Towards a mechanism for incentivating privacy

Piero Bonatti, Marco Faella, Clemente Galdi, Luigi Sauro

Università di Napoli “Federico II”, Italy

Leuven, 14/9/2011

Introduction

The economic value of user profiles

- Rich user profiles = Money
- An incentive for providers to collect lots of personal (sensitive) information (and sell it!)
  - user name, birth date, gender, detailed address, credit card information

Introduction

The economic value of user profiles

- Rich user profiles = Money
- An incentive for providers to collect lots of personal (sensitive) information (and sell it!)
  - user name, birth date, gender, detailed address, credit card information
  - lots of *quasi-identifiers*

Introduction

The economic value of user profiles

- Rich user profiles = Money
- An incentive for providers to collect lots of personal (sensitive) information (and sell it!)
  - user name, birth date, gender, detailed address, credit card information
  - lots of *quasi-identifiers*
  - even sex preferences, and political and religious views

Introduction

Privacy-related questions

- Is *all* of the profile *necessary* for deploying services effectively and securely ?

Introduction

Privacy-related questions

- Is *all* of the profile *necessary* for deploying services effectively and securely ?
- Is anything preventing providers from collecting more and more information ?

Introduction

Privacy-related questions

- Is *all* of the profile *necessary* for deploying services effectively and securely ?
- Is anything preventing providers from collecting more and more information ?
- Is there any mechanism for minimizing provider requests?

Introduction

Privacy through competition

- Many people do care about privacy
  - large groups of Facebook users threatened to leave and join other networks several times
  - Facebook had to stop and reshape some of its new services

Introduction

Privacy through competition

- Many people do care about privacy
  - large groups of Facebook users threatened to leave and join other networks several times
  - Facebook had to stop and reshape some of its new services
- Several analysts say that privacy may become a factor of competition

Introduction

Privacy through competition

- Many people do care about privacy
  - large groups of Facebook users threatened to leave and join other networks several times
  - Facebook had to stop and reshape some of its new services
- Several analysts say that privacy may become a factor of competition
- Our ultimate goal:
  - *developing mechanisms that moderate profile collection through provider competition*

The first step

(this paper)

- Truthful mechanisms
  - i.e. providers ask for the user information they *really* need
  - because that's the best strategy

The first step

(this paper)

- Truthful mechanisms
  - i.e. providers ask for the user information they *really* need
  - because that's the best strategy
- Second-price auctions (a.k.a. Vickrey's auctions)
  - perhaps the most popular truthful mechanism

The first step

(this paper)

- Truthful mechanisms
  - i.e. providers ask for the user information they *really* need
  - because that's the best strategy
- Second-price auctions (a.k.a. Vickrey's auctions)
  - perhaps the most popular truthful mechanism
- Technical problems
  - our "currency" (profiles) is only *partially* ordered
  - there is no "second price"

The first step

(this paper)

- Truthful mechanisms
  - i.e. providers ask for the user information they *really* need
  - because that's the best strategy
- Second-price auctions (a.k.a. Vickrey's auctions)
  - perhaps the most popular truthful mechanism
- Technical problems
  - our "currency" (profiles) is only *partially* ordered
  - there is no "second price"
- First technical investigation
  - Is there any truthful mechanism compatible with the structure of our scenarios ?

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service
  2. Providers respond with their information requests, e.g. {*login, password*} *or* {*credit-card, ID*}

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service
  2. Providers respond with their information requests, e.g. {*login, password*} *or* {*credit-card, ID*}
  3. User selects provider (user $\sim$ auctioneer, providers $\sim$ bidders)

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service
  2. Providers respond with their information requests, e.g. $\{login, password\}$ *or* $\{credit\text{-}card, ID\}$
  3. User selects provider (user $\sim$ auctioneer, providers $\sim$ bidders)

- Information items (called *credentials*) are not equally sensitive
  - $\{prepaid\text{-}card\} \prec \{birthdate, zip\}$ (strict partial order)

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service
  2. Providers respond with their information requests, e.g. $\{login, password\}$ *or* $\{credit\text{-}card, ID\}$
  3. User selects provider (user $\sim$ auctioneer, providers $\sim$ bidders)

- Information items (called *credentials*) are not equally sensitive
  - $\{prepaid\text{-}card\} \prec \{birthdate,zip\}$ (strict partial order)

- Simplifying assumptions (to be dropped)
  - providers offer functionally equivalent services
  - information-disclosure costs only (e.g. flight booking like Kayak, Momondo, ...)

The Formal Framework – Auction-like mechanism

V 0.0

- Protocol:
  1. User asks for a service
  2. Providers respond with their information requests, e.g. $\{login, password\}$ *or* $\{credit\text{-}card, ID\}$
  3. User selects provider (user $\sim$ auctioneer, providers $\sim$ bidders)

- Information items (called *credentials*) are not equally sensitive
  - $\{prepaid\text{-}card\} \prec \{birthdate, zip\}$ (strict partial order)

- Simplifying assumptions (to be dropped)
  - providers offer functionally equivalent services
  - information-disclosure costs only (e.g. flight booking like Kayak, Momondo, ...)

$\Downarrow$

  - users choose providers based on information requests only
  - repeated service usage has no additional costs

The Formal Framework – User privacy constraints

V 0.0

- User privacy constraints (user policy): maximal disclosable sets
  - {*zip,nationality*} *or* {*credit-card, birthdate*}

The Formal Framework – User privacy constraints

V 0.0

- User privacy constraints (user policy): maximal disclosable sets
  - {*zip,nationality*} *or* {*credit-card, birthdate*}
  - *zip* is OK; *credit-card* + *birthdate* is OK

The Formal Framework – User privacy constraints

V 0.0

- User privacy constraints (user policy): maximal disclosable sets
  - {*zip,nationality*} *or* {*credit-card, birthdate*}
  - *zip* is OK; *credit-card* + *birthdate* is OK
  - *zip* + *birthdate* not releasable

The Formal Framework – User privacy constraints

V 0.0

- User privacy constraints (user policy): maximal disclosable sets
  - {*zip,nationality*} *or* {*credit-card, birthdate*}
  - *zip* is OK; *credit-card* + *birthdate* is OK
  - *zip* + *birthdate* not releasable
- Admissible requests
  - Let *adm* be the set of all requests (sets of items) that satisfy the user's privacy preferences

The Formal Framework – Provider policy

V 0.0

- Provider policy: minimal acceptable sets (for service access)
  - {*login,password*} *or* {*credit-card, exp-date,username,...*}

The Formal Framework – Provider policy

V 0.0

- Provider policy: minimal acceptable sets (for service access)
  - {*login*,*password*} *or* {*credit-card*, *exp-date*,*username*,...}
  - *login* + *password* + *credit-card* is OK

The Formal Framework – Provider policy

V 0.0

- Provider policy: minimal acceptable sets (for service access)
  - {*login*,*password*} *or* {*credit-card*, *exp-date*,*username*,...}
  - *login* + *password* + *credit-card* is OK
  - *login* + *credit-card* not enough

The Formal Framework – Provider policy

V 0.0

- Provider policy: minimal acceptable sets (for service access)
  - {*login*,*password*} *or* {*credit-card*, *exp-date*,*username*,...}
  - *login* + *password* + *credit-card* is OK
  - *login* + *credit-card* not enough
- Fulfilling disclosures
  - Let $ful(pol_i)$ be all sets of items that satisfy provider *i*'s policy

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request ≠ policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request ≠ policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)
- Providers may ask for larger information sets
  - {*credit-card, ID, SSN*} *or ...*

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request* ≠ *policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)
- Providers may ask for larger information sets
  - {*credit-card, ID, SSN*} *or ...*
- Providers may omit alternatives
  - e.g. omit *student-id* because *passport* is "richer"
  - ~~{*credit-card, student-id*}~~ *or* {*credit-card, passport*}

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request* ≠ *policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)
- Providers may ask for larger information sets
  - {*credit-card, ID, SSN*} *or ...*
- Providers may omit alternatives
  - e.g. omit *student-id* because *passport* is "richer"
  - ~~{*credit-card, student-id*}~~ *or* {*credit-card, passport*}
- A strategy req_i is *truthful* if $req_i = pol_i$

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request* ≠ *policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)
- Providers may ask for larger information sets
  - {*credit-card, ID, SSN*} *or ...*
- Providers may omit alternatives
  - e.g. omit *student-id* because *passport* is "richer"
  - ~~{*credit-card, student-id*}~~ *or* {*credit-card, passport*}
- A strategy req_i is *truthful* if $req_i = pol_i$
- Users must release a set in $ful(req_i)$

The Formal Framework – Provider *requests* (strategies)

V 0.0

- *Request* ≠ *policy*
  - they have the same structure, though (a list of info sets)
  - req_i denotes the information request of provider *i* (its *strategy*)
- Providers may ask for larger information sets
  - {*credit-card, ID, SSN*} *or ...*
- Providers may omit alternatives
  - e.g. omit *student-id* because *passport* is "richer"
  - ~~{*credit-card, student-id*}~~ *or* {*credit-card, passport*}
- A strategy req_i is *truthful* if $req_i = pol_i$
- Users must release a set in $ful(req_i)$
- Each set in req_i must be in $ful(pol_i)$

Provider goals

- Which information sets do they prefer?
  - larger (w.r.t. $\subseteq$)
  - more sensitive (w.r.t. $\prec$)
    - hypothesis: more sensitive $\Rightarrow$ more valuable

Provider goals

- Which information sets do they prefer?
  - larger (w.r.t. $\subseteq$)
  - more sensitive (w.r.t. $\prec$)
    - hypothesis: more sensitive $\Rightarrow$ more valuable
- What are their priorities?
  - getting preferred info sets
  - winning (i.e. being selected)

Profiles

- A profile π is a vector that summarizes the whole scenario
  - user policy
  - all provider policies, strategies, and preferences

The mechanism

- Candidate winners $cw(\pi)$
  - those who make an optimal request in the current scenario π
  - $req_i \cap opt(\pi) \neq \emptyset$

$$opt(\pi) = \min_{\prec} \left(\bigcup_{j=1}^{N} req_j \cap adm \right)$$

The mechanism

- Candidate winners $cw(\pi)$
  - those who make an optimal request in the current scenario π
  - $req_i \cap opt(\pi) \neq \emptyset$

$$opt(\pi) = \min_{\prec} \left(\bigcup_{j=1}^{N} req_j \cap adm \right)$$

The mechanism

- Candidate winners $cw(\pi)$
  - those who make an optimal request in the current scenario π
  - $req_i \cap opt(\pi) \neq \emptyset$

$$opt(\pi) = \min_{\prec} \left(\bigcup_{j=1}^{N} req_j \cap adm \right)$$

The mechanism

- Candidate winners $cw(\pi)$
  - those who make an optimal request in the current scenario π
  - $req_i \cap opt(\pi) \neq \emptyset$

$$opt(\pi) = \min_{\prec} \left(\bigcup_{j=1}^{N} req_j \cap adm \right)$$

1. Choose some provider $i \in cw(\pi)$ (randomly)

The mechanism

- Candidate winners $cw(\pi)$
  - those who make an optimal request in the current scenario π
  - $req_i \cap opt(\pi) \neq \emptyset$

$$opt(\pi) = \min_{\prec} \left(\bigcup_{j=1}^{N} req_j \cap adm \right)$$

1. Choose some provider $i \in cw(\pi)$ (randomly)
2. Choose a set of credentials from $res(\pi, i)$ and disclose it to i
   - if $res(\pi, i) = \emptyset$ the transaction fails
   - how to define $res(\pi, i)$?

The right notion of response

- Some definitions introduce additional failures (see the paper)
- Some don't, but release lots of information items (see the paper)
- Other variants make it profitable to lie
- Vaults are the best solution so far
  - the largest admissible responses that are not more sensitive than any other provider's request

$$vault(\pi, i) = \max_{\subseteq}\left\{r \mid r \in adm \wedge \forall r' \in opt_{-i}(\pi).\ r' \not\prec r\right\}.$$

The right notion of response

- Some definitions introduce additional failures (see the paper)
- Some don't, but release lots of information items (see the paper)
- Other variants make it profitable to lie
- Vaults are the best solution so far
  - the largest admissible responses that are not more sensitive than any other provider's request

$$vault(\pi, i) = \max_{\subseteq} \left\{ r \mid r \in adm \wedge \forall r' \in opt_{-i}(\pi).\ r' \not\prec r \right\}.$$

The right notion of response

- Some definitions introduce additional failures (see the paper)
- Some don't, but release lots of information items (see the paper)
- Other variants make it profitable to lie
- Vaults are the best solution so far
  - the largest admissible responses that are not more sensitive than any other provider's request

$$vault(\pi, i) = \max_{\subseteq}\left\{r \mid r \in adm \wedge \forall r' \in opt_{-i}(\pi).\ r' \not\prec r\right\}.$$

The right notion of response

- Some definitions introduce additional failures (see the paper)
- Some don't, but release lots of information items (see the paper)
- Other variants make it profitable to lie
- Vaults are the best solution so far
  - the largest admissible responses that are not more sensitive than any other provider's request

$$vault(\pi, i) = \max_{\subseteq}\left\{r \mid r \in adm \wedge \forall r' \in opt_{-i}(\pi).\ r' \not\prec r\right\}.$$

The right notion of response

- Some definitions introduce additional failures (see the paper)
- Some don't, but release lots of information items (see the paper)
- Other variants make it profitable to lie
- Vaults are the best solution so far
  - the largest admissible responses that are not more sensitive than any other provider's request

$$vault(\pi, i) = \max_{\subseteq} \left\{ r \mid r \in adm \wedge \forall r' \in opt_{-i}(\pi).\ r' \not\prec r \right\}.$$

- Responses must also fulfil some of i's optimal requests

$$res(\pi, i) = vault(\pi, i) \cap ful(opt(\pi) \cap req_i)\,.$$

Analogy with second price

Vickrey's auctions

The winner pays the minimum price that is not worse (i.e., smaller) than any other offer (and satisfies the winner's request)

Vault-based mechanism

The winner gets a maximal response that is not worse (i.e., more sensitive) than any other offer, and satisfies both the user's policy and the winner's request

Results

comparison with other *res* we tried

- The vault-based definition of *res*
  - does not fail if at least one provider makes an admissible request
  - it never releases more information than the other response functions with the same property

Results

releasing maximal admissible sets

- In general, a provider may get more than what it asked for
  - as in 2nd price auctions
  - the price to pay for truthfulness
  - nonetheless...

Results

releasing maximal admissible sets

- In general, a provider may get more than what it asked for
  - as in 2nd price auctions
  - the price to pay for truthfulness
  - nonetheless...
- The vault-based definition of *res* may release a maximal admissible set *r* only if
  - either there is no competition
  - or some *j* asks exactly for *r*
    - in practice, systematic exploitation requires exact knowledge of user preferences

Results

truthfulness

- The vault-based mechanism is truthful, i.e. $req_i = pol_i$ is the most effective strategy
  - both for the providers that give higher priority to getting more preferred sets (larger or more sensitive)
  - and for the providers that give higher priority to winning

Results

truthfulness

- The vault-based mechanism is truthful, i.e. $req_i = pol_i$ is the most effective strategy
  - both for the providers that give higher priority to getting more preferred sets (larger or more sensitive)
  - and for the providers that give higher priority to winning
- Knowledge about the other agents' behavior does *not* affect truthfulness

Results

truthfulness

- The vault-based mechanism is truthful, i.e. $req_i = pol_i$ is the most effective strategy
  - both for the providers that give higher priority to getting more preferred sets (larger or more sensitive)
  - and for the providers that give higher priority to winning
- Knowledge about the other agents' behavior does *not* affect truthfulness
- (Minimal disclosures) If all providers have the same policy
  - by exhogenous technological constraints
  - e.g. because they support the same credit card companies

and i is rational/truthful, then:

Results

truthfulness

- The vault-based mechanism is truthful, i.e. $req_i = pol_i$ is the most effective strategy
  - both for the providers that give higher priority to getting more preferred sets (larger or more sensitive)
  - and for the providers that give higher priority to winning
- Knowledge about the other agents' behavior does *not* affect truthfulness
- (Minimal disclosures) If all providers have the same policy
  - by exhogenous technological constraints
  - e.g. because they support the same credit card companies

  and i is rational/truthful, then:
  - all other agents $j \neq i$ can get only elements of pol_j
  - if some $k \neq i$ is rational/truthful, too, then *all* providers j can get only elements of pol_j

Related work

nothing really similar

- In trust negotiation
  - no equivalent to pol_i: TN policies $\approx$ req_i
  - no attempt to minimize provider requests

Related work

nothing really similar

- In trust negotiation
  - no equivalent to pol_i: TN policies $\approx$ req_i
  - no attempt to minimize provider requests
- In [Feigenbaum et al 2010] the goal is minimizing the information that *bidders* (providers) have to disclose to the auctioneer

Related work

nothing really similar

- In trust negotiation
  - no equivalent to pol_i: TN policies $\approx$ req_i
  - no attempt to minimize provider requests
- In [Feigenbaum et al 2010] the goal is minimizing the information that *bidders* (providers) have to disclose to the auctioneer
- In [Kleinberg et al 2001] the goal is inducing users to release *more* (and more accurate) information about their preferences, by means of compensation

Related work

nothing really similar

- In trust negotiation
  - no equivalent to pol_i: TN policies $\approx$ req_i
  - no attempt to minimize provider requests
- In [Feigenbaum et al 2010] the goal is minimizing the information that *bidders* (providers) have to disclose to the auctioneer
- In [Kleinberg et al 2001] the goal is inducing users to release *more* (and more accurate) information about their preferences, by means of compensation
- To the best of our knowledge, no auction mechanism deals with partially ordered payment means.

Conclusion

Achievements

- Competition between equivalent applications provably minimizes the amount of personal information requested by rational providers

Conclusion

Achievements

- Competition between equivalent applications provably minimizes the amount of personal information requested by rational providers
- Possible applications
  - preventing attacks to TN strategies that gradually extract all releasable information from the user agent

Conclusion

Achievements

- Competition between equivalent applications provably minimizes the amount of personal information requested by rational providers
- Possible applications
  - preventing attacks to TN strategies that gradually extract all releasable information from the user agent
  - enhancing the privacy of profile transfers (as in OpenID)
    - transfer only what the new provider asks for (minimized through competition)

Conclusion

Future work: A long to-do list (details in the paper)

- Introduce service costs, functional differences, quality of service...
  - information requests are not the only choice criterion any longer
  - opportunities for compensation and negotiation/repeated auctions

Conclusion

Future work: A long to-do list (details in the paper)

- Introduce service costs, functional differences, quality of service...
  - information requests are not the only choice criterion any longer
  - opportunities for compensation and negotiation/repeated auctions
- Deployment issues
  - Providing guarantees to *providers*, e.g.
    - Cryptographic protocols for checking that the user carries out the auction correctly (e.g. via commitments & blind signatures, secure multiparty computations)
    - Trusted third parties: a new role for portals like Kayak, Momondo etc.?

The End

Question time