

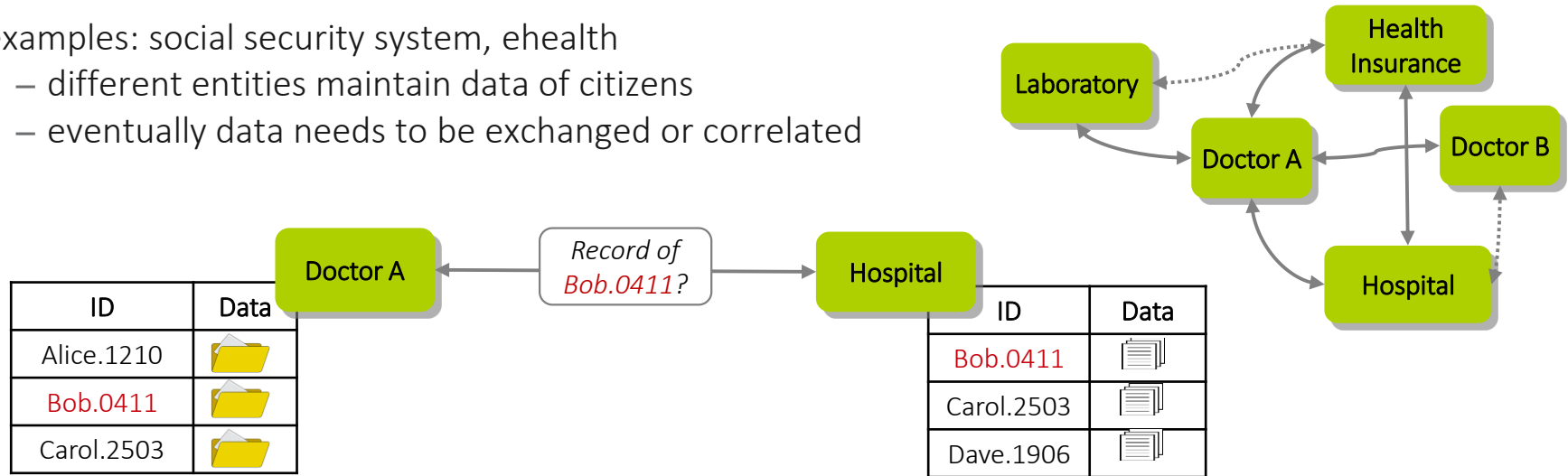
Privacy-Preserving & User-Auditable Pseudonym Systems

Jan Camenisch, Anja Lehmann

IBM Research – Zurich

Motivation: How to maintain related yet distributed data ?

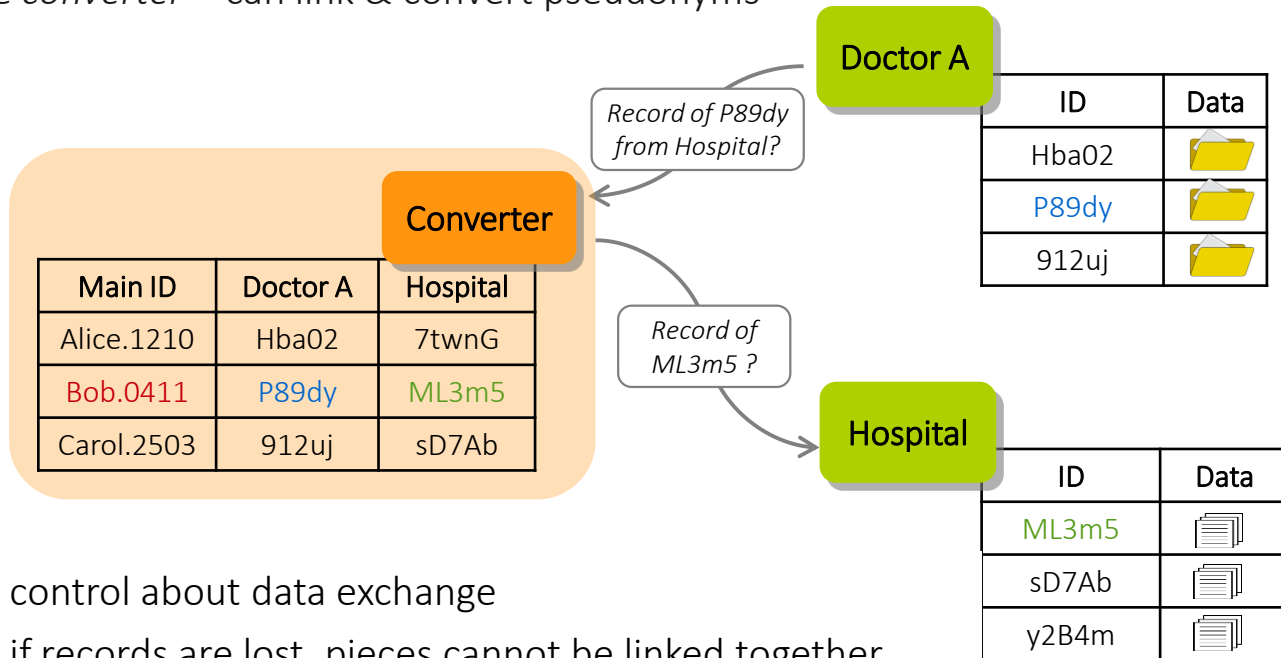
- examples: social security system, ehealth
 - different entities maintain data of citizens
 - eventually data needs to be exchanged or correlated



- 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

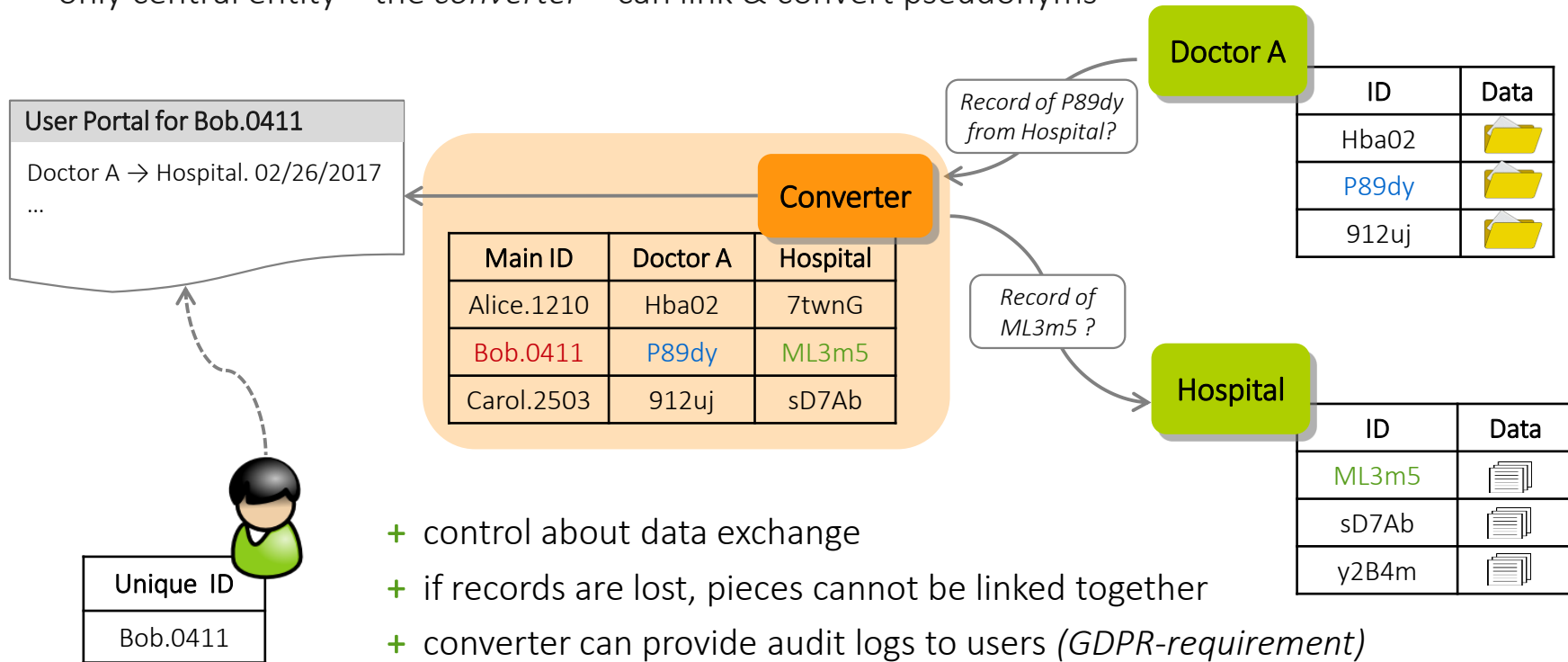
Local Pseudonyms & *Trusted Converter*

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



Local Pseudonyms & *Trusted Converter*

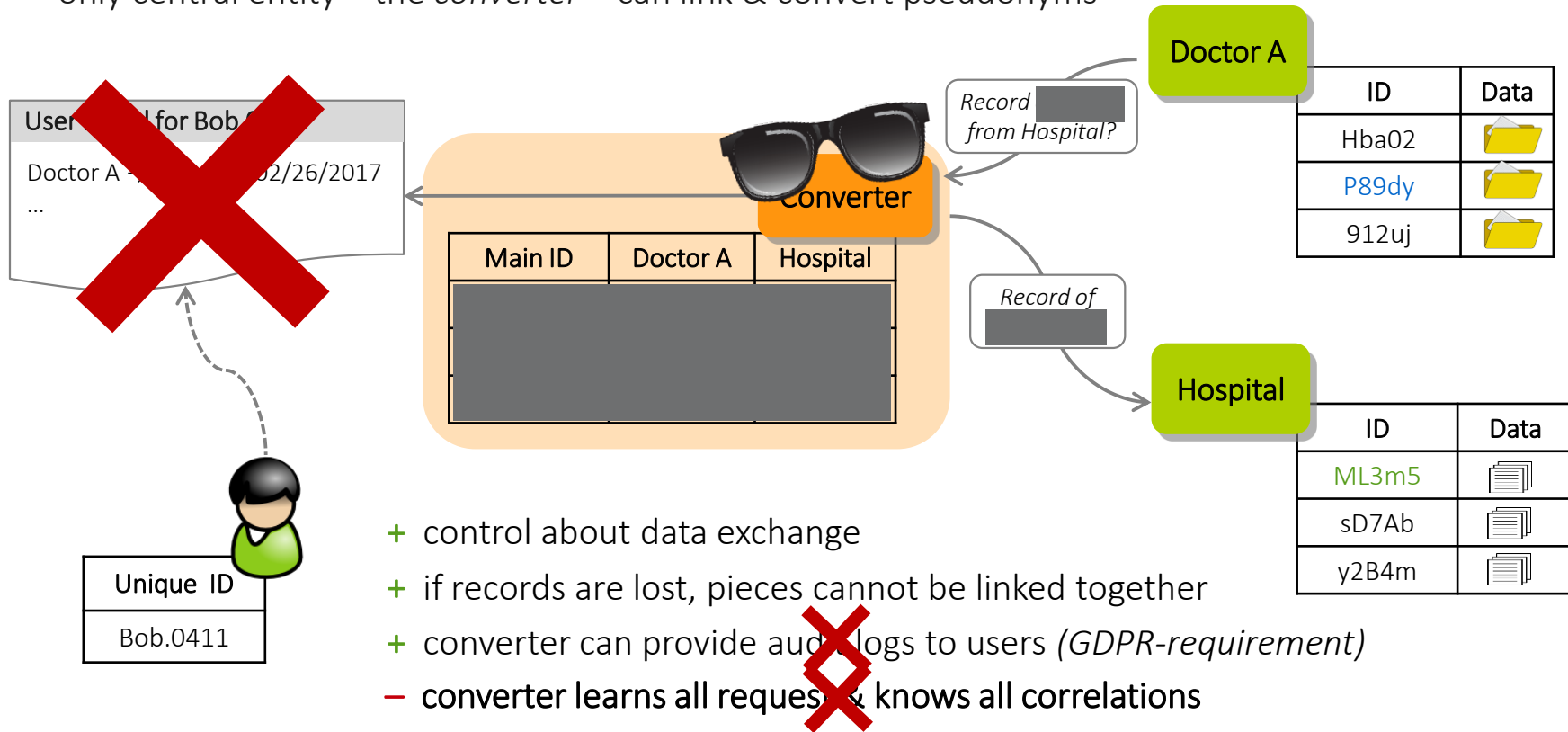
- user data is associated with random looking local identifiers – the *pseudonyms*
- only central entity – the *converter* – can link & convert pseudonyms



- + 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

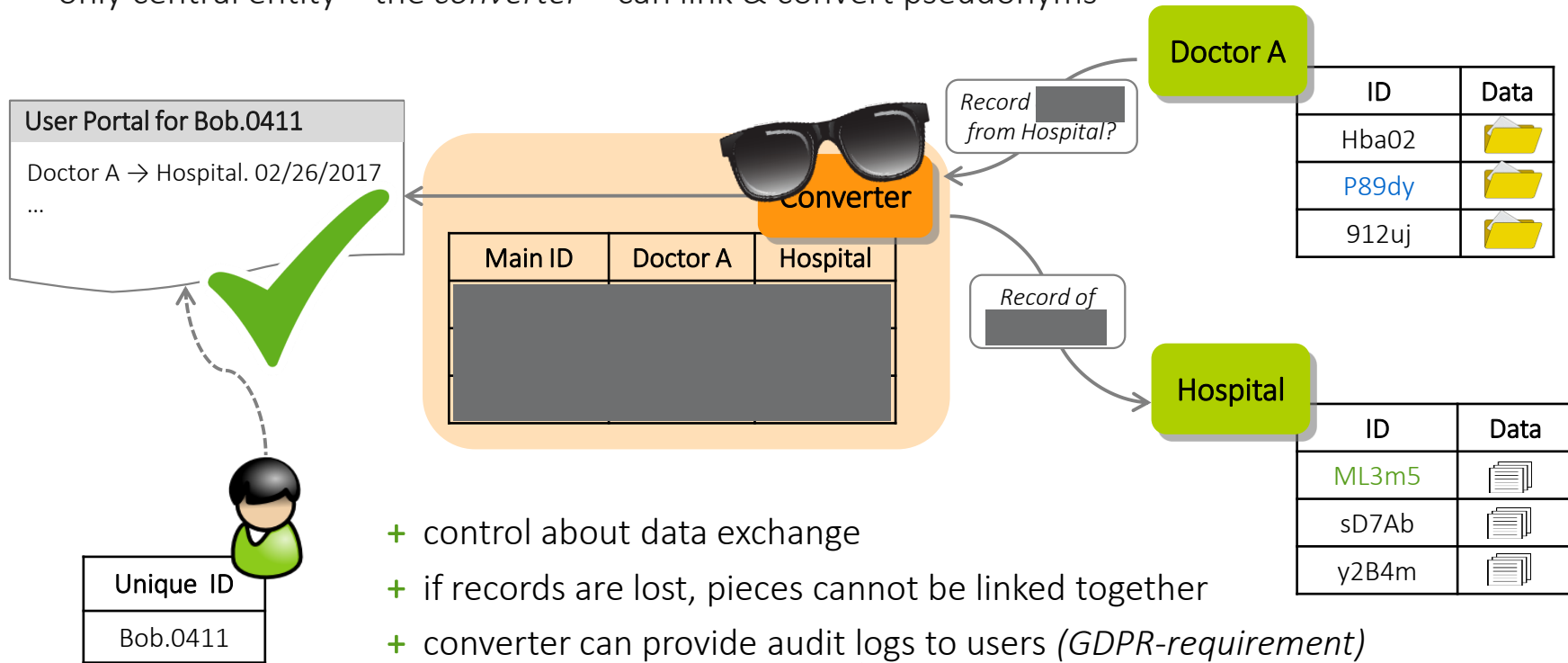
Privacy-Preserving Pseudonym System (CCS'15)

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



This work: Privacy-Preserving & User-Auditable Pseudonym System

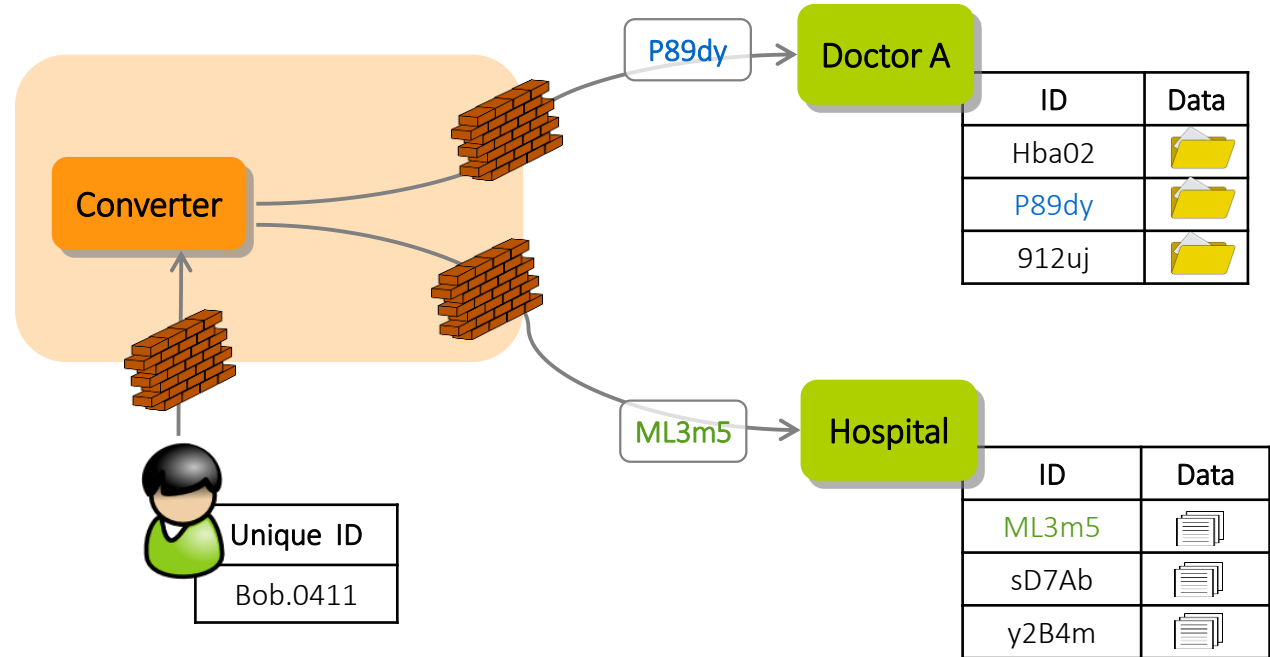
- user data is associated with random looking local identifiers – the *pseudonyms*
- only central entity – the *converter* – can link & convert pseudonyms



- + 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

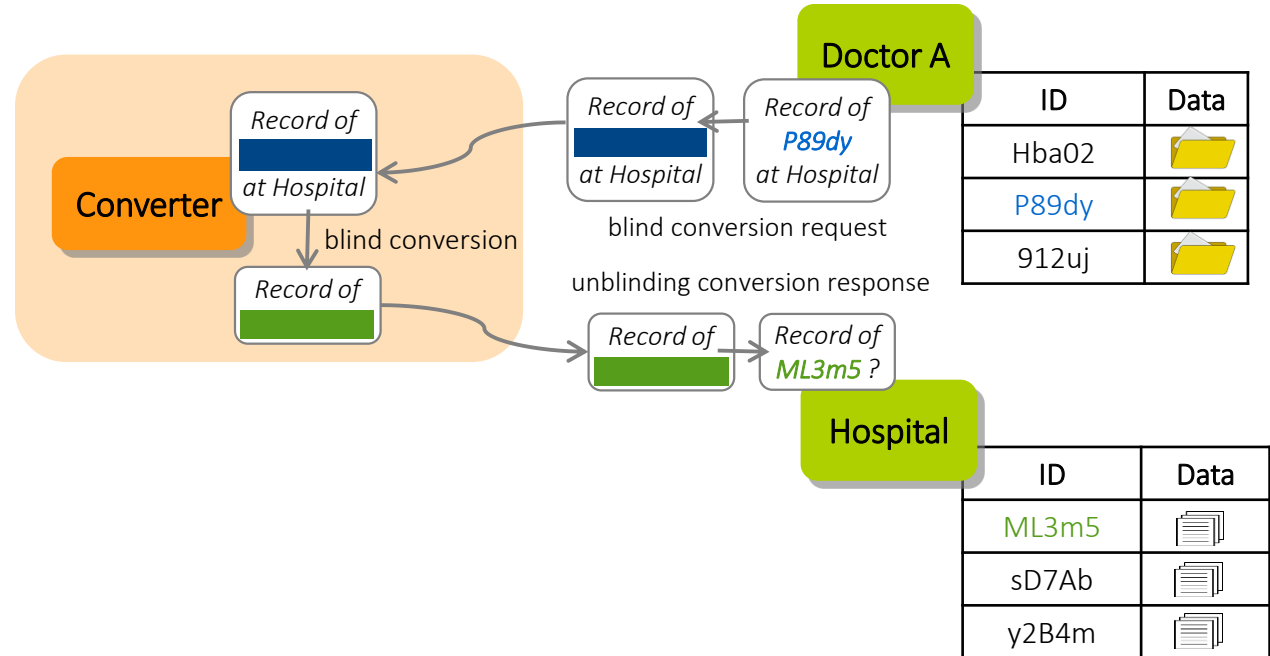
- user, converter & server jointly derive pseudonyms from unique identifiers



- previous work [CL15]: generation required converter to know user IDs
- this work: *oblivious* pseudonym generation triggered by user

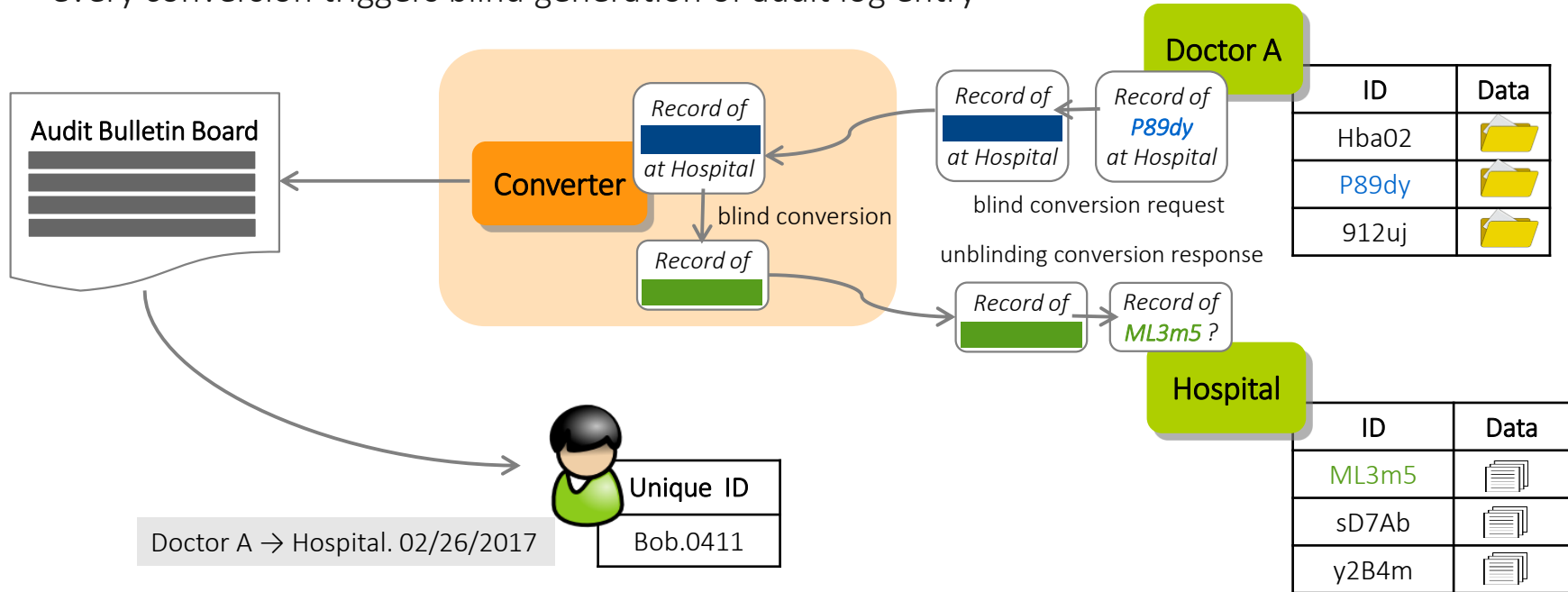
(Un)linkable Pseudonyms | Pseudonym Conversion

- only converter can link & convert pseudonyms, but does so in a blind way



(Un)linkable Pseudonyms | Pseudonym Conversion & Audit Logs

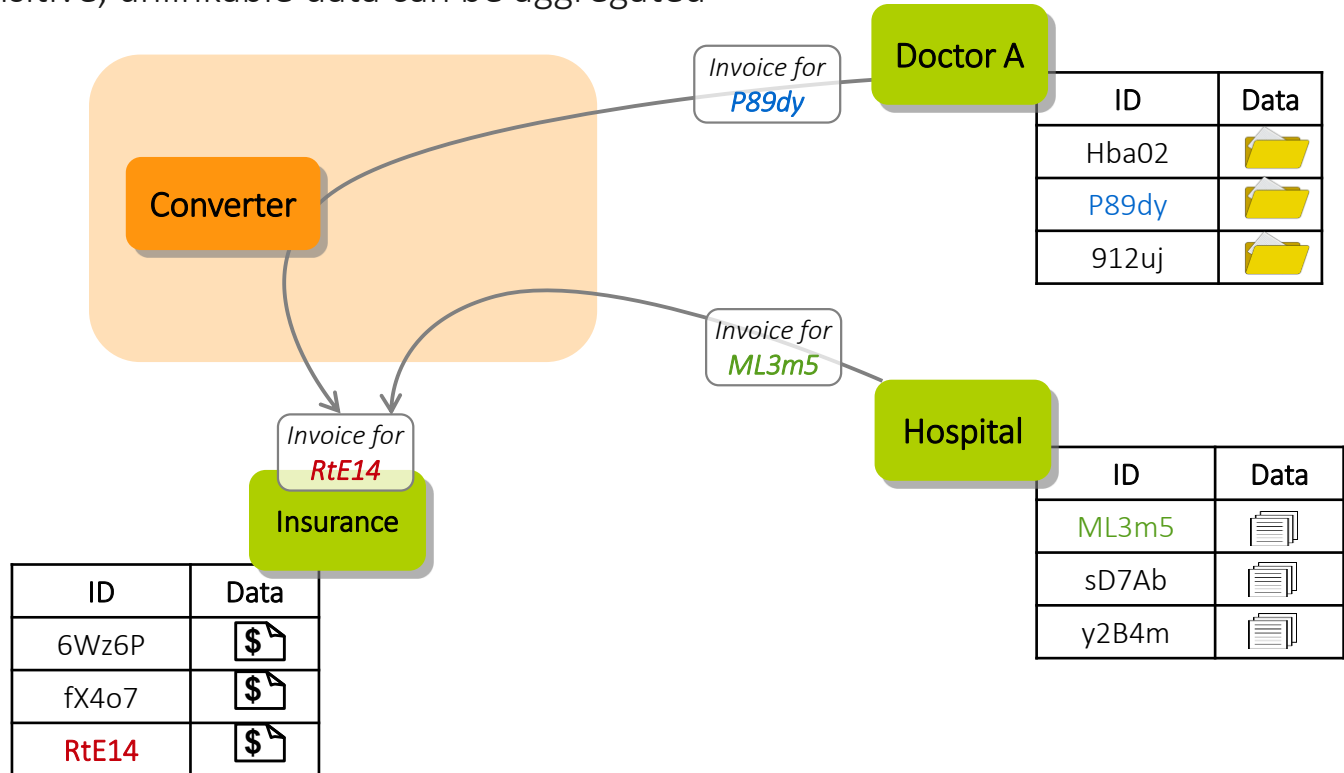
- only converter can link & convert pseudonyms, but does so in a blind way
- every conversion triggers blind generation of audit log entry



- audit log entries are only accessible by the affected user

(Un)linkable Pseudonyms | Consistency

- pseudonym conversions & generatios are fully consistent
- conversions are transitive, unlinkable data can be aggregated



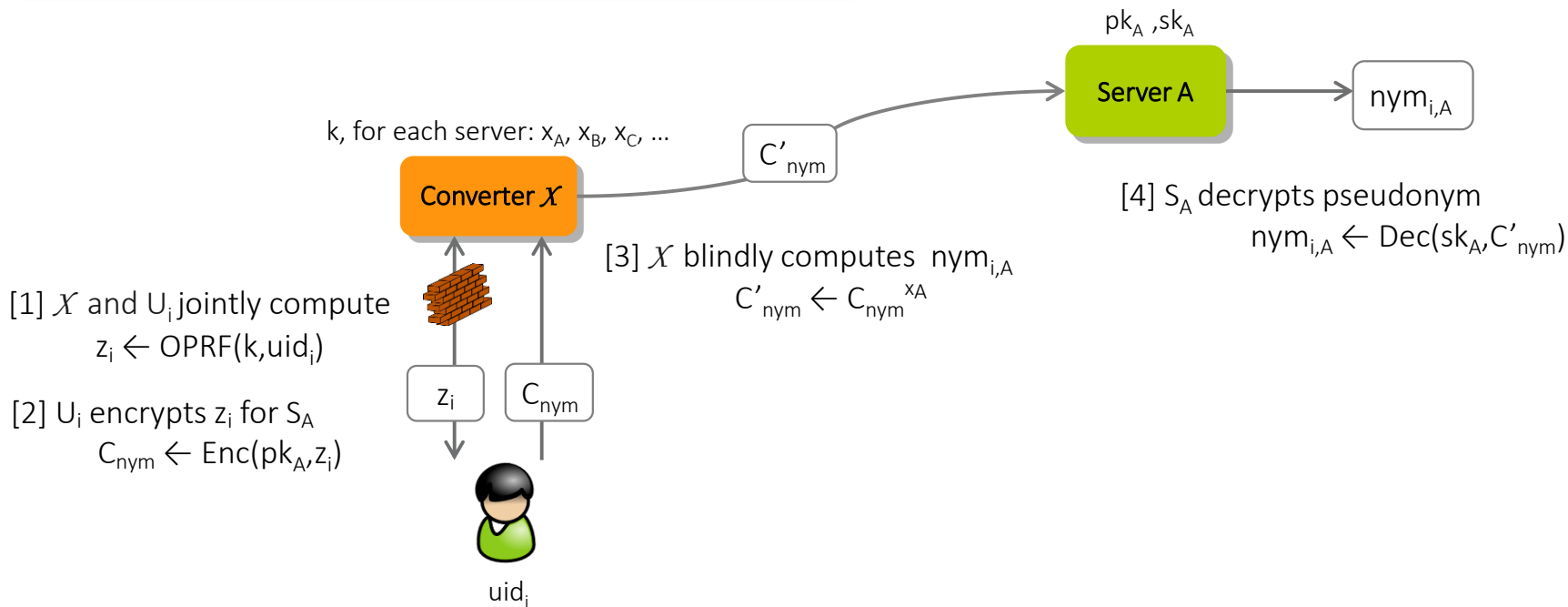
Our Protocol

- 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 $\text{nym}_{i,A} \leftarrow \text{PRF}(k, \text{uid}_i)^{x_A}$

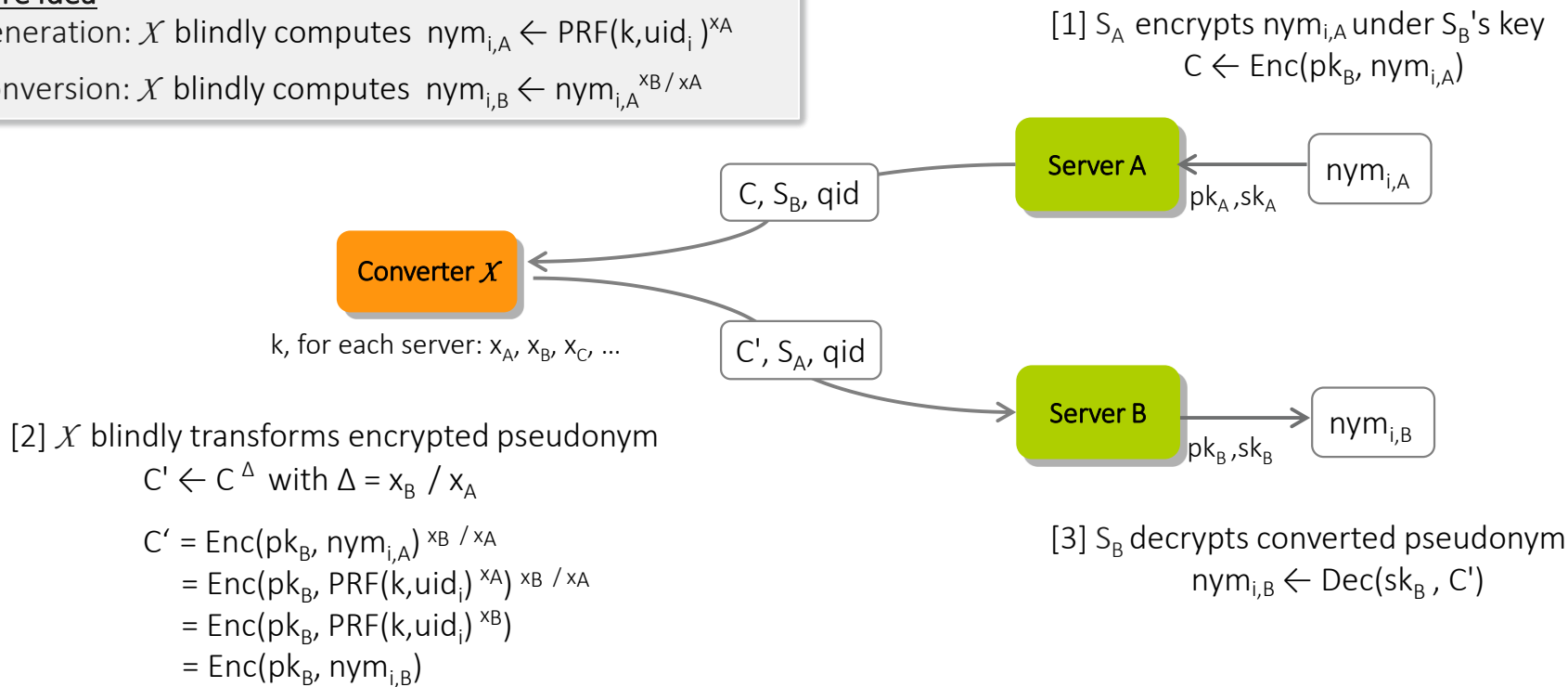


High-level Idea | Pseudonym Conversion

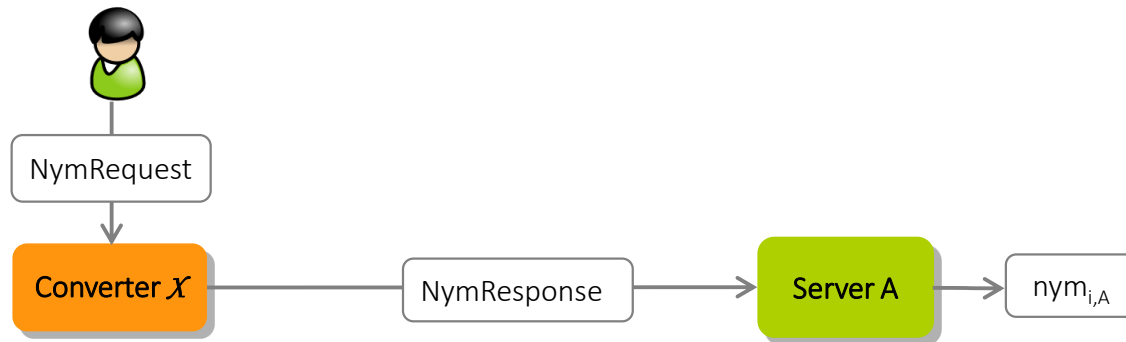
Core Idea

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

Conversion: \mathcal{X} blindly computes $\text{nym}_{i,B} \leftarrow \text{nym}_{i,A}^{x_B / x_A}$

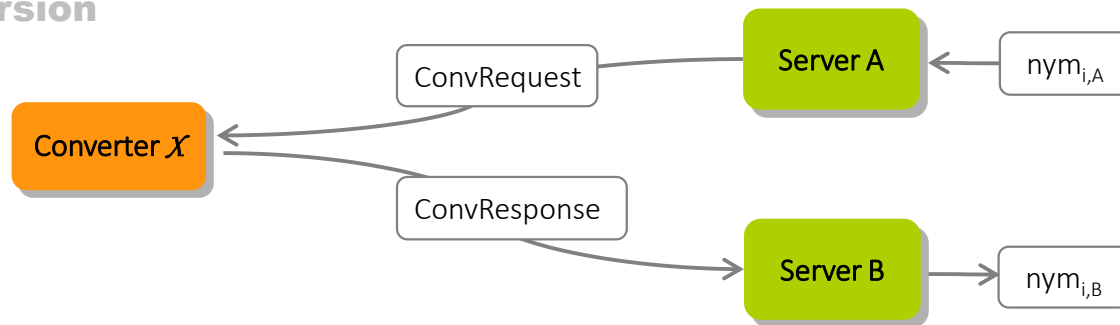


High-level Idea

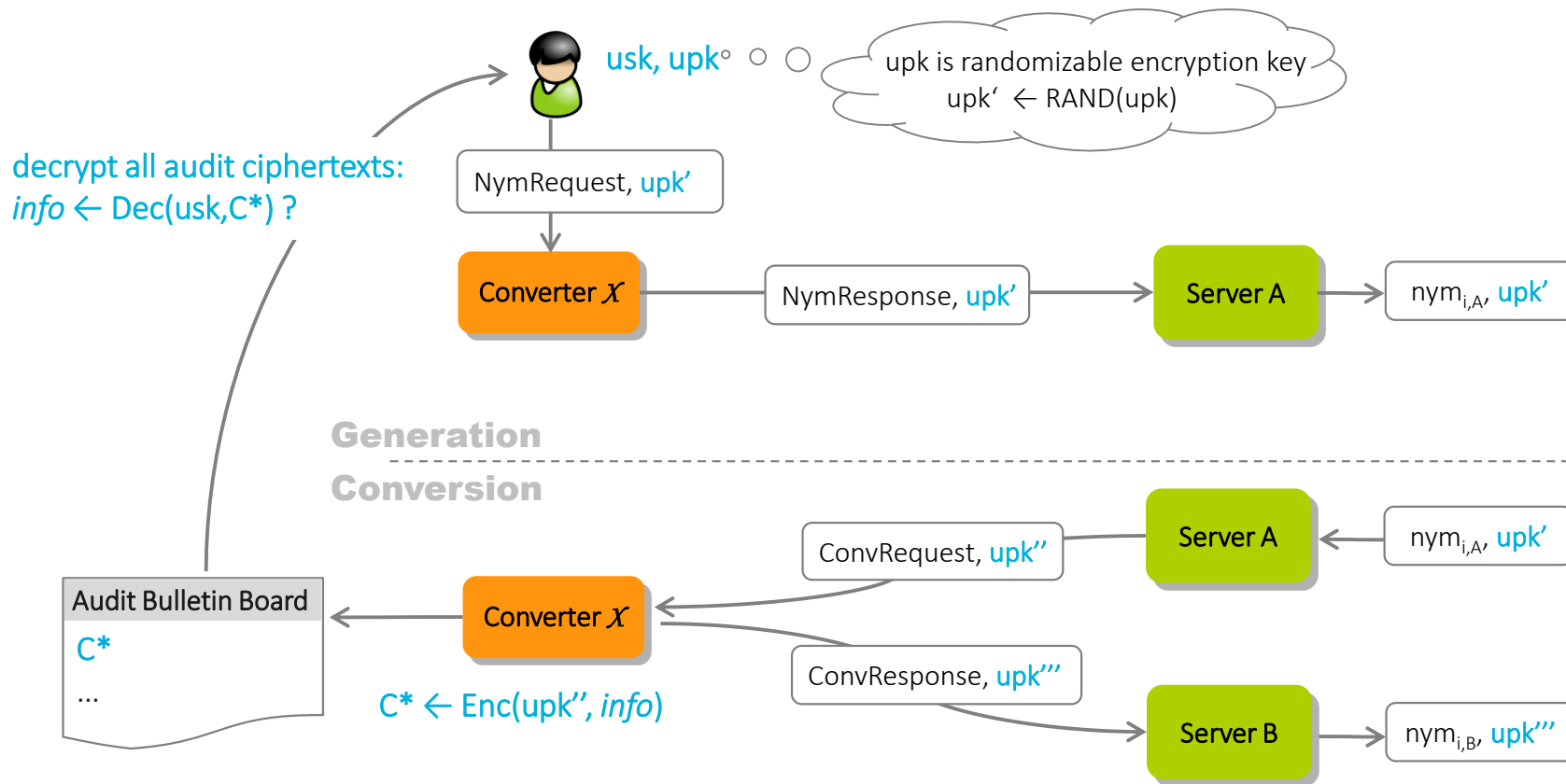


Generation

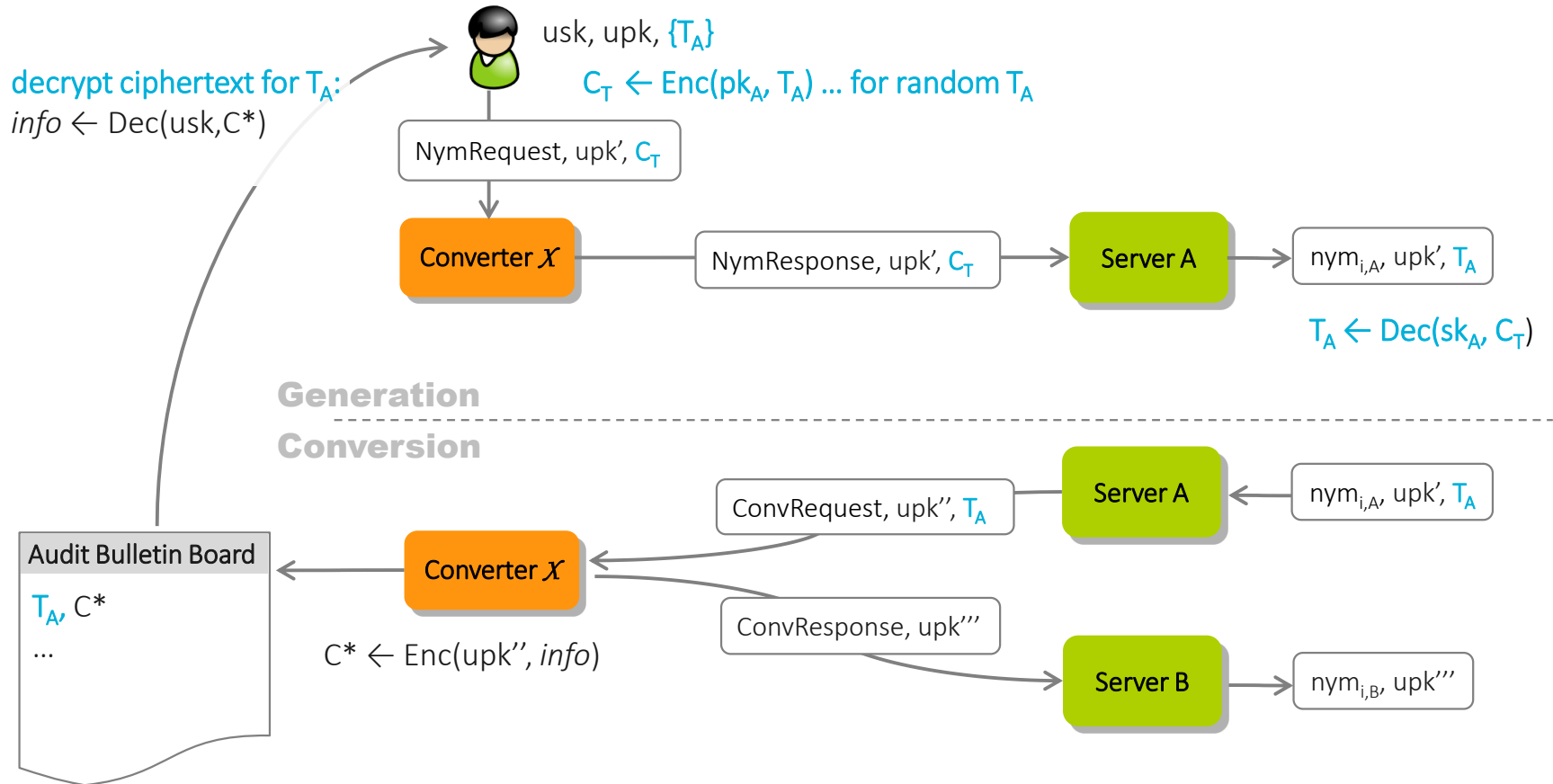
Conversion



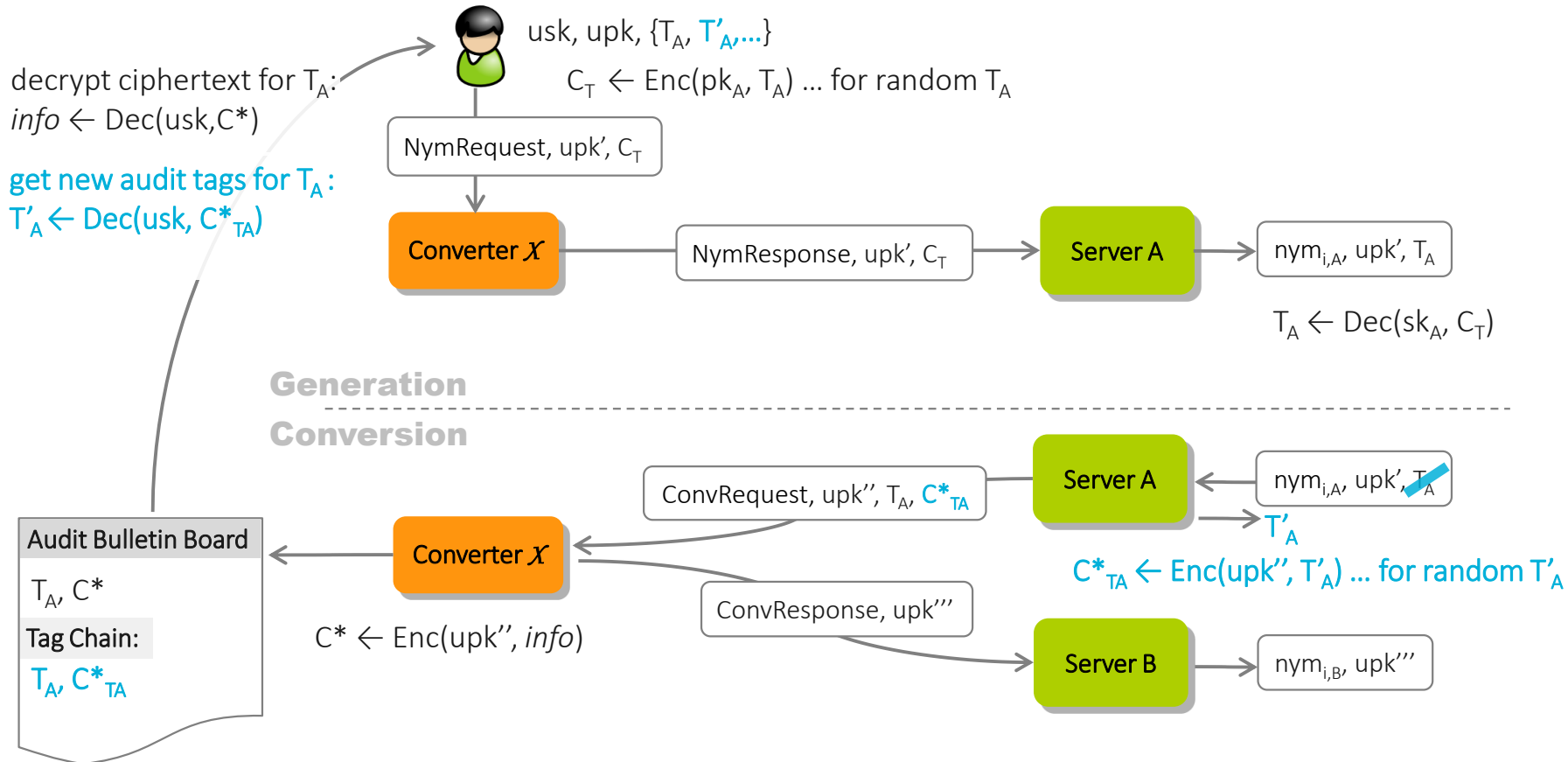
High-level Idea | Adding Auditability



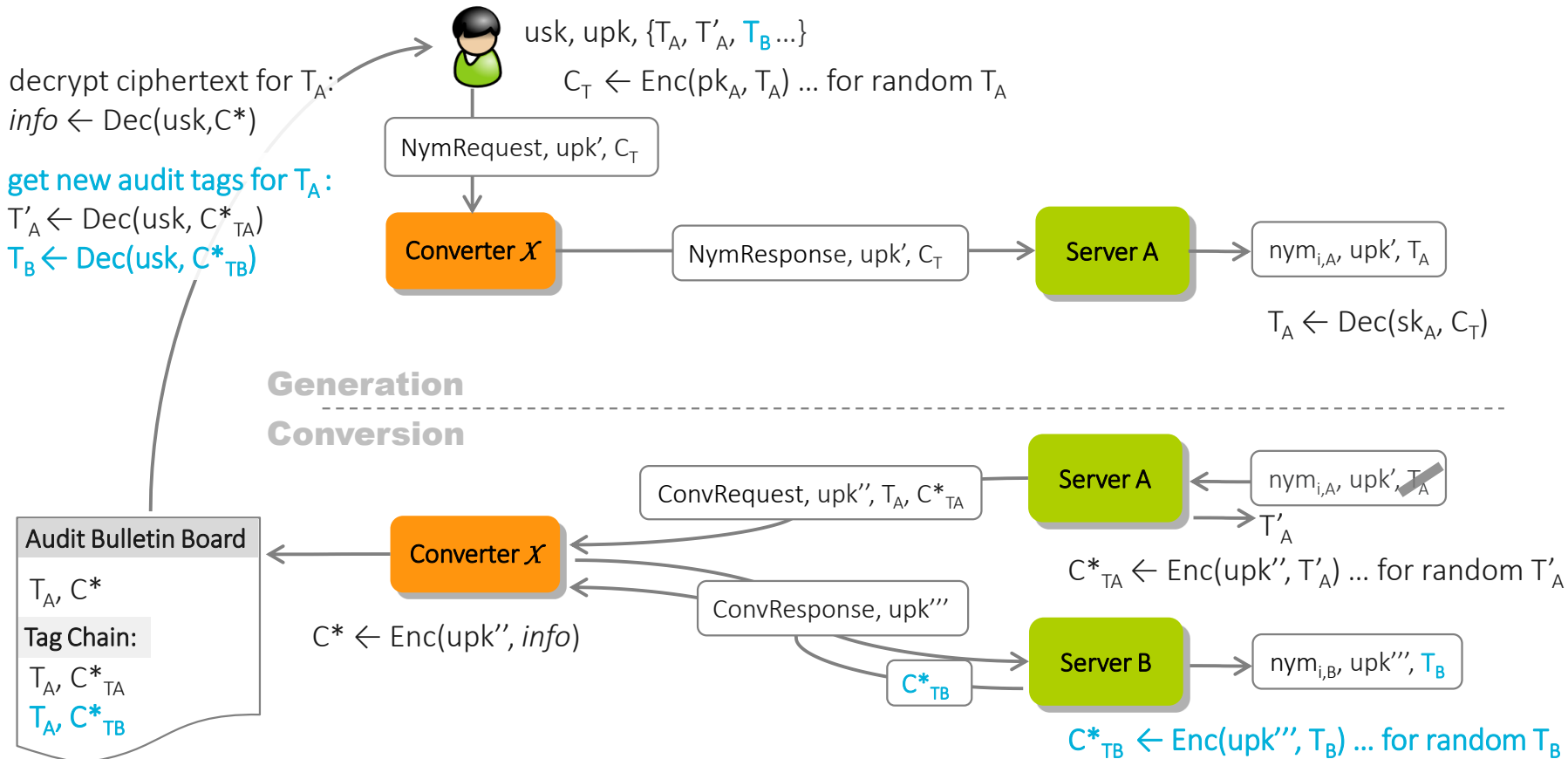
High-level Idea | Adding *Efficient* Auditability (via Audit Tags)



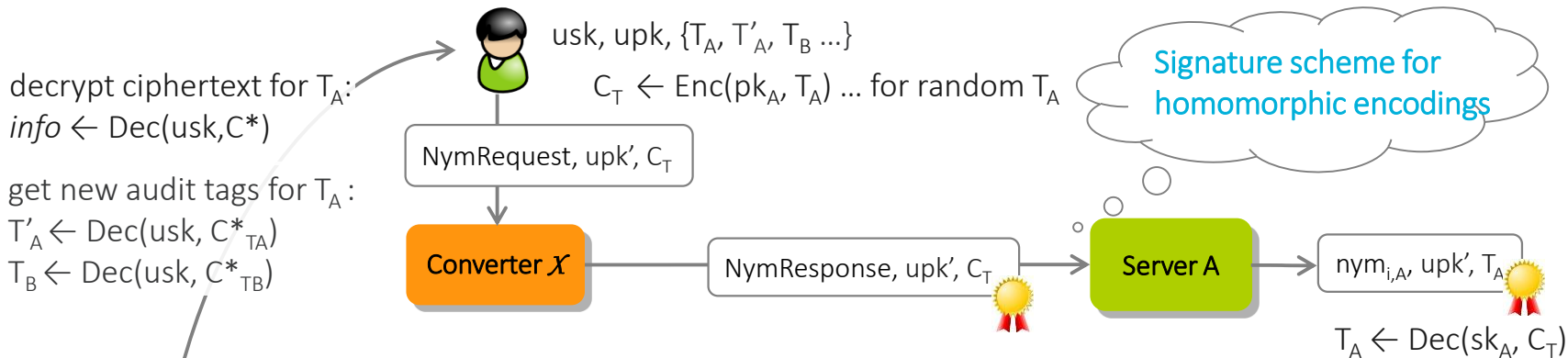
High-level Idea | Adding *Efficient* Auditability (via Audit Tags)



High-level Idea | Adding *Efficient* Auditability (via Audit Tags)

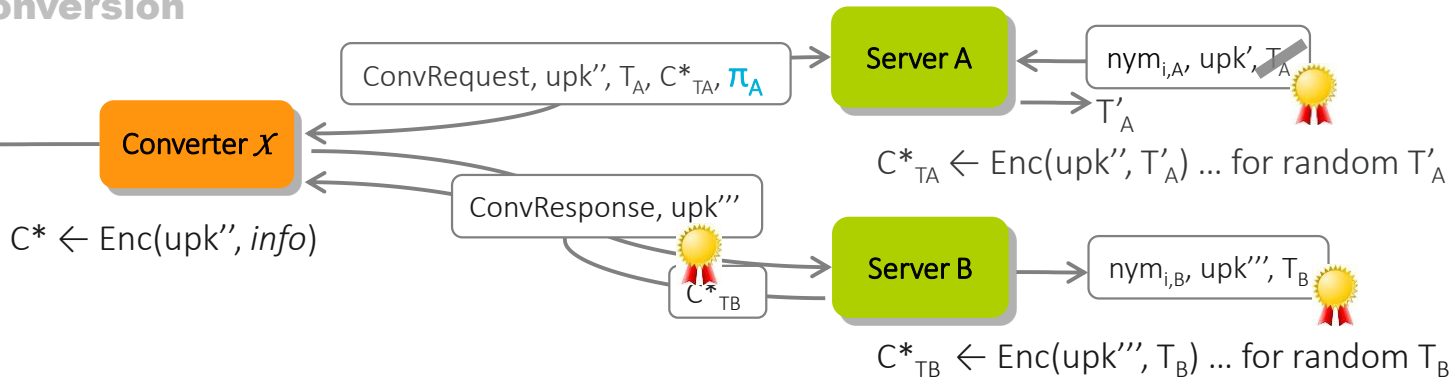
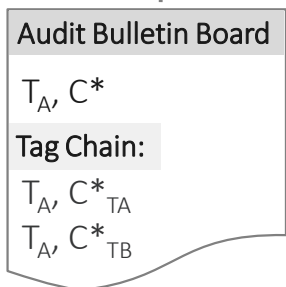


High-level Idea | Security against Active Adversaries



Generation

Conversion



(Un)linkable & Auditable Pseudonyms | Security & Efficiency

- 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

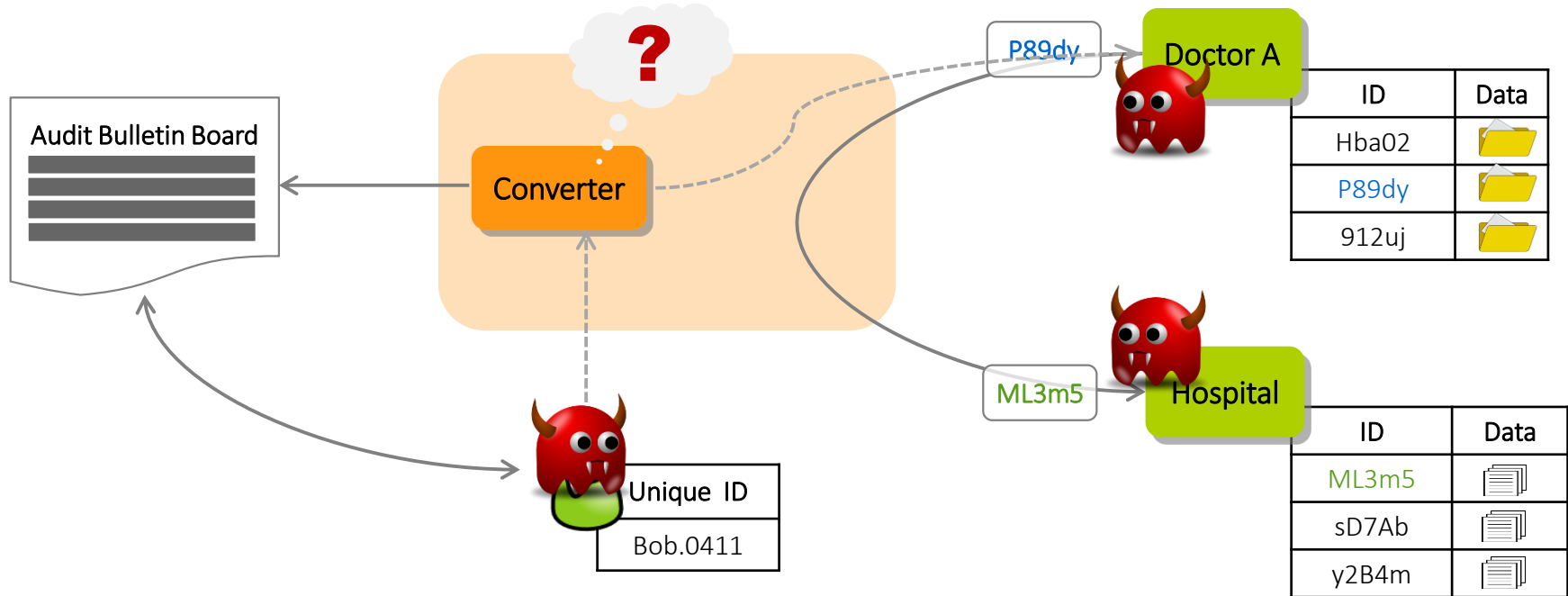
Thanks! Questions?

anj@zurich.ibm.com

(Un)linkable & Auditable Pseudonyms | Efficiency

Pseudonym Generation :	\mathcal{U}_i $22\mathbb{G} + 6\mathbb{Z}_{n^2}^*$	\mathcal{X} $22\mathbb{G} + \tilde{\mathbb{G}} + 6\mathbb{Z}_{n^2}^*$	\mathcal{S}_A $4\mathbb{G}$
Pseudonym Conversion :	\mathcal{S}_A $37\mathbb{G} + 4P$	\mathcal{X} $47\mathbb{G} + 2\tilde{\mathbb{G}} + 4P$	\mathcal{S}_B $13\mathbb{G}$
User Audit :	\mathcal{U}_i $3c \cdot \mathbb{G}$	<i>with c denoting the amount of conversions for \mathcal{U}_i</i>	

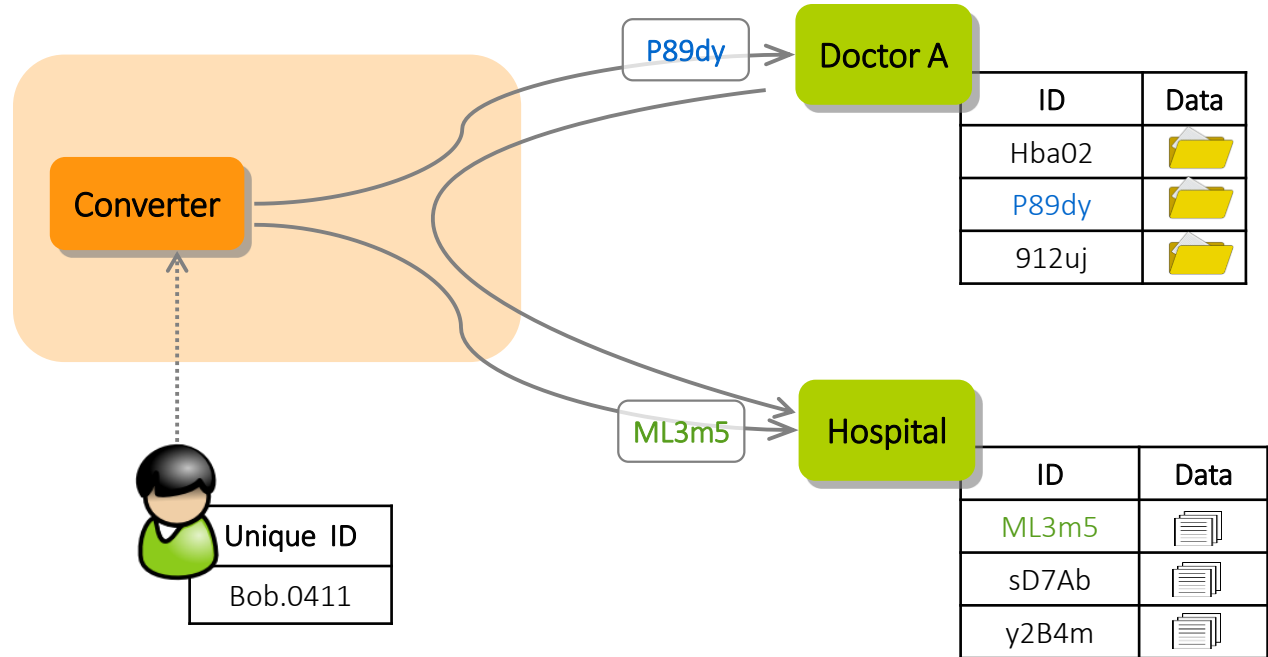
(Un)linkable Pseudonyms | Corruption Model



- 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

