrypted user IDs to the application, and accept these encrypted IDs in its API calls.

Table 2. A proposed list of best practices to protect personally identifiable information.

The Argument Against Allowing ID Storage

A challenge with the preceding arguments is that they are tied to particular policies of current social networks. These policies will change. We must decide whether our IRB policies should migrate with them to adapt to changing online norms. We want consistency, but don't want to hamstring ourselves.

IRBs want to guarantee anonymity of who participated in the study. Even if basic profile information is public, the information about which of these profiles are active in the study is not public knowledge. If your database is hacked, then user IDs are essentially a list of study participants. This violates IRB policy.

The application's logs are the most dangerous. The Facebook applications we have built maintain central databases of information about people, much of which is not visible to non-friends. This information could damage the participant's reputation if disclosed outside the study. (Imagine it becoming public knowledge that your significant other recommended to you articles about a medical condition you keep quiet.)

Conclusion

We need to peel apart the real danger of storing Facebook IDs. There is danger associated with maintaining the ID, and danger associated with the information that the ID gives us access to via an API. We have argued that the first is a real concern, and that the second is less so. Having the ID accessible grants access to private information: who participated in the study, and what data they generated while using the system. The data accessible via API is less dangerous. Much of it is public already – I can generate a random numerical string and gather this data about the resulting Facebook or Twitter user. The user opts in to enable API access, so stolen IDs pose no danger unless the user grants access to a rogue application.

Where do we draw the line, and what solutions offer sufficient protection to participants while being technically tractable? We offer some potential solutions in Table 2.

The Facebook era is redefining personally identifiable information. This information is still just as identifiable, but it is also now more necessary for research code to function, more public than before, and the lines are now much blurrier between study data and personal information. As a field, we need a clear and consistent position on how to handle this data.

Acknowledgements

We thank MIT's Committee On the Use of Humans as Experimental Subjects (COUHES), and our users.

References

[1] Bernstein, M., et al. 2009. Collabio: A Game for Annotating People within Social Networks. In *Proc. UIST '09.* ACM Press, pp. 97-100.

[2] Good, N. S., et al. 2007. Noticing notice: a large-scale experiment on the timing of software license agreements. In *Proc. CHI '07*. ACM Press, pp. 607-616.

[3] Ostrow, A. Facebook Dating Ad Hooks Up Married Man … With His Wife. Mashable. http://mashable.com/2009/07/17/facebook-dating-ads-2/

[4] National Human Subjects Protection Advisory Committee. 2002. Recommendations on Public Use Data Files. http://hhs.gov/ohrp/nhrpac/documents/dataltr.pdf

[5] Schönberg, C-M., Boublil, A., and Kretzmer, H. 1985. "Who Am I?" Lyrics. Les Miserables Original London Cast Recording. Relativity, 1985.