Mutual Contact Discovery

Jaap-Henk Hoepman
Privacy & Identity Lab
iHub
Radboud University
Karlstad University
University of Groningen
jhh@cs.ru.nl // www.cs.ru.nl/~jhh // blog.xot.nl // @xotoxot

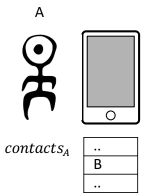
1

It started with a monkey…

2

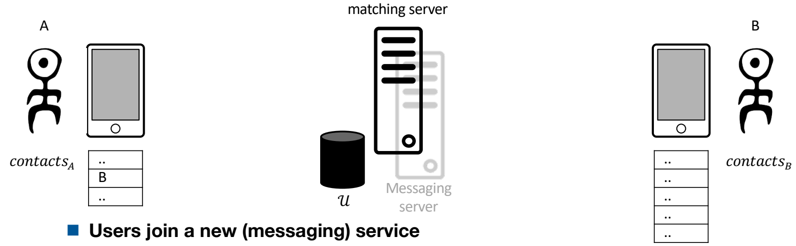
Contact discovery: preliminaries

- **There is an underlying, existing, social graph (V, E)**
  - Unique identifiers $A, B, \ldots$ (e.g. phone numbers).
  - Low entropy (enumerable; guessable).
  - User/device A maintains contact list $contacts_A = \{B \in V \mid (A, B) \in E\}$



3

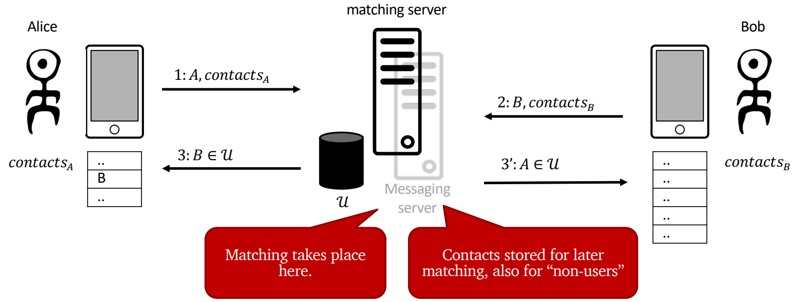
Contact discovery: goal



- **Users join a new (messaging) service**
  - And what to learn who is also a member of the service
- **Requirement**
  - If B $\in$ $contacts_A$ then A gets notified that $B \in \mathcal{U}$ when B joins (or when A adds B to $contacts_A$ when $B \in \mathcal{U}$ already).

4

The classic (WhatsApp) approach



Privacy issue

- **Matching service learns all contacts of a user on the underlying social graph, including the identifiers of 'non users'**

C 2/50 | EN | Official Journal of the European Union | 3.1.2022

Action brought on 1 November 2021 — WhatsApp Ireland v EDPB

(Case T-709/21)

(2022/C 2/69)

Language of the case: English

Parties

Applicant: WhatsApp Ireland Ltd (Dublin, Ireland) (represented by: H.-G. Kamann, F. Louis, A. Vallery, lawyers, P. Nolan, B. Johnston, C. Monaghan, Solicitors, P. Sreenan, D. McGrath, C. Geoghegan and E. Egan McGrath, Barristers-at-Law)

Defendant: European Data Protection Board

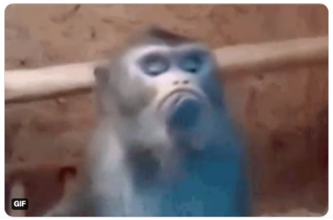
Solutions

- **Hashing**
  - Store the contacts as hashes on the matching database.
  - Improvement: hash the contactlist before sending it to the matching service.
  - Even better: use a key derivation function (KDF).
- **Problem**
  - NL: 2^{24} phonenumbers.
  - Dictionary easily computed in seconds, and storable on disk.
- **Trusted hardware (Signal)**
  - Use hashing like above, but
  - Run matching code on server in trusted hardware.
- **Private Set Intersection**
  - To compute $\mathcal{U} \cap contacts_A$.
  - Problem
    - *Expensive.*
    - *Needs to be done for every user whenever a new user joins.*

Still some privacy issues remain

Dorothea Baur (Dr.)
@DorotheaBaur

I have finally installed #SignalApp but I am really not happy with the fact that apparently everyone who is on it already and who has my number, gets notified about that. Even contacts I have long deleted from my phone. Is this #privacy-friendly? Does anyone share my anger?



- **Not mutual**
  - If $A \in contacts_B$ then B gets notified that $A \in \mathcal{U}$ even if $B \notin contacts_A$.
- **This is problematic**
  - Ex-es, former bosses, doctors.

Mutual contact discovery: requirements

- **Correctness**
  - Output out_A
  - $B \in out_A$ if $(B \in \mathcal{U})$
    $\wedge (A \in contacts_B)$
    $\wedge (B \in contacts_A)$.
- **Security**
  - Let $B \in contacts_A$. If $B \notin \mathcal{U}$ then X cannot force $B \in out_A$.

  (X stands for the matching server or a user respectively.)
- **Membership privacy**
  - If $X \notin contacts_A$ then X does not learn whether $A \in \mathcal{U}$ (for any $A \neq X$ of its choosing).
- **Contact privacy**
  - X does not learn whether $B \in contacts_A$ (for any A, B both unequal to X of its choosing).

9

Threat model

- **Active adversary**
  - May behave arbitrarily.
  - May block and observe messages (as channels are secure, eavsdropping, replaying or modifying messages is prevented).
- **May use prior knowledge to maximise chance of success**
  - Knows list of identifiers in use,
  - Knows identifiers for persons of interest, and
  - May have knowledge of potential contacts.

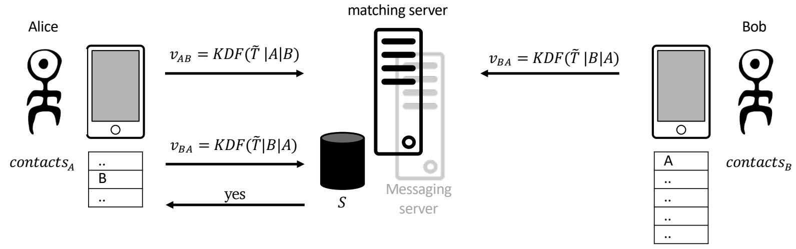
10

Protocol #1

- **Notation**
  - Time divided into slots T, starting at 0.
  - Epoch $\tilde{T} = T \ div\ 2$.
  - Token $v_{AB} = KDF(\tilde{T}\ |A|B)$.
- **Submission phase, even epoch**
  - Each user B sends v_{AB} to the server, for all $A \in contacts_B$.
  - The server stores these values in S.
- **Query phase, odd epoch**
  - Each B now sends v_{BA} to the server, for all $A \in contacts_B$.
  - The server returns whether $v_{BA} \in S$.
  - B adds A to out_B.
- **After every odd epoch**
  - The server deletes S.

11

Protocol #1 example



12

Key derivation function

- **Like a (cryptographic) hash function**
  - Takes a 'significant' time to compute (to make constructing a dictionary expensive).
- **Hashing single phone numbers offers no significant protection**
  - Small dictionary can be computed in seconds
- **But, what about the concatenation of two phone numbers**
  - NL: 2^{24} phonenumbers.
  - Input to KDF then 48 bits; dictionary 2^{48}~32 TB.
  - Assume averge customer hardware 2^{20} times slower than attacker computing power.
  - If average contact list contains 2^8 (hundreds) of contacts, then work for attacker compared to work for user is factor $2^{48}/(2^{20}2^8) = 2^{20}$ higher.

Analysis

- **Correctness**
  - Yes
- **Security**
  - Server: can always reply yes in query phase.
  - User: can guess A, B such that $B \in contacts_A$, then compute v_{AB} and send it in the submission phase. A will query this value in the query phase and (wrongly) conclude $B \in out_A$
- **Membership privacy**
  - Server, user: can choose A and subsequently guess B such that $B \in contacts_A$, and then submit v_{AB} in the query phase to verify this guess.
- **Contact privacy**
  - Server, user: Compute v_{AB} to test $B \in contacts_A$.
- **Limited protection?**
  - Infeasible in general because KDF limits number of guesses

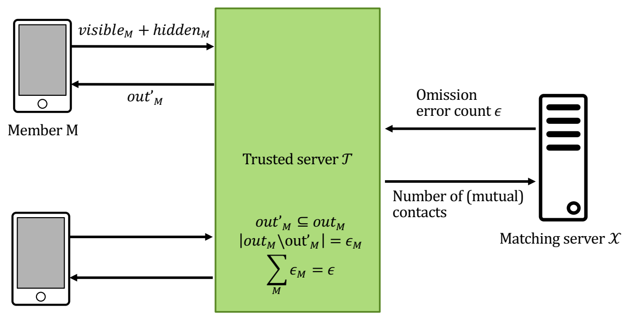
Analysis (cont.)

- **Better than single sided contact discovery**
  - Adversary X needs to guess a contact of A in order to detect $A \in \mathcal{U}$.
  - Infeasible unless adversary has sufficient prior knowledge
  - In particular, it is infeasible for the server to reconstruct the social graph.
- **To improve even further**
  - Either autenticate submissions or queries (so that A cannot submit a value that isn't related to itself).
  - Or make sure that submissions or queries depend on a secret that only users themselves know
- **Idea: using certified identifiers**
  - Messaging server, or from underlying social graph (eg SIM)

A more formal treatment

- **We have social graph (V, E)**
  - Stored by clients: $contacts_A = \{B \in V \mid (A, B) \in E\}$
  - Split in $visible_A + hidden_A$
  - $hidden_A = \emptyset$ for honest A
- **Define $A \rightrightarrows B$ if**
  - $(A \in visible_B) \wedge (B \in contacts_A)$
  - Is only symmetric for honest users
- **Correctness**
  - $B \in out_A$ if $B \in \mathcal{U} \wedge A \rightrightarrows B$
- **Static setting**
  - Users and contacts are fixed
- **Sender anonymous star network**
  - Tor, or Apple Private Relay

The ideal model



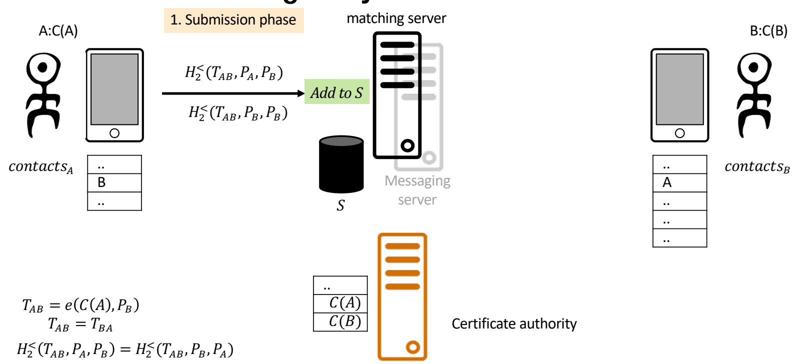
17

Protocol #2: using a key server

- **Definitions**
  - Let $\boldsymbol{e}: \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$ be a pairing function:
    - *$e(aP, bQ) = e(P, Q)^{ab}$ for any points P, Q*
  - Let $H_1 : V \mapsto \mathbb{G}_1$ be cryptographic hash function
    - *Define $P_A = H_1(P)$*
  - Let s be a secret of the certificate authority
    - *Define the certificate for A as $C(A) = sP_A$*
  - Define a *token* $T_{AB} = e(C(A), P_B)$
    - *Then $T_{AB} = T_{BA}$*
    - *Only A and B can create it*
  - Define another cryptographic hashfunction $H_2 : \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \{0, 1\}^n$
  - Let < be a total order over $\mathbb{G}_1$
  - Define $H_2^{<}(X, Y, Z) = \begin{cases} H_2(X, Y, Z) \text{ if } Y < Z \\ H_2(X, Z, Y) \text{ otherwise} \end{cases}$
  - Then $H_2^{<}(X, Y, Z) = H_2^{<}(X, Z, Y)$

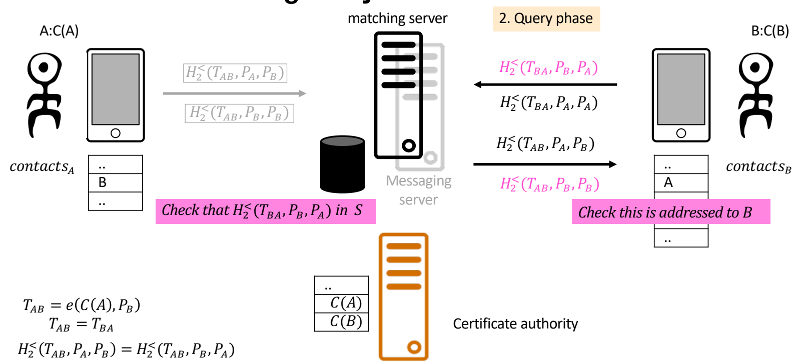
18

Protocol #2: using a key server



19

Protocol #2: using a key server



20

Analysis

- **Sketch**
  - Only X knows $C(X)$
  - So, only A and B can contstruct T_{AB}, by Bilinear Diffie-Hellman (BDH) problem
  - A can pretend that B in $contacts_A$, but this is modelled as if $B \in contacts_A$
  - T_{AB} only sent as $H_2^{<}(T_{AB}, X, Y)$, so no other party learns it
  - So, meaningful $H_2^{<}(T_{AB}, X, Y)$ can only be constructed by A or B
  - B adds A to out_B only if it receives $H_2^{<}(T_{AB}, P_A, P_B), H_2^{<}(T_{AB}, P_B, P_B)$
    - *which is/can only be constructed by A*
  - The server can prevent A from adding B to out_A by not sending $H_2^{<}(T_{AB}, P_A, P_B), H_2^{<}(T_{AB}, P_A, P_A)$ but this is modelled as $A \in hidden_B$

21

Generalise to dynamic setting

- **Make asynchronous**
  - Note how in query phase members submit the same information as in the submission phase
  - Therefore, omit submission phase
  - Members only execute the query phase, regularly
  - Query tuples added to database
  - Responses sent when a match is detected
- **Support deletion**
  - Add delete command, sending same tuple as in query command
  - Server is supposed to honestly delete the tuple from its database

22

Discussion

[Monty Python's Argument Clinic sketch]

23