Privacy-Preserving & User-Auditable Pseudonym Systems

Jan Camenisch, Anja Lehmann
IBM Research – Zurich
Motivation: How to maintain related yet distributed data?

- examples: social security system, ehealth
  - different entities maintain data of citizens
  - eventually data needs to be exchanged or correlated

- simple solution: data gets associated with globally unique identifiers (e.g., US, Belgium, Sweden, ...)
- unique identifiers are security & privacy risk
  - no control about data exchange & usage
  - if associated data is lost, all pieces can be linked together
  - user is fully traceable
Local Pseudonyms & *Trusted* Converter

- user data is associated with random looking local identifiers – the *pseudonyms*
- only central entity – the *converter* – can link & convert pseudonyms

+ control about data exchange
+ if records are lost, pieces cannot be linked together
Local Pseudonyms & Trusted Converter

- user data is associated with random looking local identifiers – the pseudonyms
- only central entity – the converter – can link & convert pseudonyms

+ control about data exchange
+ if records are lost, pieces cannot be linked together
+ converter can provide audit logs to users (GDPR-requirement)
  - converter learns all request & knows all correlations
Privacy-Preserving Pseudonym System (CCS’15)

- user data is associated with random looking local identifiers – the *pseudonyms*
- only central entity – the *converter* – can link & convert pseudonyms

+ control about data exchange
+ if records are lost, pieces cannot be linked together
+ converter can provide audit logs to users (*GDPR*-requirement)
  - converter learns all requests & knows all correlations
This work: Privacy-Preserving & User-Auditable Pseudonym System

- user data is associated with random looking local identifiers – the pseudonyms
- only central entity – the converter – can link & convert pseudonyms

+ control about data exchange
+ if records are lost, pieces cannot be linked together
+ converter can provide audit logs to users (GDPR-requirement)
  - converter learns all requests & knows all correlations
(Un)linkable Pseudonyms | Pseudonym Generation

- user, converter & server jointly derive pseudonyms from unique identifiers

- previous work [CL15]: generation required converter to know user IDs
- this work: *oblivious* pseudonym generation triggered by user
(Un)linkable Pseudonyms | Pseudonym Conversion

- only converter can link & convert pseudonyms, but does so in a blind way
(Un)linkable Pseudonyms | Pseudonym Conversion & Audit Logs

- only converter can link & convert pseudonyms, but does so in a blind way
- every conversion triggers blind generation of audit log entry

- audit log entries are only accessible by the affected user
(Un)linkable Pseudonyms | Consistency

- pseudonym conversions & generations are fully consistent
- conversions are transitive, un-linkable data can be aggregated
Our Protocol

- high-level idea of convertible pseudonyms
- adding (efficient) auditability
- security against active adversaries
**High-level Idea | Pseudonym Generation**

**Core Idea**
Generation: $\mathcal{X}$ blindly computes $\text{nym}_{i,A} \leftarrow \text{PRF}(k, \text{uid}_i)^x_A$

- **1.** $\mathcal{X}$ and $U_i$ jointly compute $z_i \leftarrow \text{OPRF}(k, \text{uid}_i)$
- **2.** $U_i$ encrypts $z_i$ for $S_A$
  $C_{\text{nym}} \leftarrow \text{Enc}(pk_A, z_i)$
- **3.** $\mathcal{X}$ blindly computes $\text{nym}_{i,A}$ and $U_i$ jointly compute $z_i \leftarrow \text{OPRF}(k, \text{uid}_i)$
  $C'_{\text{nym}} \leftarrow C_{\text{nym}}^{x_A}$
- **4.** $S_A$ decrypts pseudonym $\text{nym}_{i,A} \leftarrow \text{Dec}(sk_A, C'_{\text{nym}})$

[k, for each server: $x_A, x_B, x_C, ...$]
High-level Idea | Pseudonym Conversion

Core Idea
Generation: $X$ blindly computes $nym_{i,A} \leftarrow \text{PRF}(k, uid_i)^{x_A}$
Conversion: $X$ blindly computes $nym_{i,B} \leftarrow nym_{i,A}^{x_B} / x_A$

[1] $S_A$ encrypts $nym_{i,A}$ under $S_B$'s key
$C \leftarrow \text{Enc}(pk_B, nym_{i,A})$

[2] $X$ blindly transforms encrypted pseudonym
$C' \leftarrow C^\Delta$ with $\Delta = x_B / x_A$
$C' = \text{Enc}(pk_B, nym_{i,A})^{x_B} / x_A$
$= \text{Enc}(pk_B, \text{PRF}(k, uid_i)^{x_A})^{x_B} / x_A$
$= \text{Enc}(pk_B, \text{PRF}(k, uid_i)^{x_B})$
$= \text{Enc}(pk_B, nym_{i,B})$

[3] $S_B$ decrypts converted pseudonym
$nym_{i,B} \leftarrow \text{Dec}(sk_B, C')$

Converter $X$

k, for each server: $x_A, x_B, x_C, ...$
High-level Idea

Generation
Conversion
High-level Idea | Adding Auditability

decrypt all audit ciphertexts: info ← Dec(usk,C*) ?

NymRequest, upk’

Converter X

NymResponse, upk’

Server A

nym_{i,A}, upk’

ConvRequest, upk’’

Converter X

ConvResponse, upk’’’

Server A

nym_{i,A}, upk’’’

Server B

nym_{i,B}, upk’’’

Audit Bulletin Board

C* ← Enc(upk’, info)

C* ← Enc(upk’, info)
High-level Idea | **Adding Efficient Auditability** (via Audit Tags)

decrypt ciphertext for $T_A$:  
\[ info \leftarrow \text{Dec}(usk, C^*) \]

**Converter $\mathcal{X}$**

- **NymRequest**, $upk'$, $C_T$
- **Server A**
  - **NymResponse**, $upk'$, $C_T$
  - $nym_{i,A}$, $upk'$, $T_A$
  - $T_A \leftarrow \text{Dec}(sk_A, C_T)$

**Generation**

**Conversion**

**Audit Bulletin Board**

- $T_A$, $C^*$
- ...
High-level Idea | Adding **Efficient** Auditability (via Audit Tags)

decrypt ciphertext for $T_A$:
\[ info \leftarrow \text{Dec}(usk, C^*) \]

get new audit tags for $T_A$:
\[ T'_A \leftarrow \text{Dec}(usk, C^*_{TA}) \]

**Generation**

**Conversion**

**Audit Bulletin Board**

\[ T_A, C^* \]

**Tag Chain:**

\[ T_A, C^*_{TA} \]

\[ C^* \leftarrow \text{Enc}(upk'', info) \]

**Converter X**

\[ \text{ConvRequest, upk'', } T_A, C^*_{TA} \]

**Server A**

\[ \text{NymResponse, upk', } C_T \]

\[ \text{nym}_{i,A}, \text{ upk', } T_A \]

\[ T_A \leftarrow \text{Dec}(sk_A, C_T) \]

\[ \text{ConvResponse, upk''' } \]

**Server B**

\[ \text{nym}_{i,B}, \text{ upk''' } \]

\[ C^*_{TA} \leftarrow \text{Enc}(upk'', T'_A) \ldots \text{for random } T'_A \]
High-level Idea | Adding Efficient Auditability (via Audit Tags)

decrypt ciphertext for $T_A$: $info \leftarrow \text{Dec}(usk, C^*)$

get new audit tags for $T_A$: $T'_A \leftarrow \text{Dec}(usk, C^*_{TA})$, $T_B \leftarrow \text{Dec}(usk, C^*_{TB})$

$C_T \leftarrow \text{Enc}(pk_A, T_A)$ ... for random $T_A$

$NymRequest, upk', C_T$

$Converter \chi$

$usk, upk, \{T_A, T'_A, T_B\}$

$NymResponse, upk', C_T$

$Server A$

$nym_{i,A}, upk', T_A$

$T_A \leftarrow \text{Dec}(sk_A, C_T)$

$Tag Chain: C^* \leftarrow \text{Enc}(upk'', info)$

$ConvResponse, upk''$

$Converter \chi$

$ConvRequest, upk'', T_A, C^*_{TA}$

$Server A$

$nym_{i,A}, upk', T_A$

$T'_A \leftarrow \text{Dec}(usk, C^*)$

$C^*_{TA} \leftarrow \text{Enc}(upk'', T'_A)$ ... for random $T'_A$

$C^*_{TB} \leftarrow \text{Enc}(upk'', T_B)$ ... for random $T_B$

$Server B$

$nym_{i,B}, upk'', T_B$
High-level Idea | Security against Active Adversaries

- Decrypt ciphertext for T_A: info ← Dec(usk, C*)
- Get new audit tags for T_A: T'_A ← Dec(usk, C*_{TA})
- T_B ← Dec(usk, C*_{TB})

**Converter X**

- NymRequest, upk', C_T
- NymResponse, upk', C_T

**Server A**

- nym_{i,A}, upk, T_A
- T_A ← Dec(sk_A, C_T)

**Generation**

- ConvRequest, upk'', T_A, C*_{TA}, \pi_A

**Server A**

- nym_{i,A}, upk', T_A
- C*_{TA} ← Enc(upk'', T_A) ... for random T_A

**Conversion**

- ConvResponse, upk'''

**Server B**

- nym_{i,B}, upk''', T_B
- C*_{TB} ← Enc(upk''', T_B) ... for random T_B

**Audit Bulletin Board**

- T_A, C*
- Tag Chain:
  - T_A, C*_{TA}
  - T_A, C*_{TB}
provably secure construction in the Universal Composability (UC) framework based on
  – homomorphic encryption scheme (ElGamal encryption)
  – homomorphic encryption scheme with re-randomizable public keys (ElGamal-based)
  – oblivious pseudorandom function with committed outputs (based on Dodis-Yampolskiy-PRF)
  – signature scheme for homomorphic encoding functions (based on Groth signature scheme)
  – zero-knowledge proofs (Fiat-Shamir NIZKs)
  – commitment scheme (ElGamal based)

secure against actively corrupt users & servers, and honest-but-curious converter

concrete instantiation ~50ms computational time per party for conversion
(Un)linkable & Auditable Pseudonyms | Summary

- Pseudonym scheme for (un)linkable data storage with controlled & auditable data exchange
- Pseudonyms can only be linked via a central, but oblivious converter
- Oblivious converter blindly generates user-centric audit logs
- Conversions & audit logs are done in a blind way → converter must not be a trusted entity

→ Paradigm shift: Unlinkability per default, linkability only when necessary

Thanks! Questions?

anj@zurich.ibm.com
### (Un)linkable & Auditable Pseudonyms | Efficiency

<table>
<thead>
<tr>
<th>Pseudonym Generation</th>
<th>$U_i$</th>
<th>$X$</th>
<th>$S_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$22G + 6Z_n^*$</td>
<td>$22G + \tilde{G} + 6Z_n^*$</td>
<td>$4G$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pseudonym Conversion</th>
<th>$S_A$</th>
<th>$X$</th>
<th>$S_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$37G + 4P$</td>
<td>$47G + 2\tilde{G} + 4P$</td>
<td>$13G$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User Audit</th>
<th>$U_i$</th>
<th>with $c$ denoting the amount of conversions for $U_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$3c \cdot G$</td>
<td></td>
</tr>
</tbody>
</table>
(Un)linkable Pseudonyms | Corruption Model

- servers and users can be fully corrupt
- converter at most honest-but-curious
(Un)linkable Pseudonyms | Consistency

- pseudonym generation is deterministic & consistent with blind conversion
(Un)linkable Pseudonyms | Consistency

- pseudonym conversions are transitive, unlinkable data can be aggregated