Cryptographic Treatment of Private User Profiles

Felix Günther, Mark Manulis, Thorsten Strufe
TU Darmstadt & CASED

CRYPTOGRAPHIC PROTOCOLS

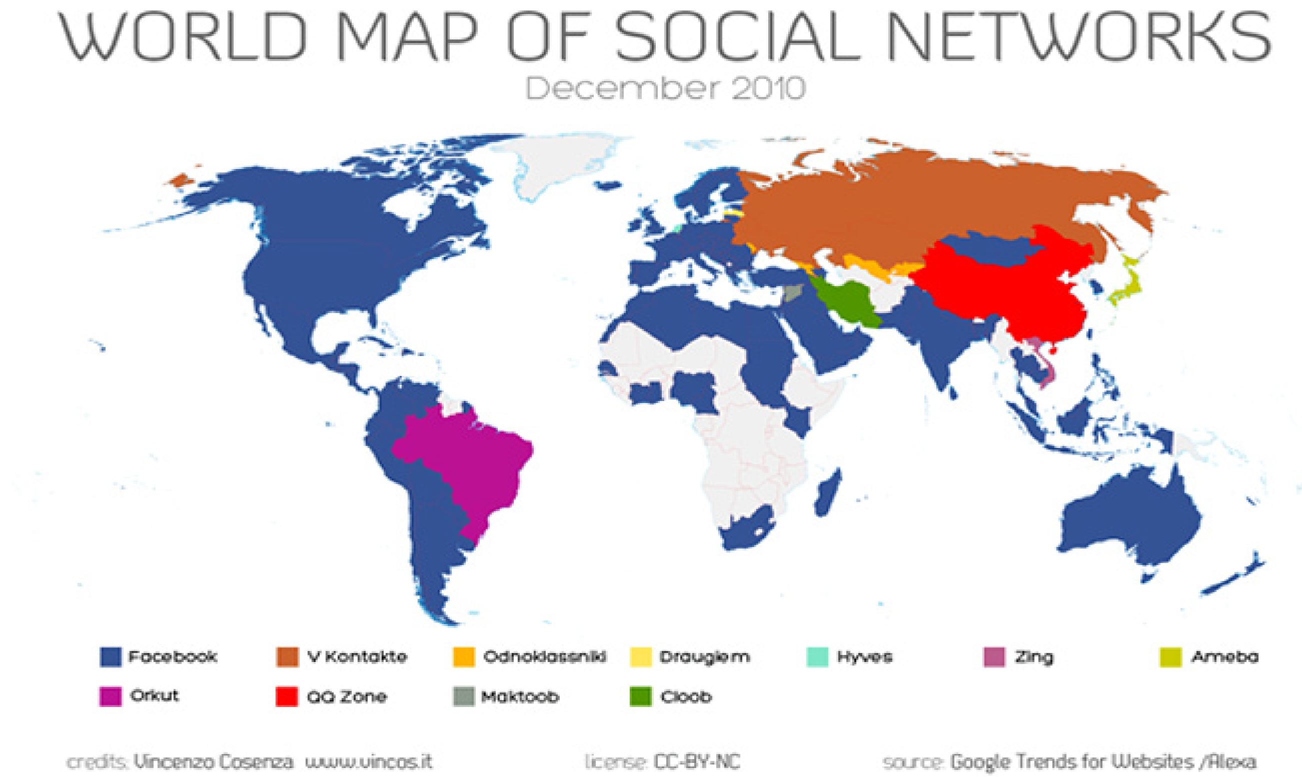
Users and the Web

Modern Web is dominated by social interaction, networking, online communities.

Services are offered by the users to the users. **Users are at the heart of the Web.**

Social Interaction: Global Phenomenon

Social networking on the Web enjoys popularity all over the world.



Facebook alone attracts more than 500 Mio. users with 50% logging users per day.

Web’s Ultimate Time Sink

The amount of time users spend on social interaction on the Web is increasing.

Top 10 Web Brands for January 2010 (U.S., Home and Work)					
RANK	Brand	Unique Audience (000)	Time Per Person (hh:mm:ss)	MOM UA % Change	MOM Time % Change
1	Google	152,708	1:23:54	4.10%	-16.90%
2	Yahoo!	134,561	2:09:14	4.30%	-26.80%
3	Facebook	116,329	7:01:41	5.80%	9.70%
4	MSN/WindowsLive/Bing	109,425	1:35:33	1.20%	-18.10%
5	YouTube	99,525	1:02:27	7.60%	-10.30%
6	AOL Media Network	82,306	1:01:14	-6.80%	-57.80%
7	Wikipedia	64,917	0:15:59	10.70%	-2.70%
8	Fox Interactive Media	62,112	1:23:28	1.00%	-9.10%
9	Amazon	60,772	0:22:34	-8.60%	-32.90%
10	Ask Search Network	57,776	0:12:35	10.70%	-11.40%
Source: The Nielsen Company					

User-Provided Content

Social interaction on the Internet proceeds on the basis of user-provided content.

User-Provided Content

- personal information: name, contact details, affiliation, …
- digital content: photographs, videos, text comments, …

Social Interaction Activities

- publish/modify own information and data
- retrieve information and data published by other users
- expand own social connectivity (make new contacts/friends)
- communicate with other users (synchronous, asynchronous)

User-provided content and social interaction may leak information about users.

Privacy of users and their data has been recognized as a major threat.

Cryptographic Approach to Privacy

Privacy comes always in a context of an application. There can be no general solution.

Many papers on attacks against user privacy (de-anonymization of users, profiling, ...)
Expected results since **All information about users is public. Just take it and analyze.**

The actual research challenge is: **How to protect user privacy?**

Benefits of Cryptographic Approach for Privacy-Protection

- Treat application as a building block.
- Formal model — Define precisely what privacy means for this application.
- Formal proofs — Design privacy mechanisms that provably fulfill these goals.

Providers can change but users and applications remain the same.

Privacy mechanisms should be independent of the network infrastructure.

Users are at the heart of social networks — not the network itself.

Privacy mechanisms should not rely on any trusted third parties.

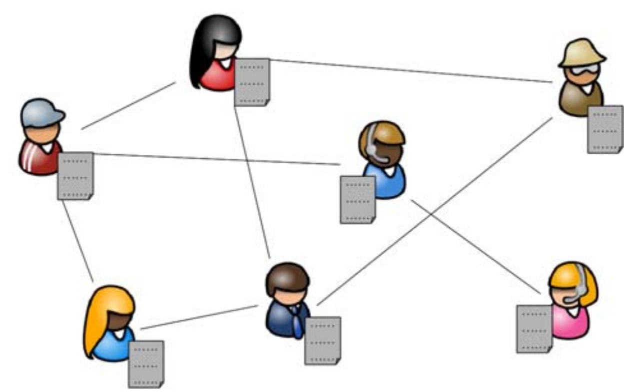
Our Focus: User Profiles

User Profiles

- the core functionality of any social platform
- in social communities: user = profile
- profiles are owned by users
- asynchronous access (owners can be offline)
- (ideally) profiles should be migratable

Main Functionality behind User Profiles

- publish of personal information, digital content
- fine-grained access control
  - grant permission to access portions of a profile
  - revocation of access rights
- social interaction through retrieval of other users' profile data



Related Work

Privacy in user profiles has been considered in the past…

… yet, without rigorous modeling / analysis (especially not for privacy).

Non-cryptographic approaches

[Carminati-Ferrari-Perego 2009]

- access control for social networks based on semantic rules and proofs
- assumes semi-centralized infrastructure, synchronous communication

(Ad-hoc) Cryptographic solutions

[Lucas-Borisov 2009]

- centralized approach, requires trust into the provider, no formal requirements/model
- tailored for use in established OSNs, e.g. Facebook

[Graffi et al. 2008, Baden et al. 2009 (OSN Persona)]

- uses attribute-based encryption, no formal analysis, only confidentiality

[Jahid-Mittal-Borisov 2011 (EASiER)]

- uses attribute-based encryption, semi-trusted server, only confidentiality

Formal Model for User Profiles

A **profile** P is modeled as a set of pairs

$$P \stackrel{\text{def}}{=} \{(a, đ) \mid a \in \mathcal{I}, đ \in \{0,1\}^*\}$$

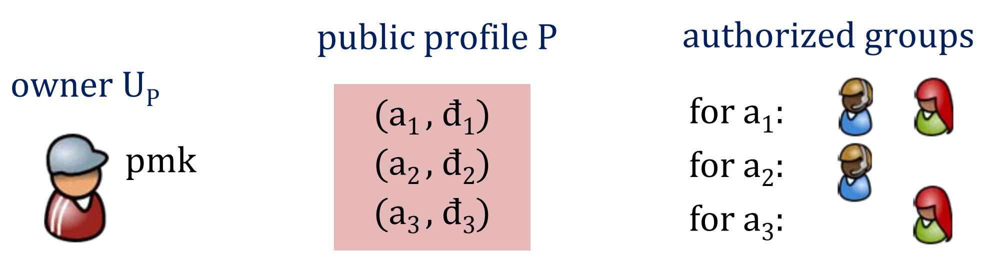
$\mathcal{I}$ is the set of possible **attribute indices** a (each a is unique per profile)

đ is the corresponding **value** stored in P.

P is assumed to be public but authenticated by its **owner** U_P

U_P has a **profile management key** pmk.

U_P given $(a, đ) \in P$ knows **attribute** d and **group** G of users authorized to access a.



indices can also be pseudonyms if one wants to protect the type of data

Examples

a = Date of Birth	a = 2123
đ = %$%§§“!	đ = §#~&“$
d = 01.01.1986	d = Bart Simpson

Profile Management Scheme

A **profile management scheme** PMS consists of

Init(κ)	Initializes P and outputs pmk.
Publish(pmk, P, (a, d), G)	Adds (a, đ) to P. Outputs *retrieveal key* rk_U for each U ∈ G.
Retrieve(rk_U, P, a)	Outputs either attribute d or ⊥.
Delete(pmk, P, a)	Removes (a, đ) from P (if such pair exists).
ModifyAccess(pmk, P, a, U)	Either grants or revokes access for U to (a, đ) ∈ P.

pmk may be updated by Publish, Retrieve, Delete, and ModifyAccess.
rk_U may be updated by Publish, Delete, and ModifyAccess.

If U_P published (a, đ) in P and did not delete it and some U has (unrevoked) access rights for index a (as part of its rk_U) then U can retrieve attribute d.

Adversary Model

We model security and privacy of PMS using (flexible) game-based approach.

PPT adversary $\mathcal{A}$ interacts with users and their public profiles using queries:

Corrupt(U)	Full corruption of U. Returns pmk and all rk_U of U.
Publish(P, (a, d), G)	U_P publishes (a, d) and grants access rights to users in G.
Retrieve(P, a,U)	Retrieves attribute with index a from P on behalf of U.
Delete(P, a)	U_P deletes (a, đ) from P (if such pair exists).
ModifyAccess(P, a, U)	U_P either grants or revokes access for U to (a, đ) $\in$ P.

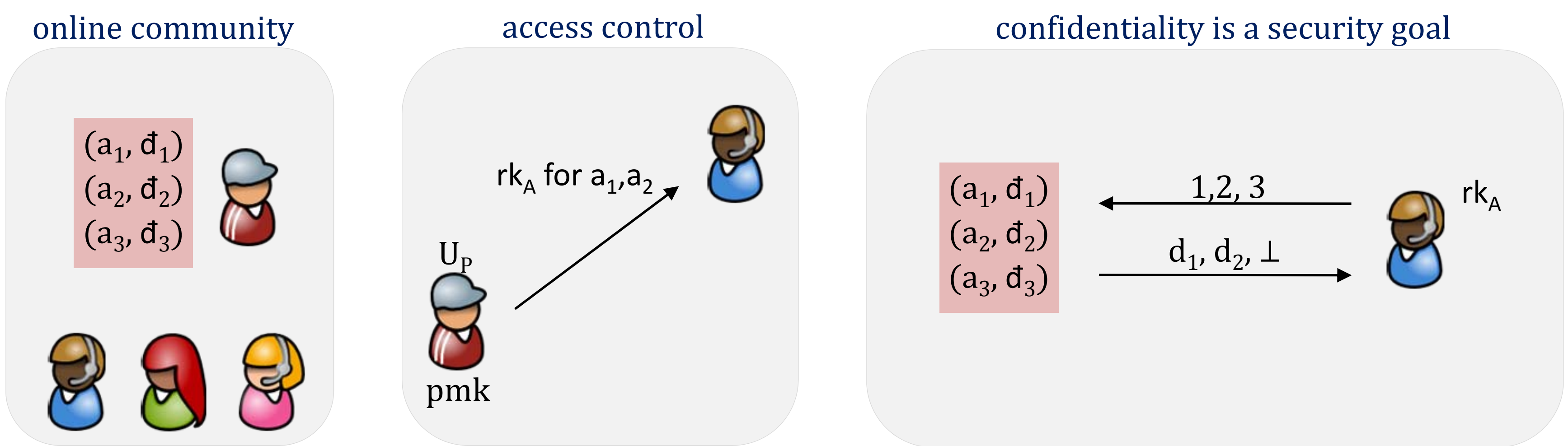
$\mathcal{A}$ is assumed to have any-time (read) **access to all profiles** in the system.

$\mathcal{A}$ is **adaptive** and can take control over all profiles and attributes using queries.

Security Goal: Confidentiality of Profile Data

U_P publishes pairs (a, đ) in P and gives U retrieval key rk_U for some indices a.

Confidentiality Attributes d should remain hidden from unauthorized users.



Indistinguishability approach:

$\mathcal{A}$ without access rights to (a, đ) should not be able to distinguish which attribute d is encrypted in đ.

.... even if $\mathcal{A}$ can access other attributes in the same profile.

Formal Definition of PMS Confidentiality

Confidentiality Game (high level)

1. Execute Init(κ) for each user U.
2. $\mathcal{A}$ interacts with PMS users through queries until it outputs
   - $(a, d_0), (a, d_1)$ two index-attribute pairs
   - G_t group of users
   - U_P profile owner who is not in G_t
3. Bit $b \in_R \{0,1\}$. Execute Publish(pmk, P $(a, d_b), G_t$).
4. $\mathcal{A}$ interacting with PMS users through queries until it outputs some bit b*.

$\mathcal{A}$ is **successful** if:

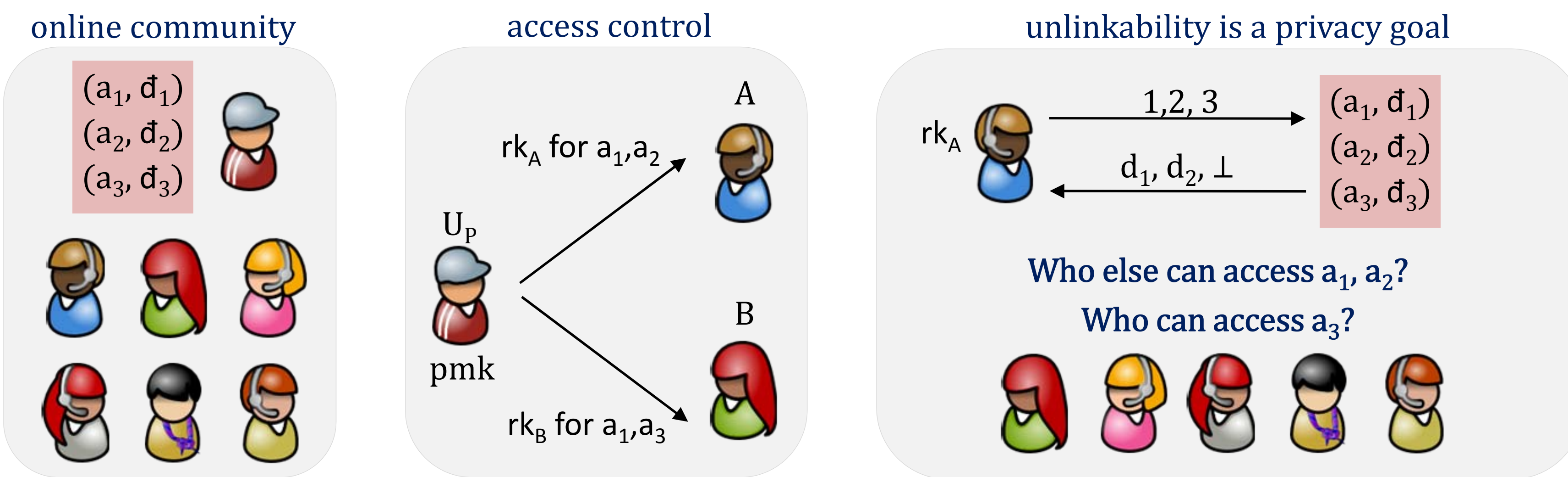
- $\mathcal{A}$ did not corrupt U_P or any user who was ever authorized to access a
- $\mathcal{A}$ did not retrieve d_b trivially via some suitable Retrieve query
- $b = b^*$

PMS is **confidential** if for all $\mathcal{A}$: |Pr[successful attack] – 1/2| is negligible in κ.

Privacy Goal: Unlinkability

Owner U_P knows which users were granted access to which pairs (a, đ) in P.

Unlinkability Profiles should hide which users can access which attributes.



Indistinguishability approach:

$\mathcal{A}$ with access rights to (a, đ) should not be able to distinguish whether user A or user B was granted access to a.

subsumes unlinkability across different profiles

Formal Definition of PMS Unlinkability

Unlinkability Game (high level)

1. Execute Init(κ) for all users U.
2. $\mathcal{A}$ interacts with PMS users through queries until it outputs
   two users U_0, U_1 index-attribute pair (a, d) profile owner U_P
3. Bit $b \in_R \{0,1\}$.
   If $(a, \bullet) \notin P$: execute Publish(pmk, P (a, d), U_b)
   If $(a, \bullet) \in P$: execute ModifyAccess(pmk, P, a, U_b)
3. $\mathcal{A}$ interacting with PMS users through queries until it outputs some bit b*.

$\mathcal{A}$ is **successful** if:

- U_P , U_0, or U_1 are uncorrupted
- $\mathcal{A}$ did not query Retrieve(P, a, U_0) or Retrieve(P, a, U_1)
- $b = b^*$

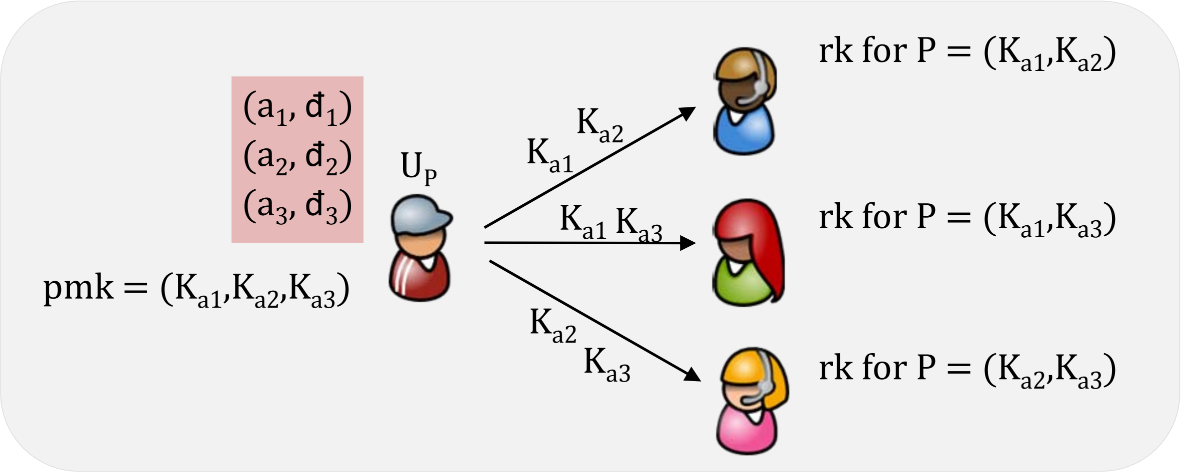
PMS is **unlinkable** if for all $\mathcal{A}$: |Pr[successful attack] – 1/2| is negligible in κ.

Shared Key (SK) Approach

- U_P uses separate key $K_a \longleftarrow SE.KGen(\kappa)$ for each pair (a, đ): đ $= SE.Enc(K_a, d)$
- Revocation: re-encryption with new K_a

(SE.KGen, SE.Enc, SE.Dec)
CCA-secure sym. enc. scheme

KGen(κ): outputs K
Enc(K, M): outputs C
Dec(K, C): outputs M or ⊥



- Each user manages own profile independently
- confidentiality and perfect* unlinkability (* if one omits key distribution)
- Each user U must store **one key per attribute index a per profile P**

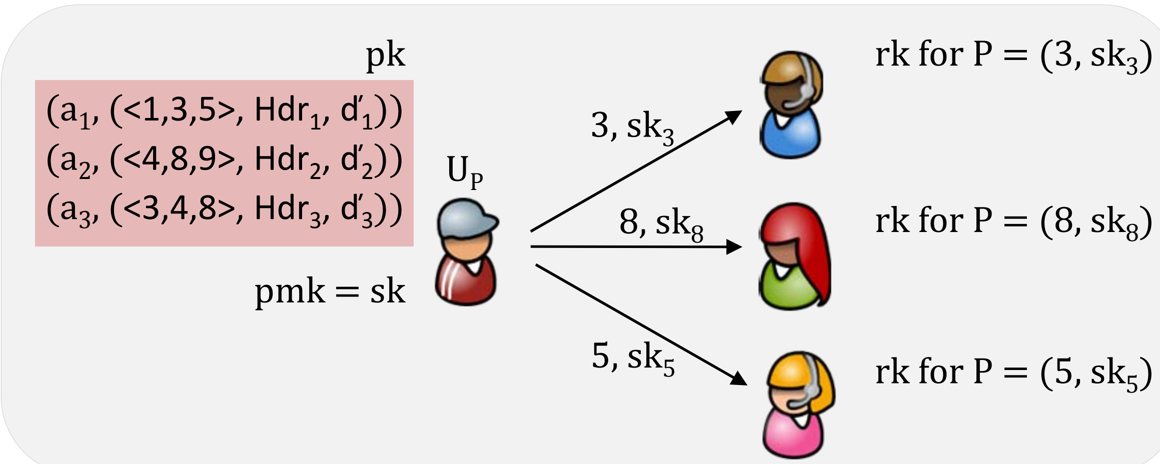
Broadcast Encryption (BE) Approach

- Each U_P manages own broadcast group. U_P has (pk, sk) ⟵ BE.Setup(κ, n).
- rk_U = (i, sk_i) with sk_i ⟵ BE.KGen(i, sk) , random pseudonym $i \in [1,n]$ for U.
- For each (a, d) : (Hdr, K_a) ⟵ BE.Enc(S, pk), authorized pseudonyms S
  d' = SE.Enc(K_a, d) and finally đ = (Hdr, S, d')
- Revocation : re-encryption with new (Hdr, K_a) for the modified set S

(BE.Setup, BE.KGen, BE.Enc, BE.Dec)
adaptive CCA-secure br. enc. scheme

Setup(κ, n): outputs (sk,pk)
KGen(i, sk): outputs (i, sk_i)
Enc(S, PK): outputs (Hdr, K)
Dec(S, i, sk_i, Hdr): outputs K or ⊥

Key K can be used with sym. enc. SE
[Gentry-Waters 2009]



- confidentiality and perfect* anonymity (* if one omits key distribution)
- provides anonymity but unlinkability ⇒ anonymity (see full version)
- Each user U must store **one key per profile** P

Overhead for Key Storage

Each user U has own profile P.

Each user U has on average n contacts (other users‘ profiles that U can access).

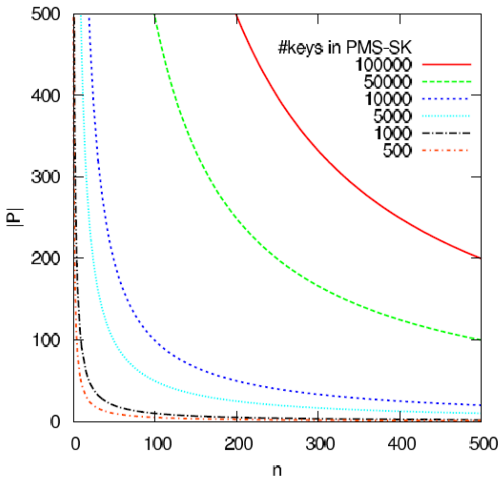
Each user U shares on average |P| attributes with each of his contacts.

trade-off
scalability vs. privacy

SK approach

(n+1)·|P| keys per user

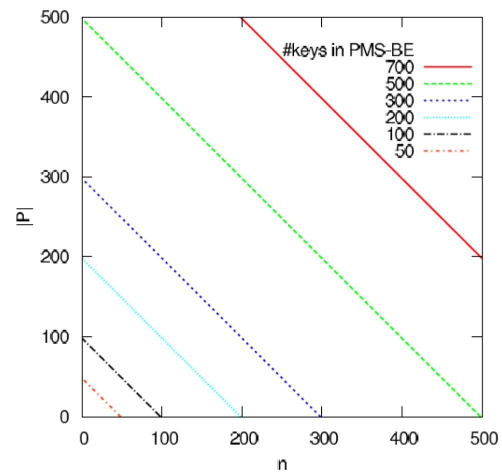
O(n·|P|)



BE approach

(n + 1+ |P|) keys per user

O(n + |P|)



Optimizing SK

Group Key Management for K_a

- LKH [WGL98, WHA99]
- OFT [Canetti et al. 1999]

Impact on Real-Life Communities

Analysis for Facebook, Twitter, XING, Flickr (based on their own statistics)

community	# contacts	# attributes	# keys		storage (KB)*	
			SK	BE	SK	BE
facebook	150	180	~27000	332	650	8
twitter	50	180	~9000	232	220	6
XING	168	~36	~8350	220	200	5
flickr	12	200	2000	214	62	5

* 192bit keys (SE and BE)

SE and BE costs differ by a factor of 10 to 80

SE and BE overhead remains below 1 MB which could be acceptable

Summary of Results

Cryptographic Model for Private User Profiles

- first cryptographic model to capture main functionality of user profiles
- security goal confidentiality of profile data (single attributes)
- privacy goal unlinkability to hide access rights across different attributes
- (full version) weaker privacy goal: anonymity to hide ids of users with access rights

Two (General) Solutions

- SK approach CCA-secure SE scheme (one retrieval key per attribute)
  $O(n \cdot |P|)$ keys per user / confidentiality + (perfect) unlinkability
- BE approach adaptive CCA-secure BE scheme (one retrieval key per profile)
  $O(n + |P|)$ keys per user / confidentiality + (perfect) anonymity

Practical Analysis for Real-Life Communities

- using statistics of Facebook, Twitter, XING, Flickr
- both approaches seem practical in the average case (in terms of key storage)