Security and privacy preservation in human-involved networks

Craig Asher Jean-Philippe Aumasson Raphael C.-W. Phan

University of Applied Sciences Northwestern Switzerland
School of Engineering

Outline

Human-involved networks

Case study: online social networking services

Modelling security

Conclusions

Human-involved networks (HIN's)

Natural extension of computer networks

Nodes can be computers or human beings

Some tasks can only be performed by humans

Capture **out-of-band** links

- human entering a PIN on a device
- human-to-human discussion
- carrying a pile of DVD's
- Bluetooth

Human-involved networks (HIN's)

HIN's are emerging, and will last

Hot issues

- preserving users' **privacy**
- preventing **impersonation**

+ risk of "real-world" attacks: **ID theft**, etc.

Open problems

- how to best model security within HIN's?

Open problems

- how to best model security within HIN's?
- how to adapt provable security to HIN's?

Social networking services (SNS)

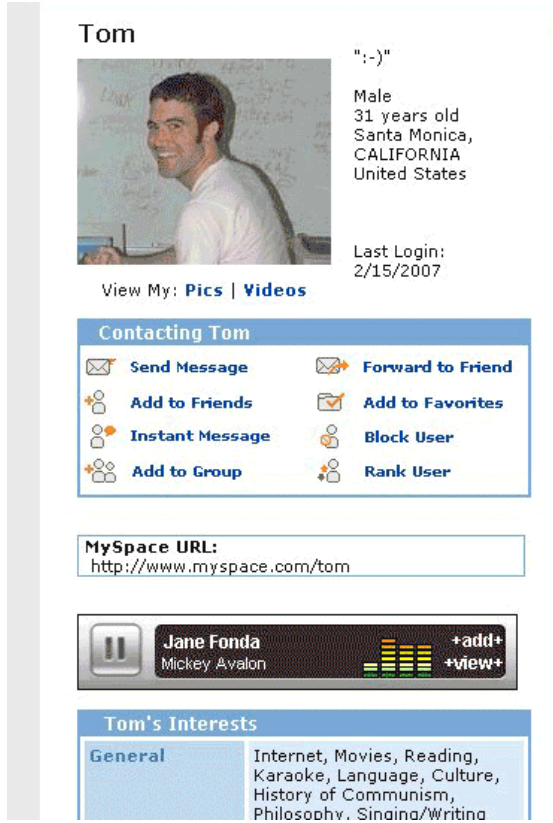
Websites on which members create a webpage to put personal and/or professional information

Interaction between members through

- contact lists
- discussion forums, chats, photos, etc.
- shared applications (e.g., agenda)
- Twitter-like live updates

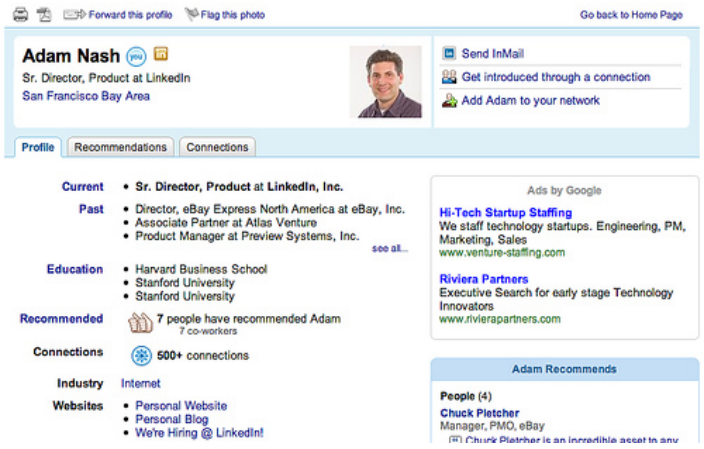
$\approx$10 sites with 50M+ members

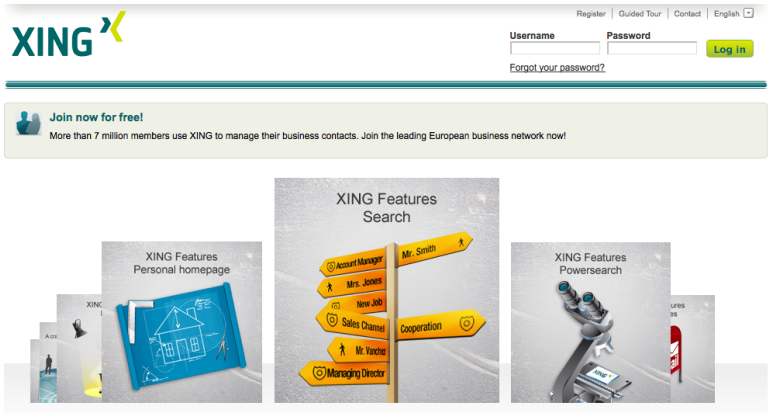
General...



Facebook, MySpace: 200M+ members

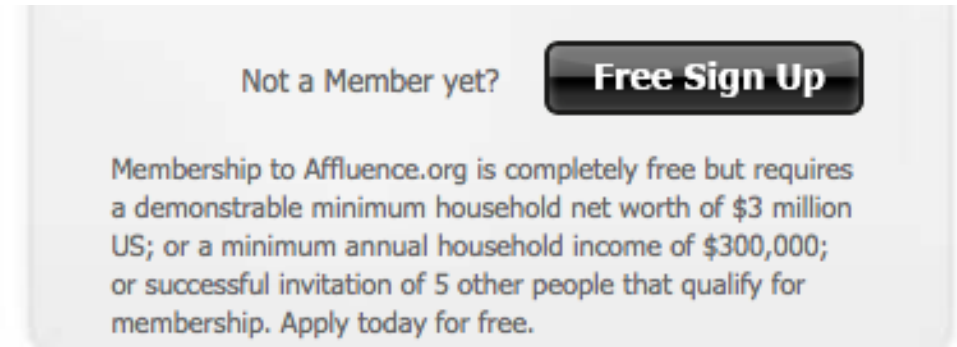
Professional...





LinkedIn: 35M

Xing: 6M



Other. . .

Privacy

Given for free:

- address, phone numbers
- list of friends/colleagues/relatives
- political and religious views
- photos, videos, etc.
- status (away on vacation, etc.)
- hobbies, interests

Public info crawled by search engines. . .

Privacy

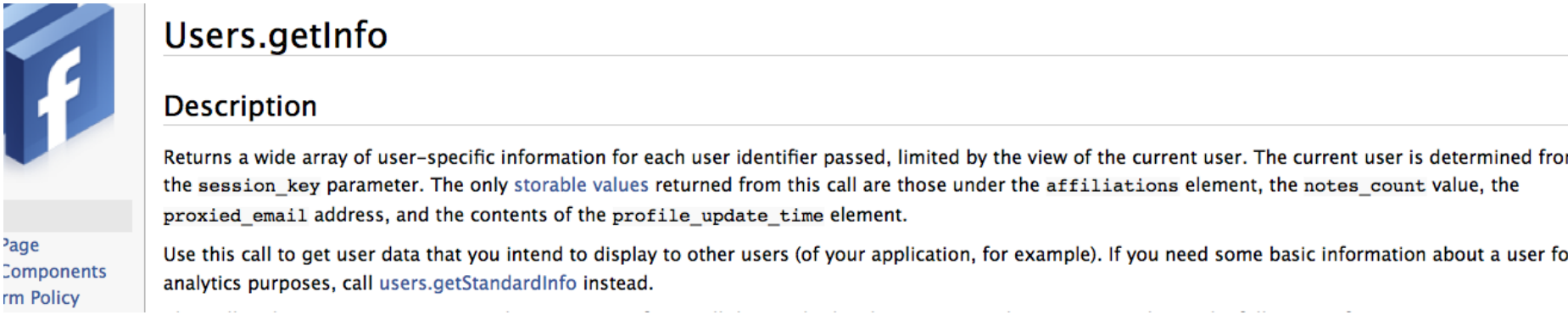
Access reserved to legitimate users, e.g., “friend”, or “friend of friend”, but. . .

- easy to get in the friends circle
- even easier to be second-degree friend

For outsiders, many leaks exploitable. . .

- simple tricks
- sloppy implementations
- thin separation users/developers

Facebook's `Users.getInfo` command



Page
Components
rm Policy

Users.getInfo

Description

Returns a wide array of user-specific information for each user identifier passed, limited by the view of the current user. The current user is determined fro
the `session_key` parameter. The only storable values returned from this call are those under the `affiliations` element, the `notes_count` value, the
`proxied_email` address, and the contents of the `profile_update_time` element.

Use this call to get user data that you intend to display to other users (of your application, for example). If you need some basic information about a user fo
analytics purposes, call users.getStandardInfo instead.

User Privacy and Visible Data

Important: Depending upon the user's privacy settings
submitted to this method, the following user fields are

- meeting_for
- meeting_sex
- religion
- significant_other_id

Available to application developers

= potentially anyone

Facebook's photos storage

[Bonneau-02/2009]

Photos hosted on external servers, e.g.,

`http://photos-c.ak.fbcdn.net/photos-ak-snc1/v2601/191/...`

Low entropy in photos URL's

$\Rightarrow$ leakage of private photos

hi5 privacy policy

If you decide to use one of the additional services that are offered by our partners, we may forward Personal Information to these partners to enable them to provide the services that you requested.
We also provide information to third-party advertising companies, as described in the next section.
Please be aware that the handling of your Personal Information by our partners or the third-party advertising companies is governed by their privacy policy, not ours.

= they do whatever they want

Google's Orkut

Privacy settings **open by default** (let unchanged by 90%)

Impersonation

- breaking into someone's account
- creating a fake account (trivial)

Enforcement based on complaints by users. . .

(but what if users collude against someone?)

Non-trivial impersonation attack

[Elgan-11/2008]

Considers two networks X and Y:

1. befriend with a stranger both on X and Y
2. spot his friends that are on X but not on Y
3. use info from X to forge fake profiles on Y
4. send friend request to the stranger

SNS need a model to. . .

- identify weaknesses more easily
- design countermeasures
- minimize privacy leaks
- complicate impersonation

$\Rightarrow$ design protocols that exploit humans' capabilities

$\Rightarrow$ use a (semi-) formal model, rather than improvising ad hoc countermeasures

Protocols for humans: ceremonies

Ceremonies $=$ protocols for HIN's [Ellison-2007]

"we don't program humans the way we do computers, and when we try, the attempt usually fails"

Protocols for humans: ceremonies

Ceremonies $=$ protocols for HIN's [Ellison-2007]

"we don't program humans the way we do computers, and when we try, the attempt usually fails"

Human node:

- state machine with memory
- receives and sends messages
- depends on a computer interface
- error-prone

Examples of ceremonies

Connection to e-banking

Authentication with SAS

POSHes (Puzzles Only Solvables by Humans)

TO COMPLETE YOUR WEB REGISTRATION, PLEASE PROVE THAT YOU'RE HUMAN:

WHEN LITTLEFOOT'S MOTHER DIED IN THE ORIGINAL 'LAND BEFORE TIME,' DID YOU FEEL SAD?

- YES
- NO

(BOTS: NO LYING)

Ceremonies for web-authentication

[Karlof-Tyger-Wagner-2009]

Associative learning of safe rules

Bad:
If [legitimate looking form] then [enter password]

Better:
Don’t teach users to distinguish real from fake, but rather condition them to make safe decisions

Crypto adversarial models

Adversaries/parties assumed human, notions as

- party corruption
- honest-but-curious
- malicious

Human as a non-constructive entity; protocols work fine with machines only

Crypto adversarial models

Key-agreement [Canetti/Krawcyz-01]

- `Session-state reveal`
- `Session-key query`
- `Party corruption`, etc.

RFID protocols [Vaudenay-07]

- `CreateTag`
- `SendReader`
- `Corrupt`, etc.

$\approx$ simple send/receive models

Model for SNS

Need to capture:

- actions proper to SNS (“friends” list, etc;)
- out-of-band interactions
- attacks involving several networks (use a network ID: NID)

Need be general enough to model different (similar) networks

Sketch of a model for SNS

- `Register`(NID, email)
- `GetPublicInfo`(NID, ID1, ID2)
- `GetPrivateInfo`(NID, ID1, ID2)
- `GetContacts`(NID, ID1, ID2)
- `ContactRequest`(NID, ID1, ID2)
- `OutOfBandInteract`(ID1, ID2)
- `Corrupt`(NID, ID1, ID2)

High-level model

Need to refine wrt the network(s) considered. . .

The impersonation attack

AttackerX $\leftarrow$ `Register`(X, `dummy@email`)
AttackerY $\leftarrow$ `Register`(Y, `dummy@email`)

until $(x, y) \neq$(success, success)
  ID1 $\leftarrow$ random name
  x $\leftarrow$ `GetPublicInfo`(X, attackerX, ID1)
  y $\leftarrow$ `GetPublicInfo`(Y, attackerY, ID1)

$\mathcal{S} \leftarrow$ `GetFriends`(X, ID, ID1) $\setminus$ `GetFriends`(Y, ID, ID1)

for all ID2's in $\mathcal{S}$
  `GetPublicInfo`(X, ID, ID2)
  ID3 $\leftarrow$ `Register`(Y, `ID2@forged.mail`)
  `FriendRequest`(Y, ID3, ID1)

Provable security

- information theoretical
- computational (via **reductions**)

Efficient “break” of the scheme $\Rightarrow$ efficient algorithm for solving some hard problem

Common assumptions: hardness of integer factoring, discrete log, Diffie-Hellman, etc.

SNS: no computational hardness assumption

Provable security

Show for example

- accessing one's private data requires authentication as a degree-1 contact
- what can (not) be done by forging N fake accounts

Or give impossibility results

- breaking into one's account doesn't require to know his password
- can't guarantee that private data remain within a bounded-degree contact circle

How to design SNS that admit proofs?

Conclusions

HIN's can't be designed/analyzed classically

Proposal:

- **ceremonies** framework
- adapted (semi-) **formal models**
- long way to provable security...

Conclusions

Future work:

- modelling human behavior? (psychology, cognitive sciences. . .)
- design of a provably secure SNS?
  - does it really make sense?
  - impossibility results?

Security and privacy preservation in human-involved networks

Craig Asher Jean-Philippe Aumasson Raphael C.-W. Phan

University of Applied Sciences Northwestern Switzerland
School of Engineering