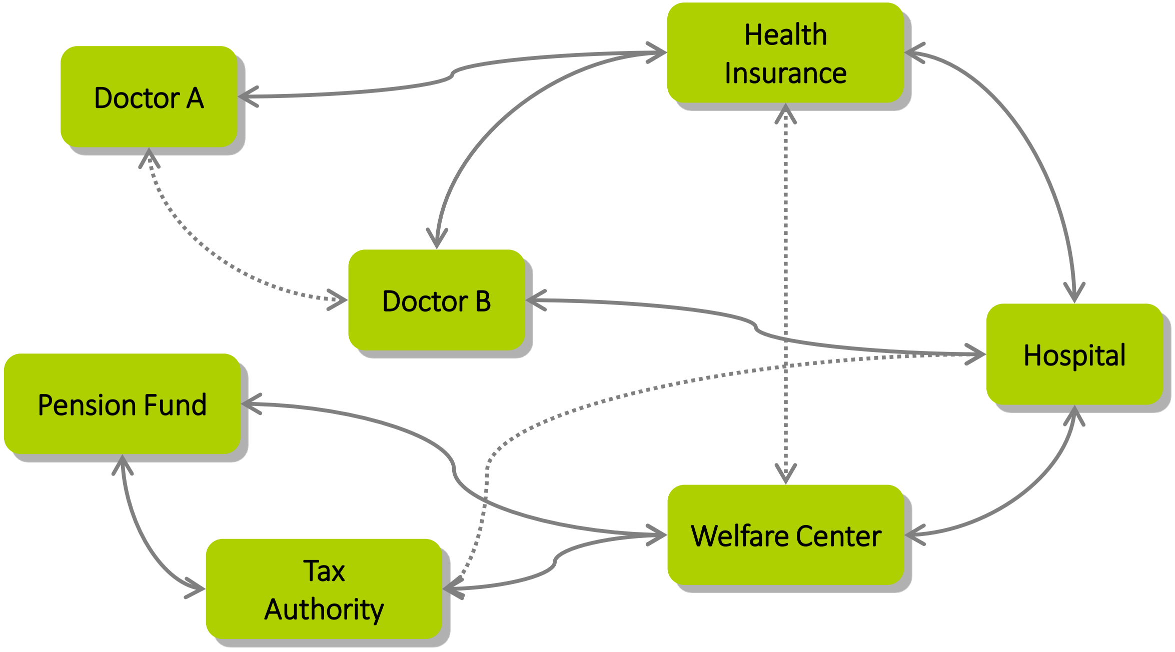
Privacy-Preserving and Auditable Data Exchange

Anja Lehmann
IBM Research – Zurich

[CL15] Camenisch, Lehmann. (Un)linkable Pseudonyms for Governmental Databases. CCS’15.
[CL17] Camenisch, Lehmann. Privacy-Preserving User-Auditable Pseudonym Systems. IEEE EuroSP’17

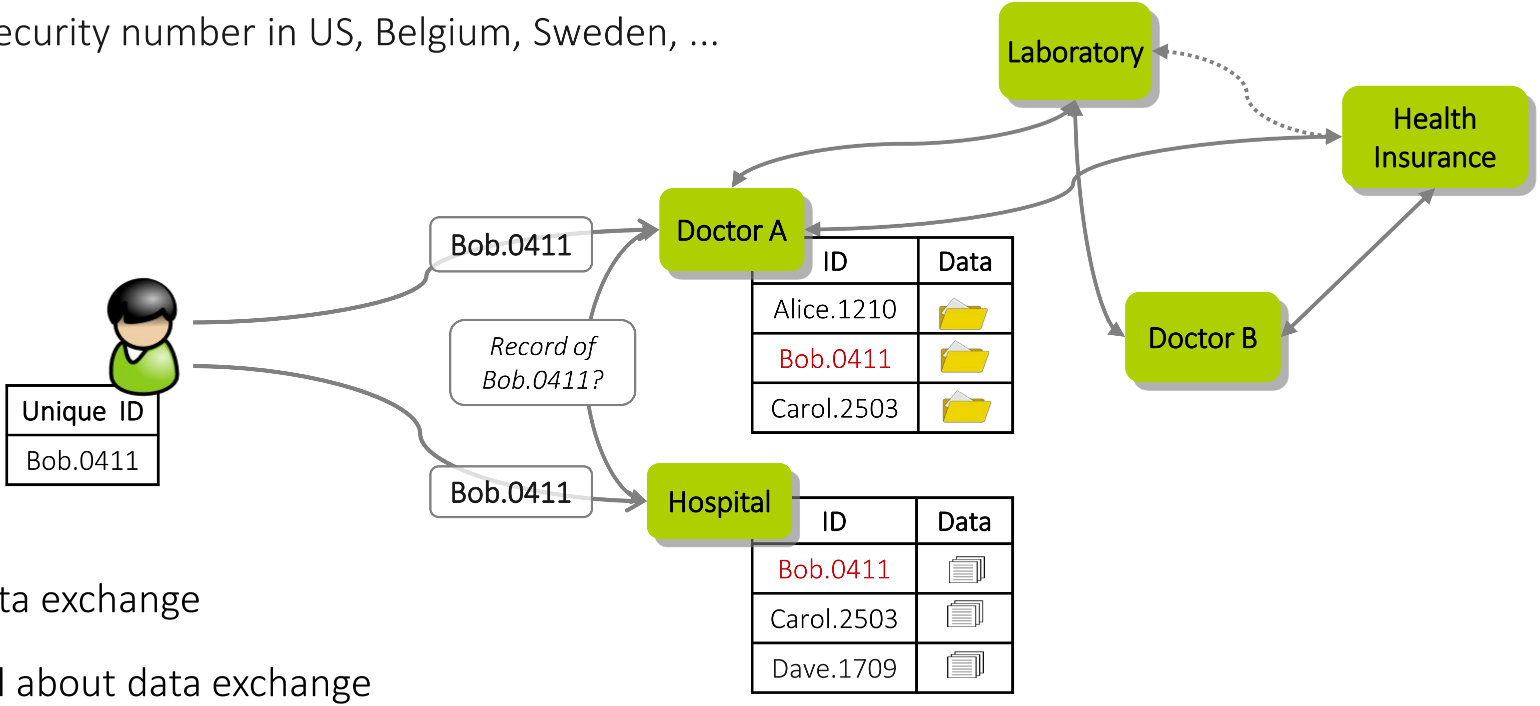
How to maintain related yet distributed data?

- Use case: social security system, eHealth …
  - Different entities maintain data of citizens
  - Eventually data needs to be exchanged or correlated



Globally Unique Identifier

- E.g., social security number in US, Belgium, Sweden, ...



Unique ID
Bob.0411

ID	Data
Alice.1210	📂
Bob.0411	📂
Carol.2503	📂

ID	Data
Bob.0411	🗐
Carol.2503	🗐
Dave.1709	🗐

+ simple data exchange

– no control about data exchange

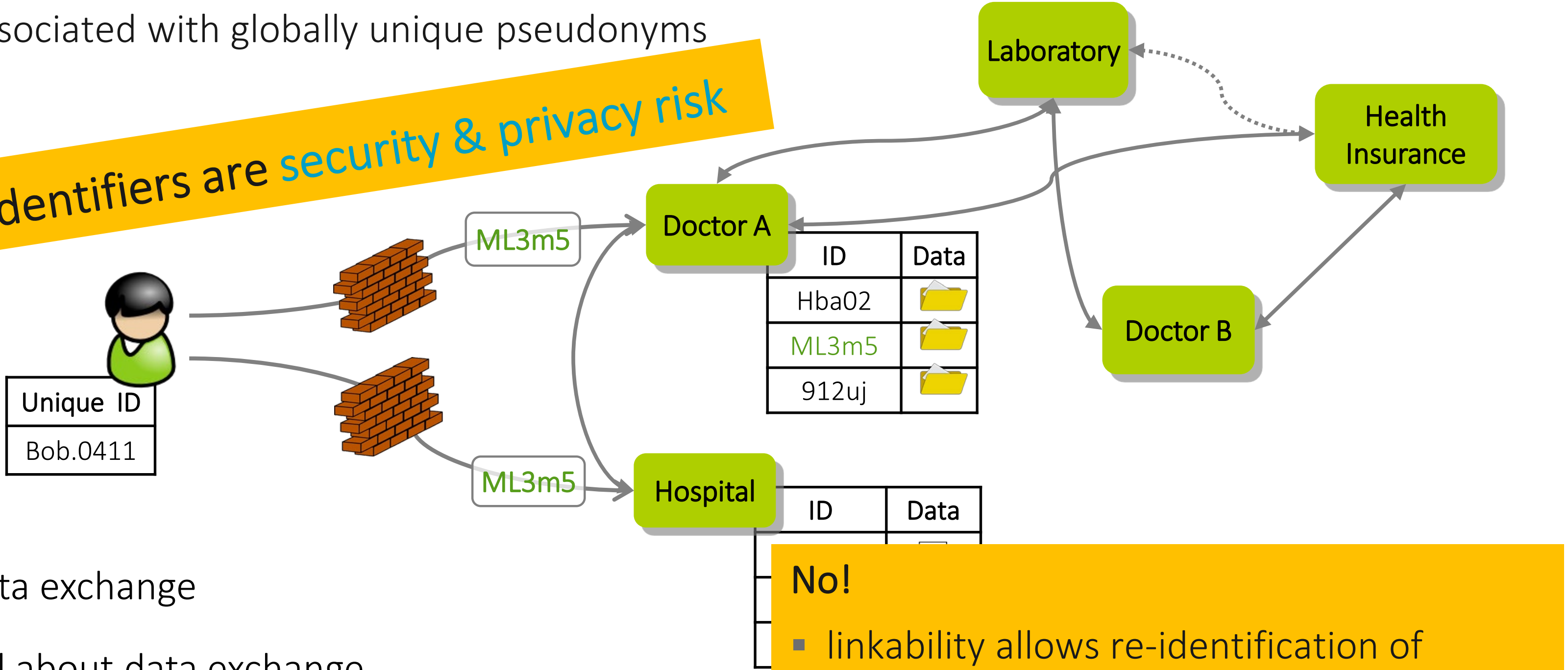
– PII data is very sensitive→ requires strong protection

– if records are lost, pieces can be linked together

Globally Unique *Pseudonyms*

- Data gets associated with globally unique pseudonyms

Unique identifiers are security & privacy risk



ID	Data
Hba02	📁
ML3m5	📁
912uj	📁

+ simple data exchange

– no control about data exchange

– ~~PII data is very sensitive → requires strong protection~~

– if records are lost, pieces can be linked together

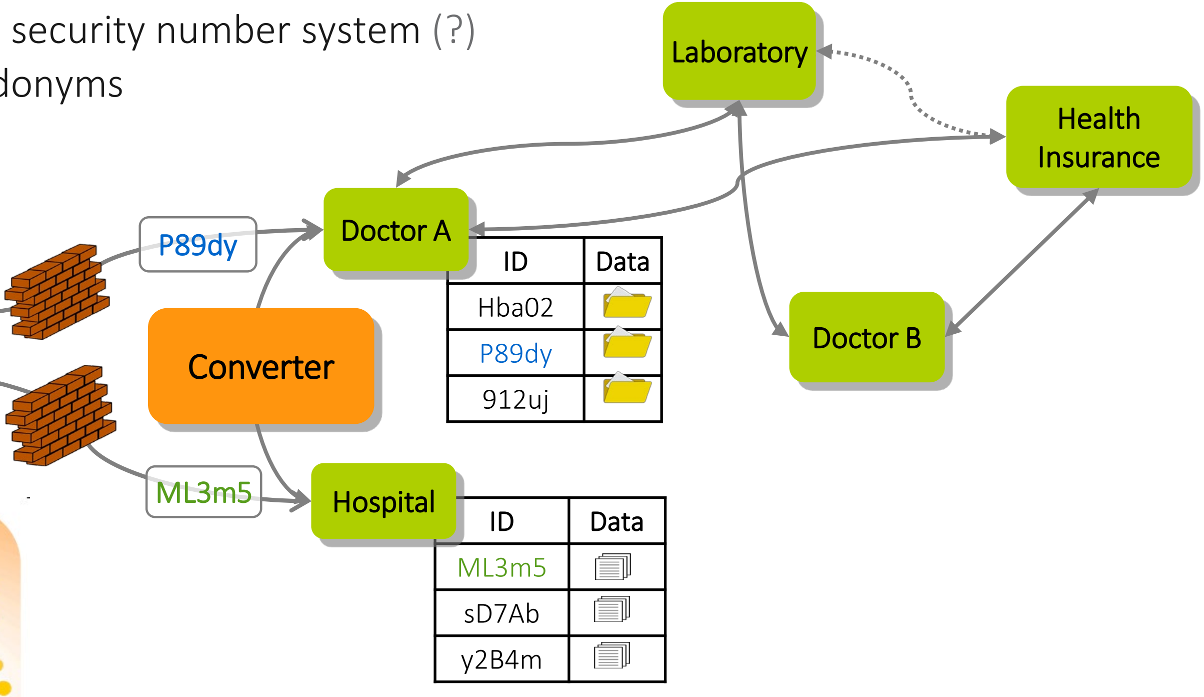
No!

- linkability allows re-identification of “anonymized” data (e.g. Netflix challenge)
- partial corruption reveals identity

Pseudonym System | Motivation

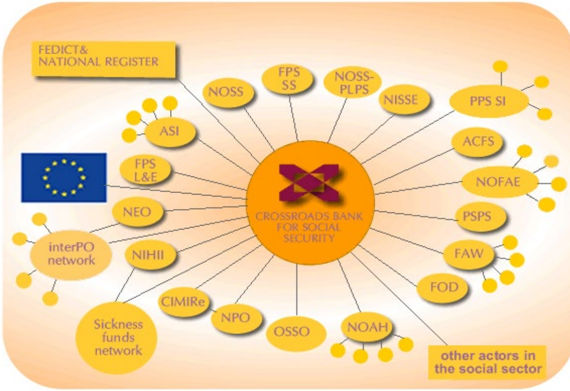
- Design of new Japan eID / social security number system (?)
- Different, context-specific pseudonyms

Unique ID
Bob.0411



ID	Data
Hba02	📁
P89dy	📁
912uj	📁

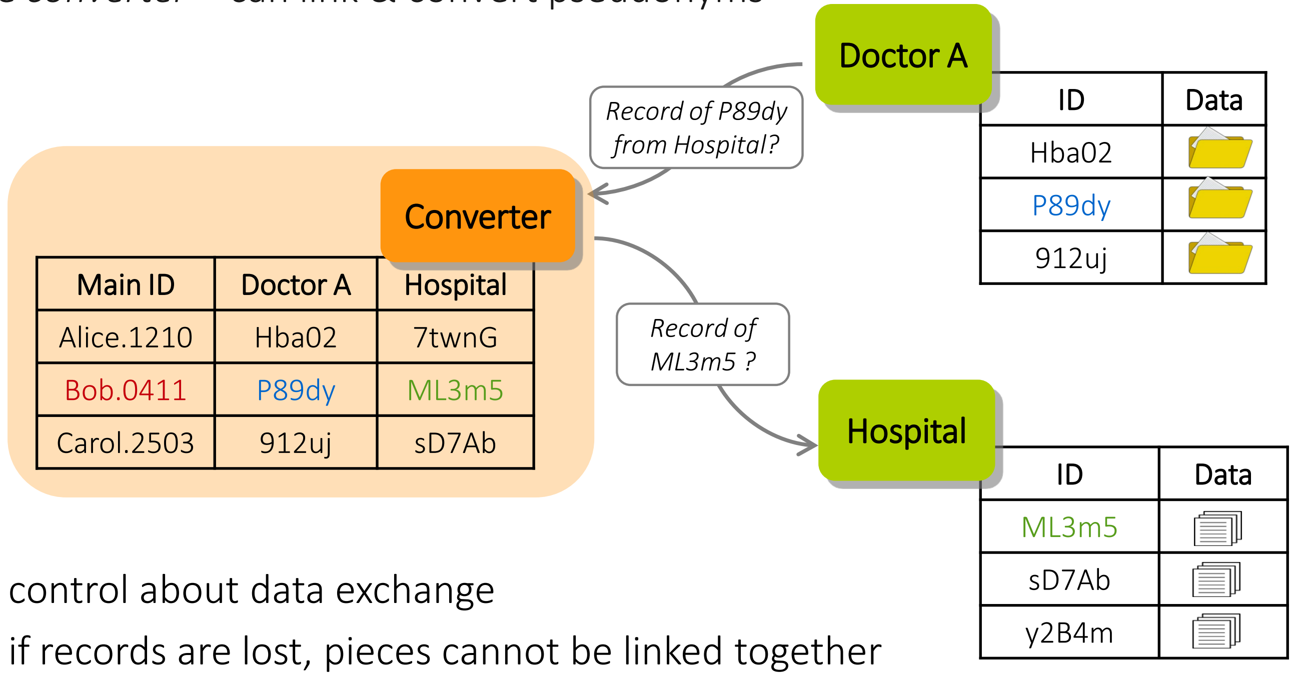
ID	Data
ML3m5	🗐
sD7Ab	🗐
y2B4m	🗐



Inspired by Belgium’s Crossroad Bank for Social Security (CBSS)

Pseudonym System | Local Pseudonyms & *Trusted* Converter

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



Converter

Main ID	Doctor A	Hospital
Alice.1210	Hba02	7twnG
Bob.0411	P89dy	ML3m5
Carol.2503	912uj	sD7Ab

Record of P89dy from Hospital?

Record of ML3m5 ?

Doctor A

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

Hospital

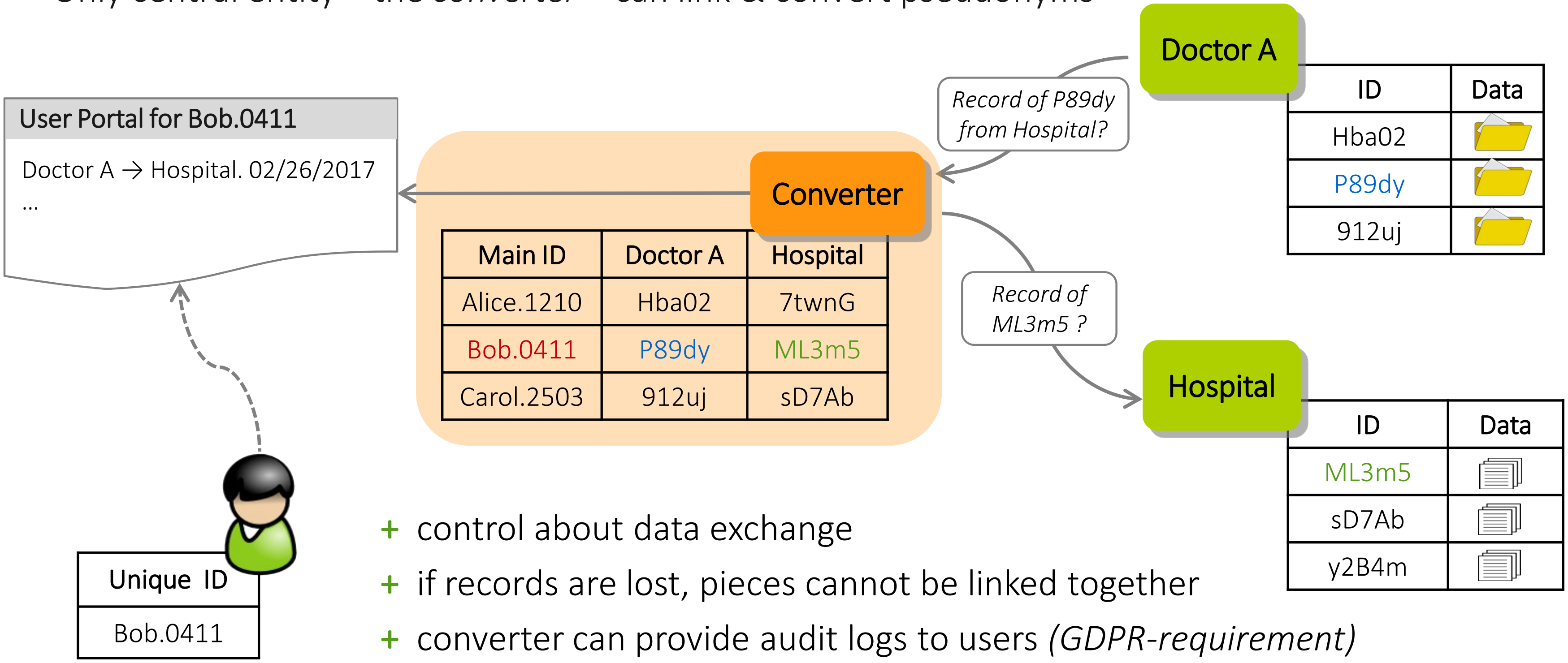
ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

+ control about data exchange

+ if records are lost, pieces cannot be linked together

Pseudonym System | Local Pseudonyms & *Trusted* Converter

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



User Portal for Bob.0411

Doctor A → Hospital. 02/26/2017
...

Converter

Main ID	Doctor A	Hospital
Alice.1210	Hba02	7twnG
Bob.0411	P89dy	ML3m5
Carol.2503	912uj	sD7Ab

Record of P89dy from Hospital?

Record of ML3m5 ?

Doctor A

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

Hospital

ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

Unique ID
Bob.0411

+ control about data exchange

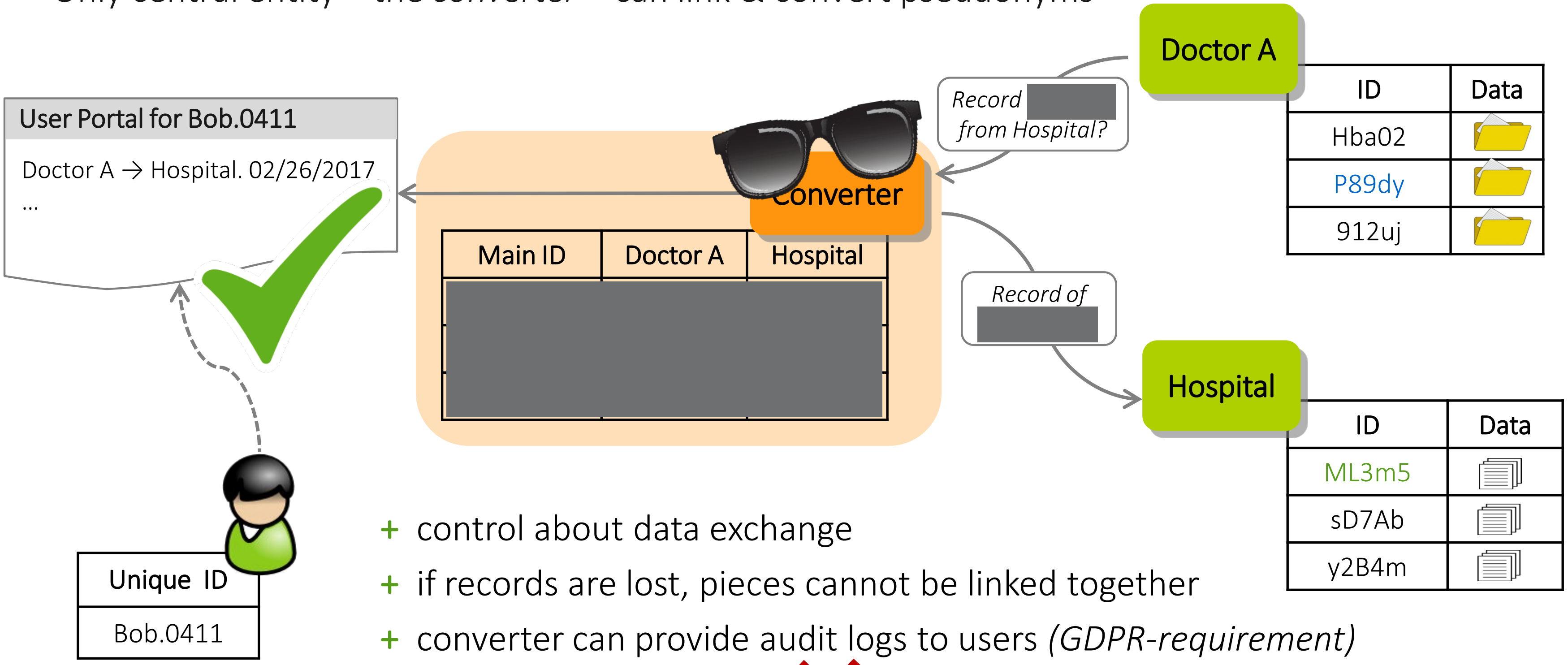
+ if records are lost, pieces cannot be linked together

+ converter can provide audit logs to users *(GDPR-requirement)*

– **converter learns all request & knows all correlations**

Pseudonym System | Local Pseudonyms & *Oblivious* Converter

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



User Portal for Bob.0411

Doctor A → Hospital. 02/26/2017
...

Unique ID
Bob.0411

Main ID	Doctor A	Hospital
[redacted]	[redacted]	[redacted]

Record [redacted] *from Hospital?*

Record of [redacted]

Doctor A

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

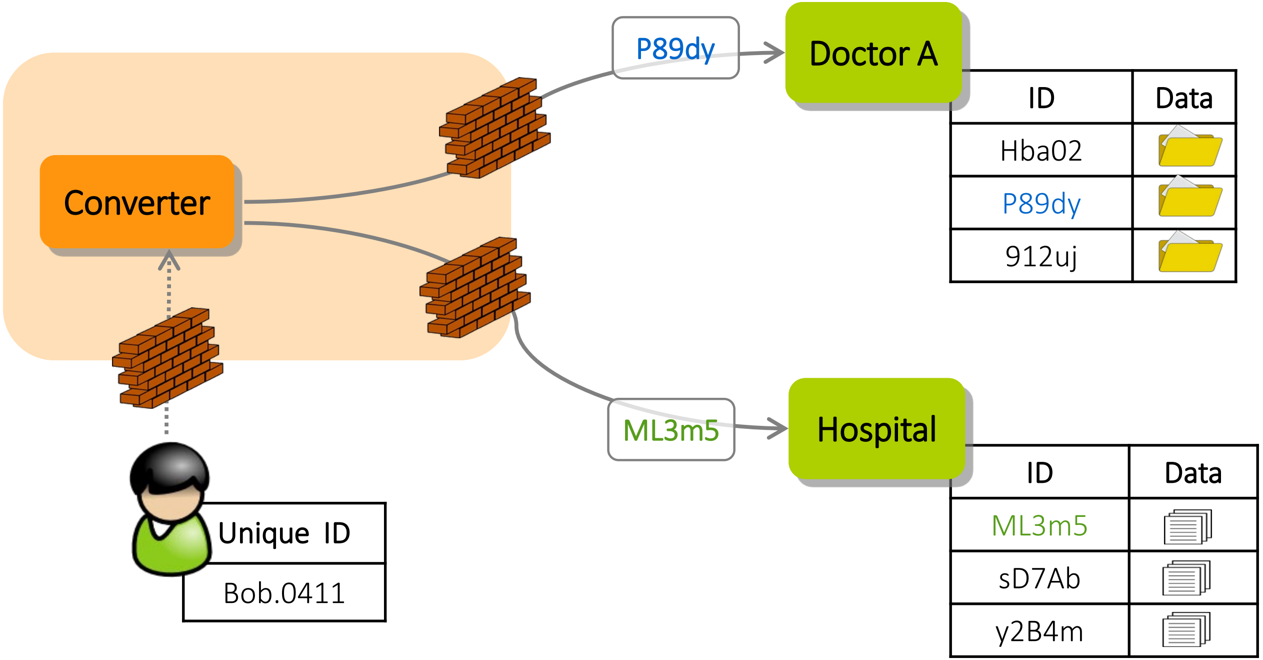
Hospital

ID	Data
ML3m5	▤
sD7Ab	▤
y2B4m	▤

+ 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



Unique ID
Bob.0411

Doctor A:

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

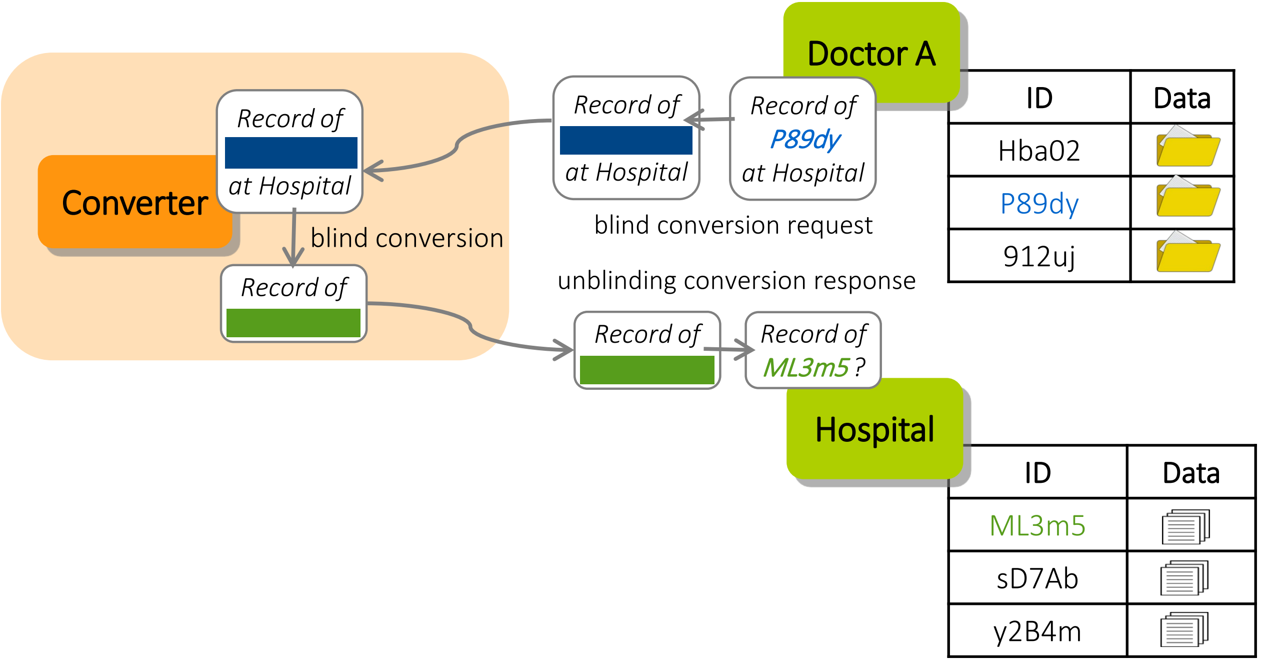
Hospital:

ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

- [CL15] generation triggered by converter, knows unique IDs
- [CL17] oblivious pseudonym generation triggered by user

(Un)linkable Pseudonyms | Pseudonym Conversion

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



Doctor A

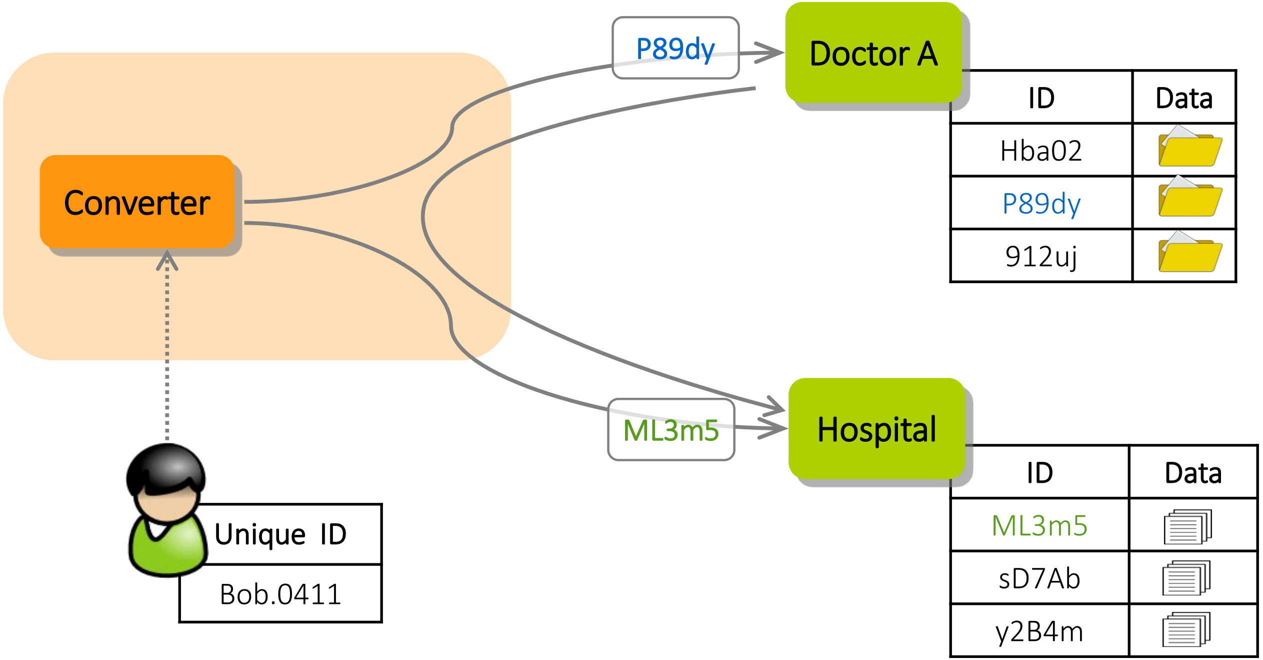
ID	Data
Hba02	📁
P89dy	📁
912uj	📁

Hospital

ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

(Un)linkable Pseudonyms | Consistency

- pseudonym generation is deterministic & consistent with blind conversion



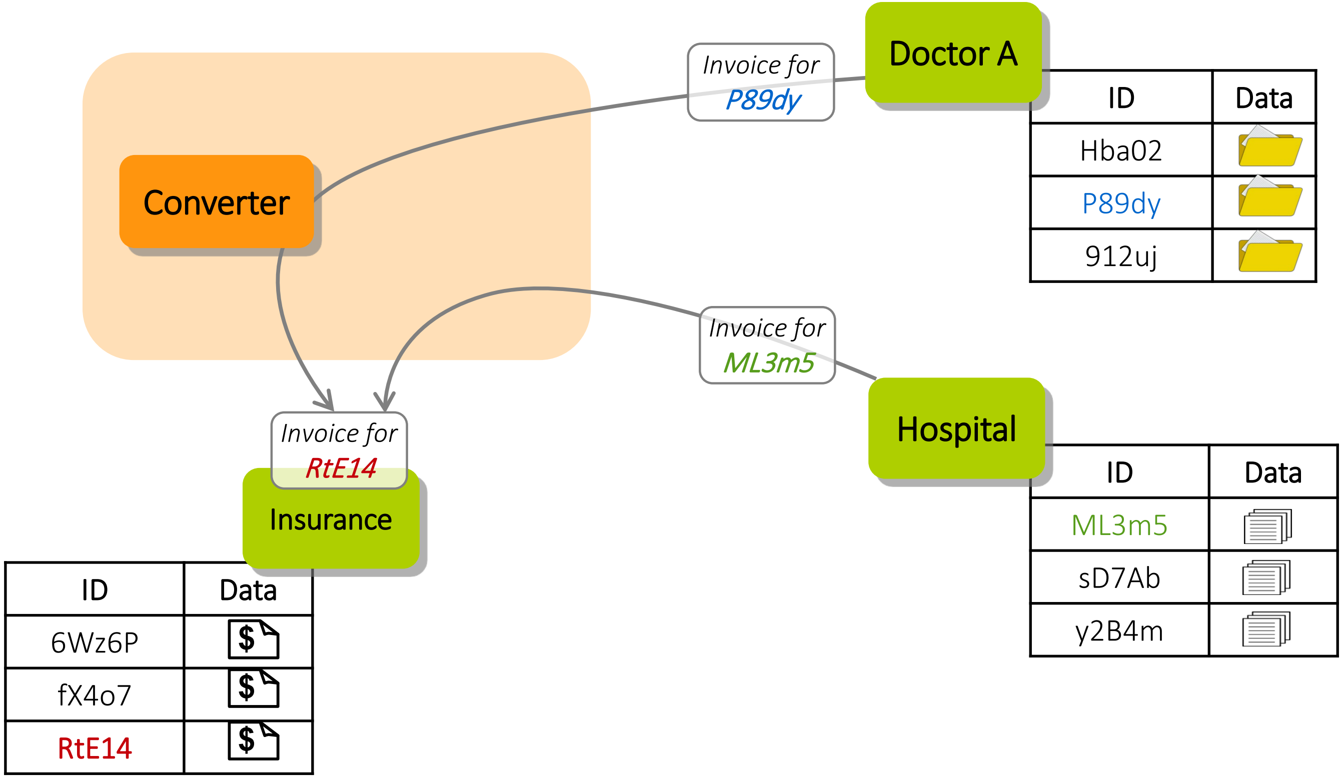
Unique ID
Bob.0411

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

(Un)linkable Pseudonyms | Consistency

- pseudonym conversions are transitive, unlinkable data can be aggregated



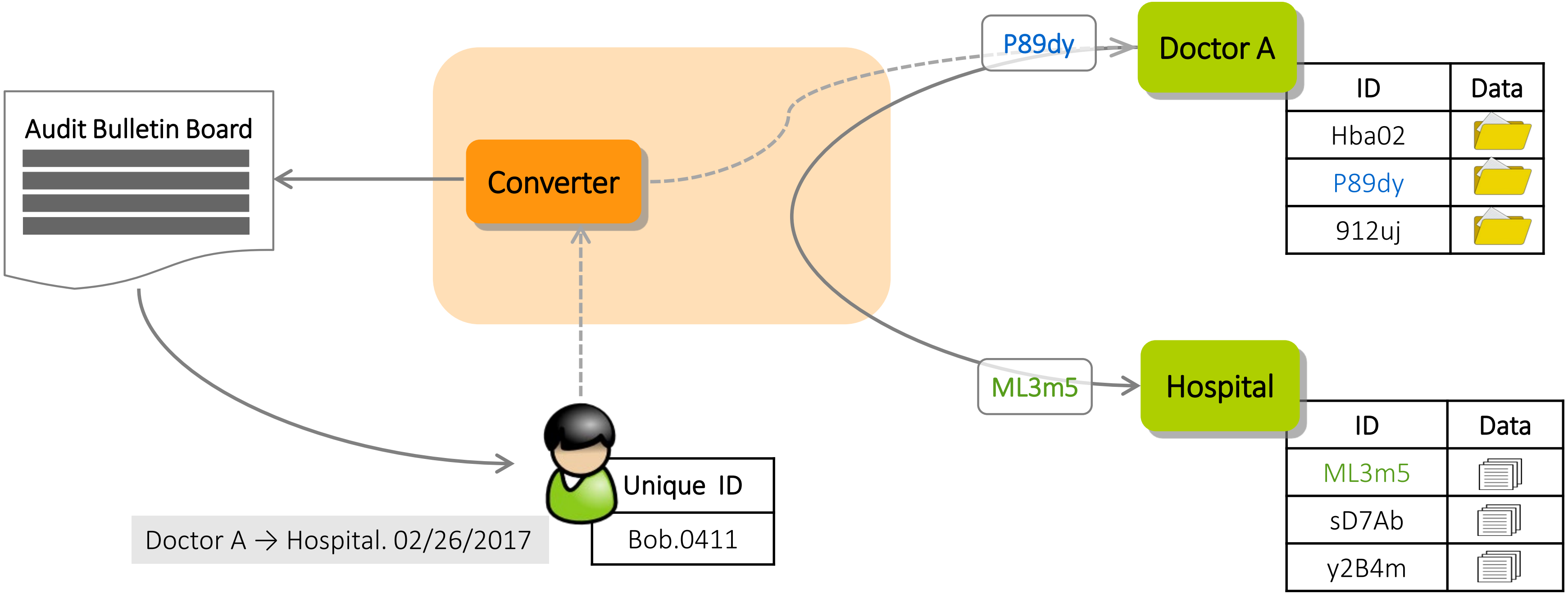
ID	Data
Hba02	📁
P89dy	📁
912uj	📁

ID	Data
ML3m5	🗐
sD7Ab	🗐
y2B4m	🗐

ID	Data
6Wz6P	$
fX4o7	$
RtE14	$

(Un)linkable Pseudonyms | User Audits

- [CL17] every pseudonym conversion triggers blind generation of audit log entry



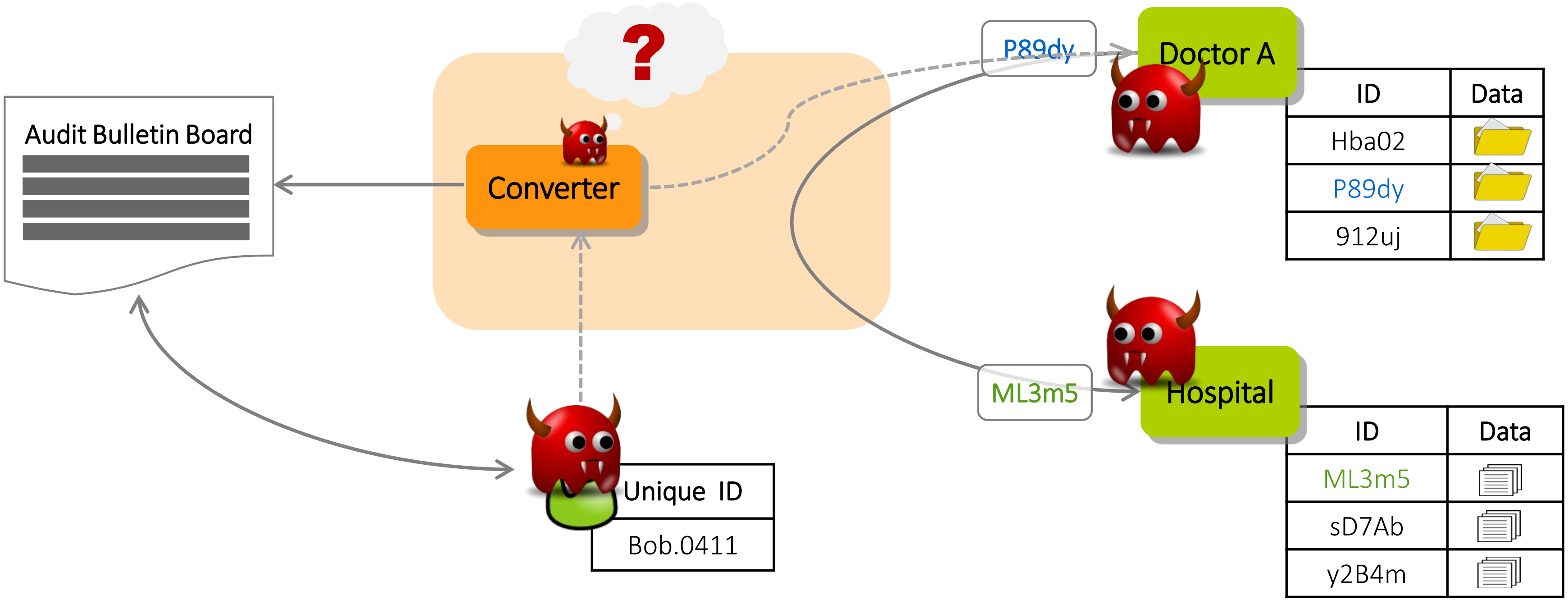
ID	Data
Hba02	📁
P89dy	📁
912uj	📁

ID	Data
ML3m5	🗐
sD7Ab	🗐
y2B4m	🗐

Unique ID
Bob.0411

Doctor A → Hospital. 02/26/2017

(Un)linkable Pseudonyms | Corruption Model



Unique ID
Bob.0411

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

- Formal security model via ideal functionality in UC Framework
  - servers and users can be fully corrupt
  - converter at most honest-but-curious (w/o audits even fully corrupt [CL15])

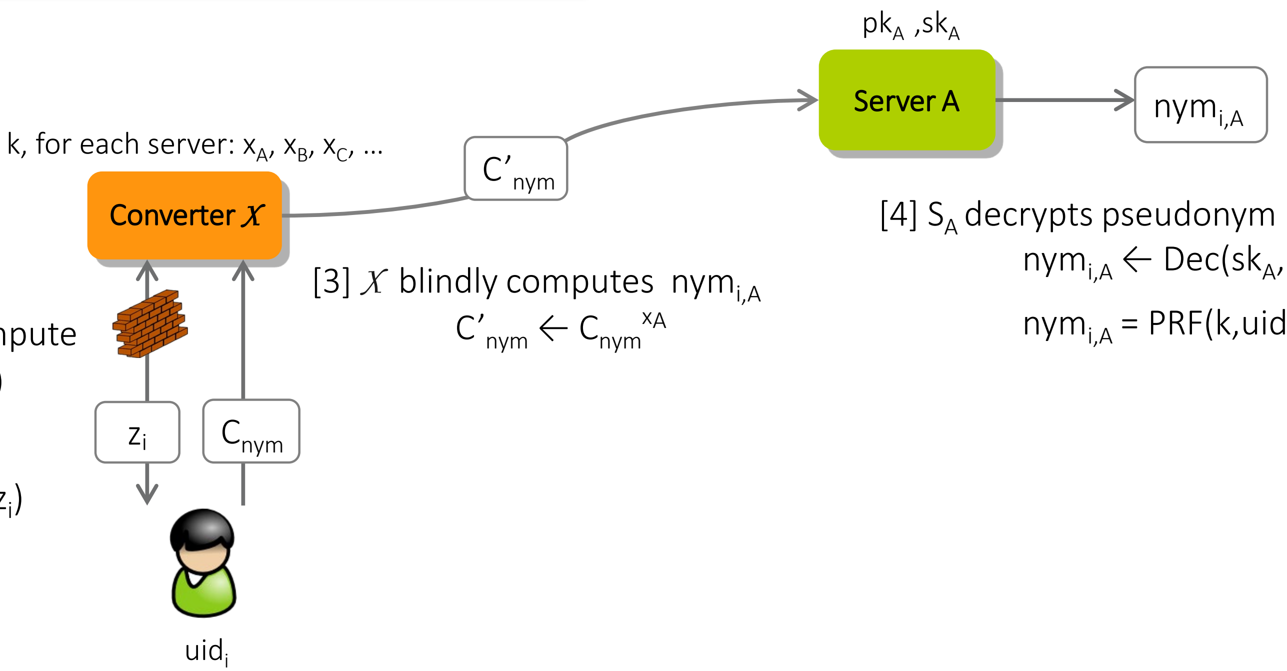
Our Protocol

- 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 $nym_{i,A} \leftarrow PRF(k,uid_i)^{xA}$



[1] $\mathcal{X}$ and U_i jointly compute

$z_i \leftarrow OPRF(k,uid_i)$

[2] U_i encrypts z_i for S_A

$C_{nym} \leftarrow Enc(pk_A,z_i)$

[3] $\mathcal{X}$ blindly computes $nym_{i,A}$

$C'_{nym} \leftarrow C_{nym}{}^{xA}$

[4] S_A decrypts pseudonym

$nym_{i,A} \leftarrow Dec(sk_A,C'_{nym})$

$nym_{i,A} = PRF(k,uid_i)^{xA}$

High-level Idea | Pseudonym Conversion

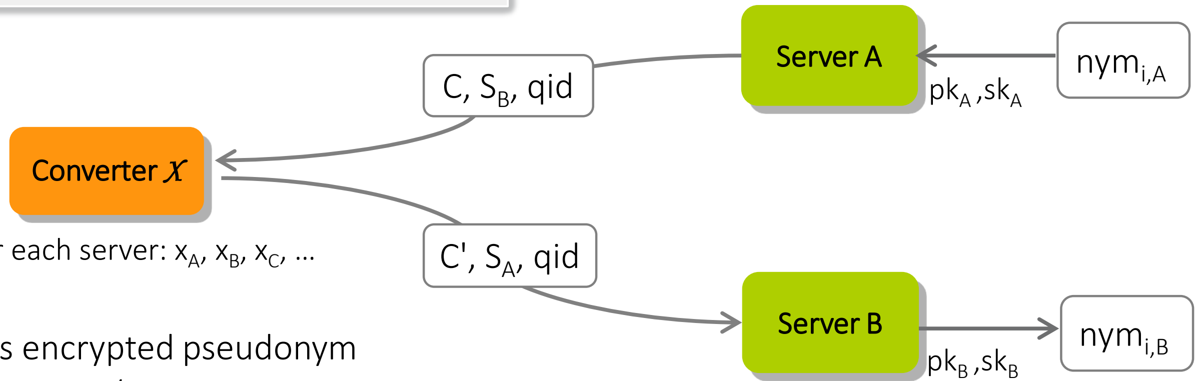
Core Idea

Generation: $\mathcal{X}$ blindly computes $nym_{i,A} \leftarrow PRF(k,uid_i)^{xA}$

Conversion: $\mathcal{X}$ blindly computes $nym_{i,B} \leftarrow nym_{i,A}^{xB/xA}$

[1] S_A encrypts $nym_{i,A}$ under S_B's key

$C \leftarrow Enc(pk_B, nym_{i,A})$



[2] $\mathcal{X}$ blindly transforms encrypted pseudonym

$C' \leftarrow C^{\Delta}$ with $\Delta = x_B / x_A$

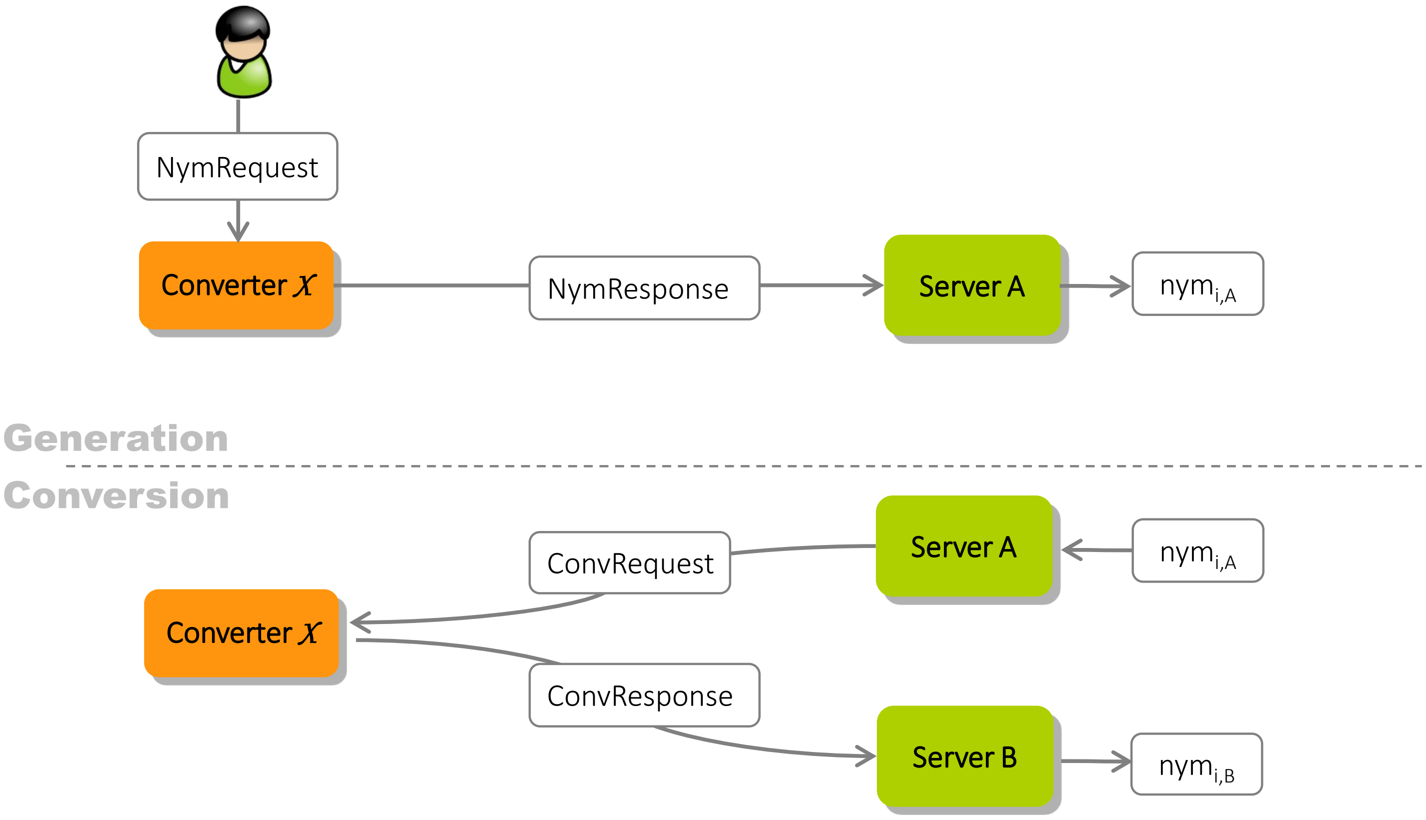
$$
\begin{aligned}
C' &= Enc(pk_B, nym_{i,A})^{xB / xA} \\
&= Enc(pk_B, PRF(k,uid_i)^{xA})^{xB / xA} \\
&= Enc(pk_B, PRF(k,uid_i)^{xB}) \\
&= Enc(pk_B, nym_{i,B})
\end{aligned}
$$

[3] S_B decrypts converted pseudonym

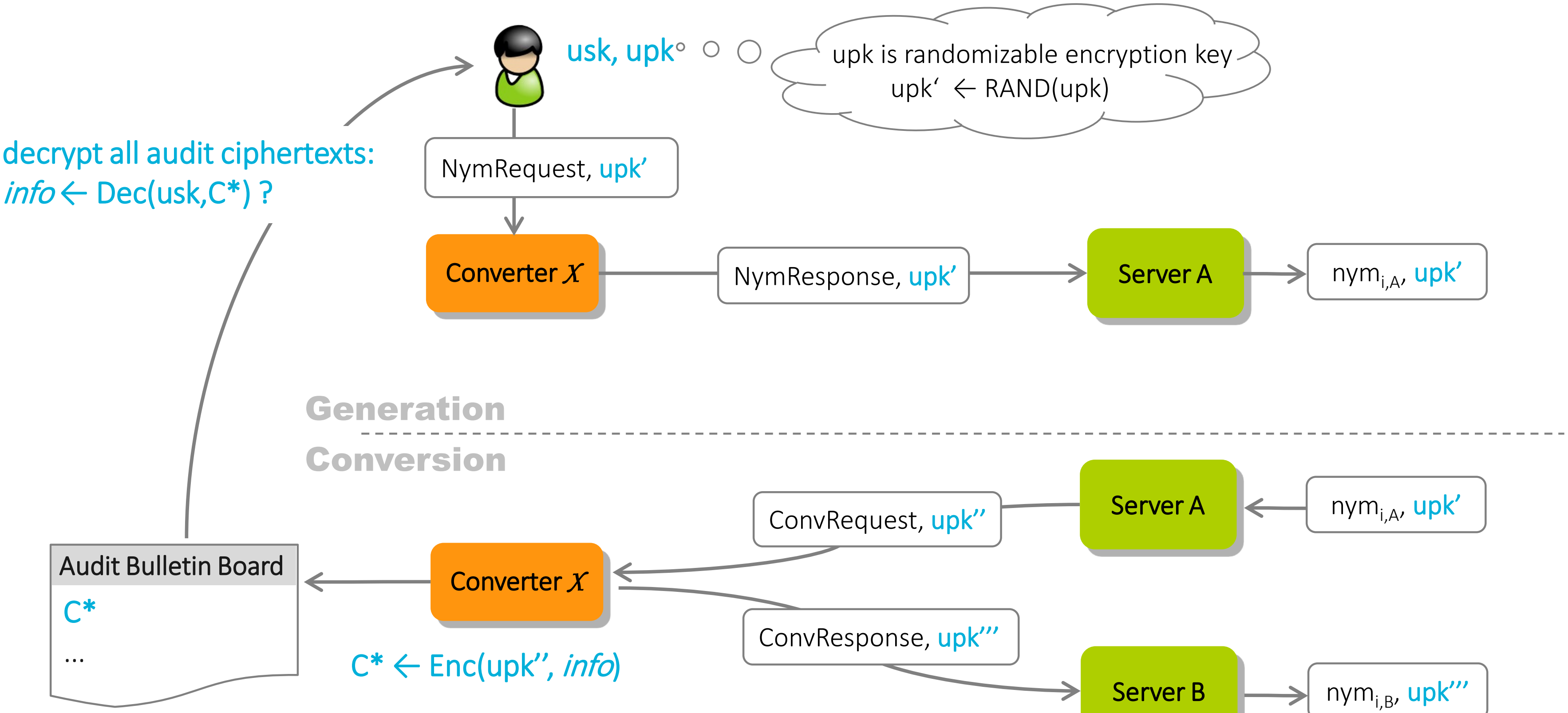
$nym_{i,B} \leftarrow Dec(sk_B, C')$

$nym_{i,B} = PRF(k,uid_i)^{xB}$

High-level Idea | Overview



High-level Idea | Adding Auditability



usk, upk

upk is randomizable encryption key
upk‘ ← RAND(upk)

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

NymRequest, upk’

Converter $\mathcal{X}$

NymResponse, upk’

Server A

nym$_{i,A}$, upk’

Generation

Conversion

Server A

nym$_{i,A}$, upk’

ConvRequest, upk’’

Audit Bulletin Board
C*
…

Converter $\mathcal{X}$

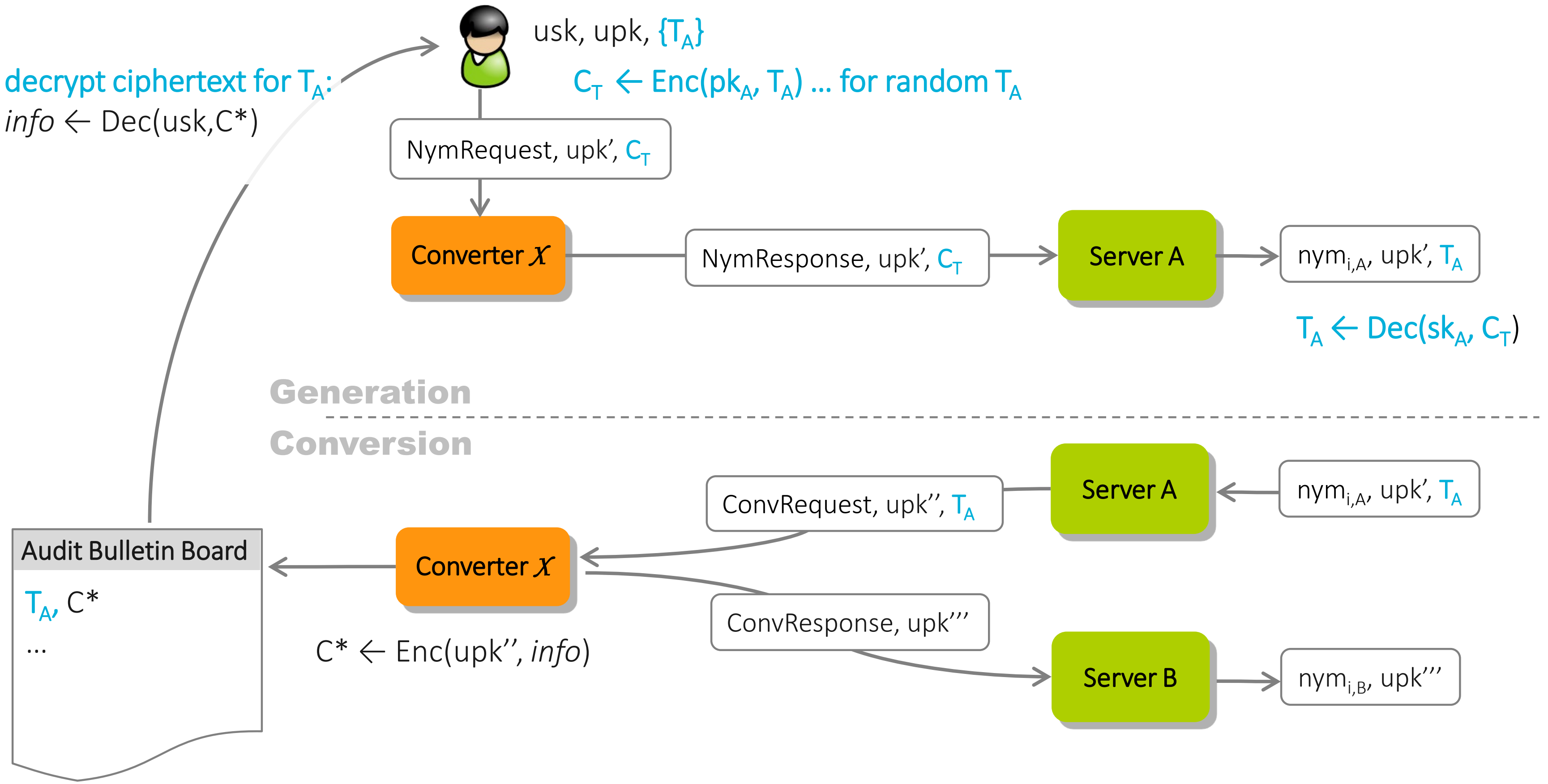
ConvResponse, upk’’’

Server B

nym$_{i,B}$, upk’’’

C* ← Enc(upk’’, *info*)

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



usk, upk, $\{T_A\}$

$C_T \leftarrow \mathrm{Enc}(pk_A, T_A)$ … for random T_A

decrypt ciphertext for T_A:

info ← Dec(usk,C*)

NymRequest, upk', C_T

Converter $\mathcal{X}$

NymResponse, upk', C_T

Server A

$\mathrm{nym}_{i,A}$, upk', T_A

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

Generation

Conversion

Server A

$\mathrm{nym}_{i,A}$, upk', T_A

ConvRequest, upk'', T_A

Audit Bulletin Board

T_A, C*

…

Converter $\mathcal{X}$

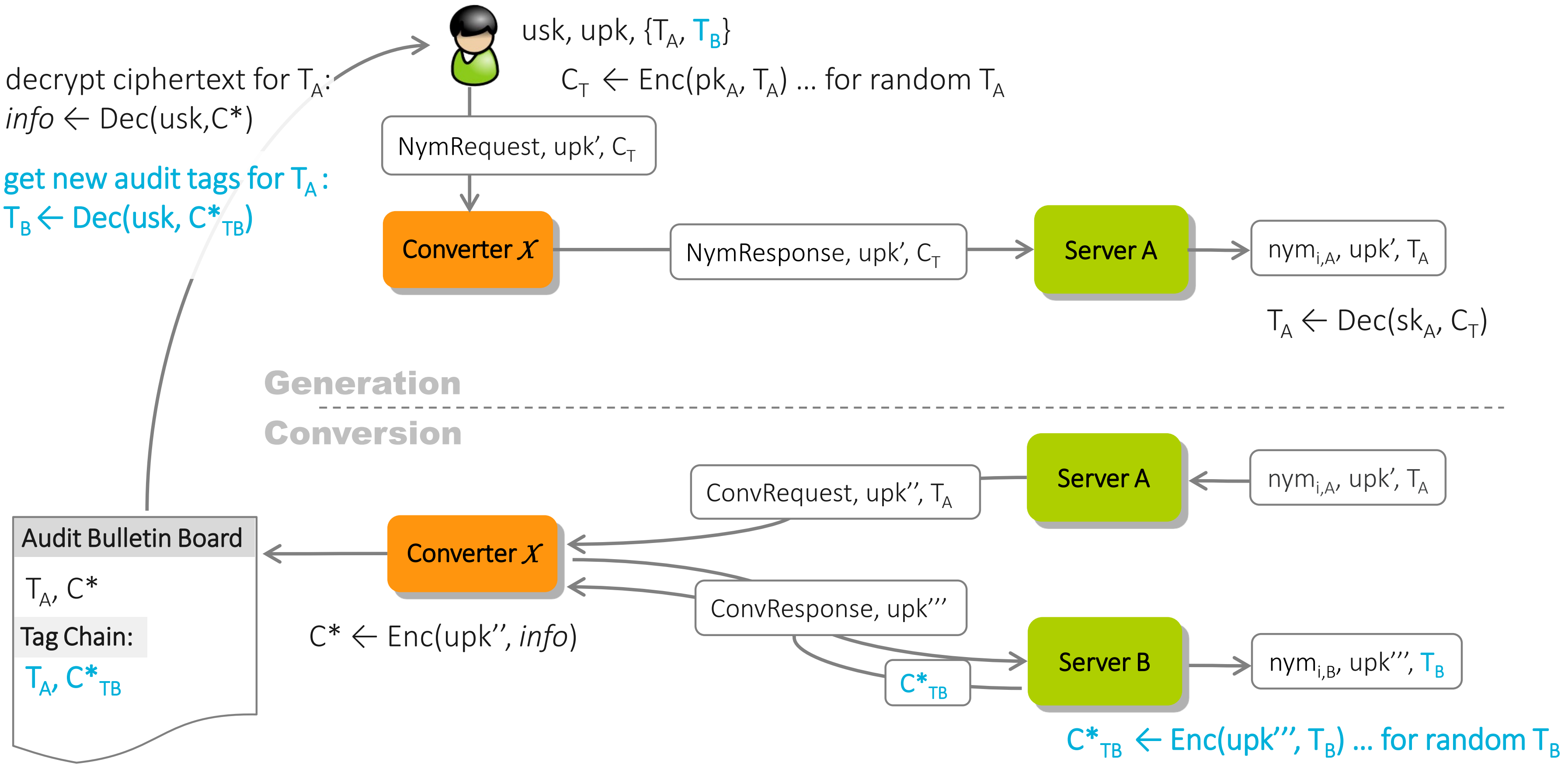
C* ← Enc(upk'', *info*)

ConvResponse, upk'''

Server B

$\mathrm{nym}_{i,B}$, upk'''

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



usk, upk, $\{T_A, T_B\}$

$C_T \leftarrow \mathrm{Enc}(pk_A, T_A)$ … for random T_A

decrypt ciphertext for T_A:
info ← Dec(usk,C*)

get new audit tags for T_A :
$T_B \leftarrow \mathrm{Dec}(usk, C^*_{TB})$

NymRequest, upk’, C_T

Converter $\mathcal{X}$

NymResponse, upk’, C_T

Server A

$nym_{i,A}$, upk’, T_A

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

Generation

Conversion

Server A

$nym_{i,A}$, upk’, T_A

ConvRequest, upk’’, T_A

Converter $\mathcal{X}$

Audit Bulletin Board

T_A, C*

Tag Chain:

T_A, C^*_{TB}

C* ← Enc(upk’’, *info*)

ConvResponse, upk’’’

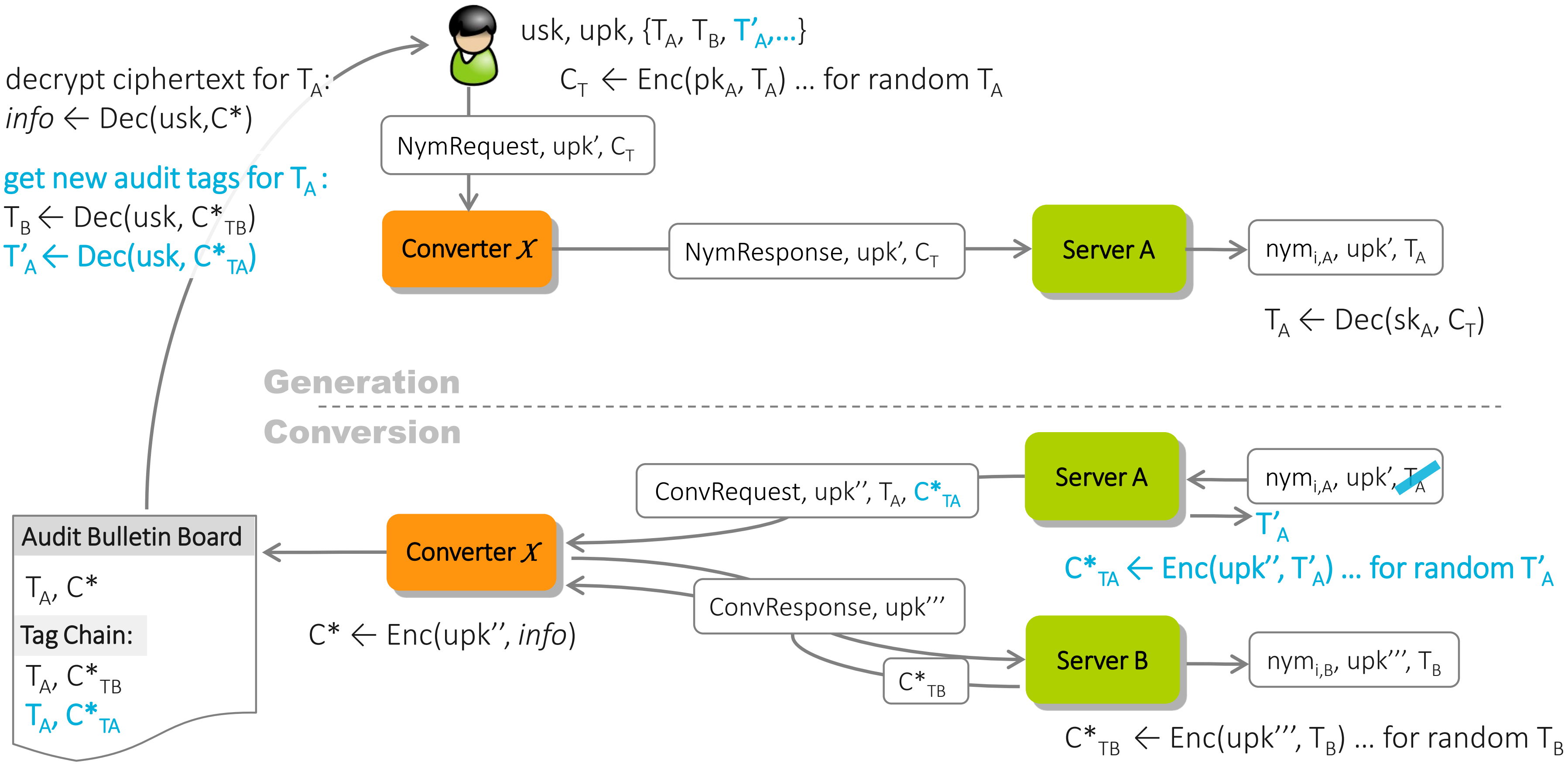
C^*_{TB}

Server B

$nym_{i,B}$, upk’’’, T_B

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

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



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

$C_T \leftarrow \mathrm{Enc}(pk_A, T_A)$ … for random T_A

decrypt ciphertext for T_A:
info ← Dec(usk,C*)

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

NymRequest, upk', C_T

Converter $\mathcal{X}$

NymResponse, upk', C_T

Server A

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

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

Generation

Conversion

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

Server A

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

T'_A

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

Audit Bulletin Board

T_A, C*

Tag Chain:

T_A, C^*_{TB}

T_A, C^*_{TA}

Converter $\mathcal{X}$

ConvResponse, upk'''

$C^* \leftarrow \mathrm{Enc}(upk'', \textit{info})$

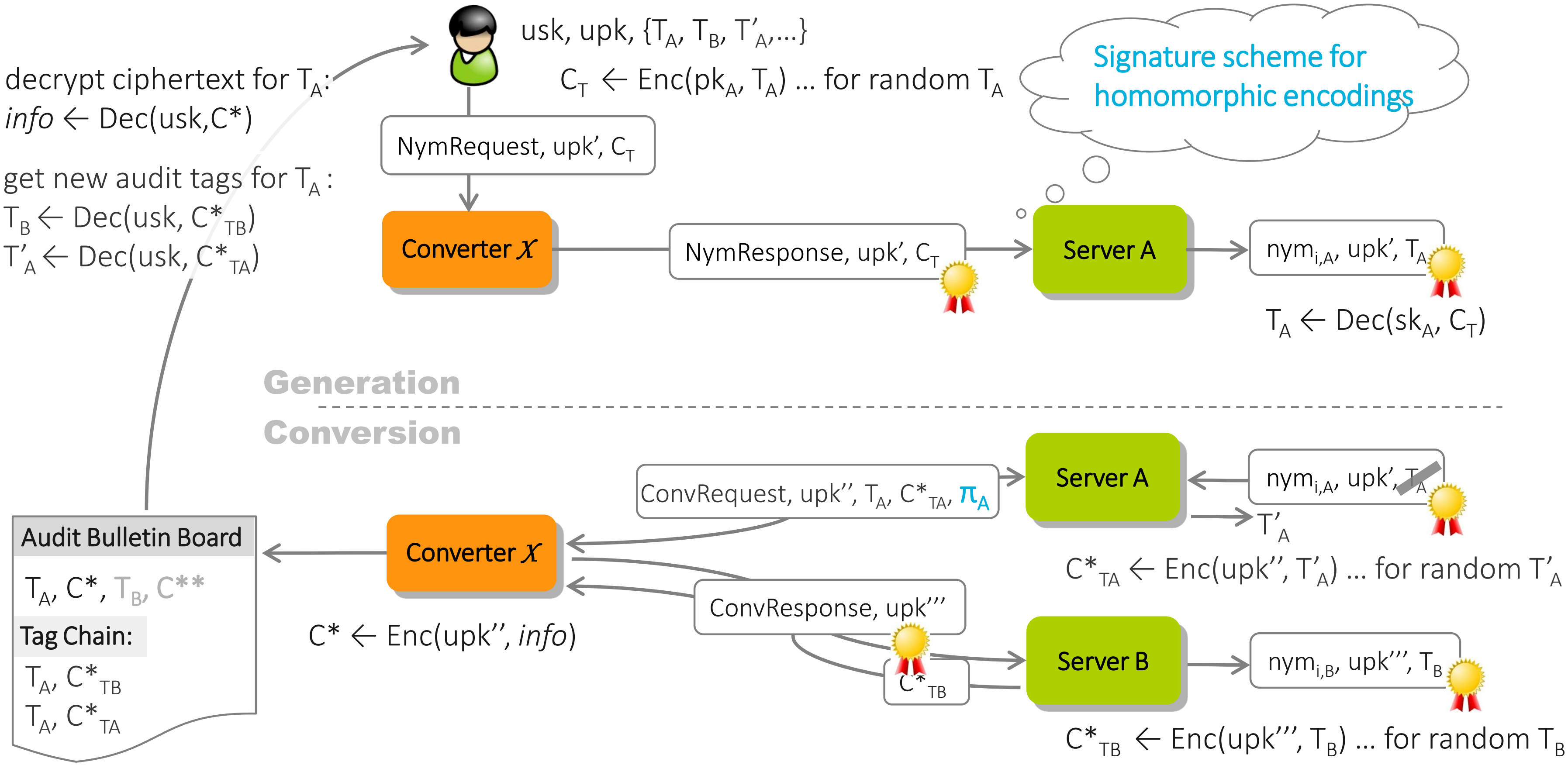
C^*_{TB}

Server B

$nym_{i,B}$, upk''', T_B

$C^*_{TB} \leftarrow \mathrm{Enc}(upk''', T_B)$ … for random 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_B ← Dec(usk, C^*_{TB})
T'_A ← Dec(usk, C^*_{TA})

usk, upk, {T_A, T_B, T'_A,…}

C_T ← Enc(pk_A, T_A) … for random T_A

Signature scheme for homomorphic encodings

NymRequest, upk', C_T

Converter $\mathcal{X}$

NymResponse, upk', C_T

Server A

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

T_A ← Dec(sk_A, C_T)

Generation

Conversion

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

Server A

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

T'_A

C^*_{TA} ← Enc(upk'', T'_A) … for random T'_A

Audit Bulletin Board

T_A, C*, T_B, C**

Tag Chain:

T_A, C^*_{TB}
T_A, C^*_{TA}

Converter $\mathcal{X}$

C* ← Enc(upk'', *info*)

ConvResponse, upk'''

C^*_{TB}

Server B

$nym_{i,B}$, upk''', T_B

C^*_{TB} ← Enc(upk''', T_B) … for random T_B

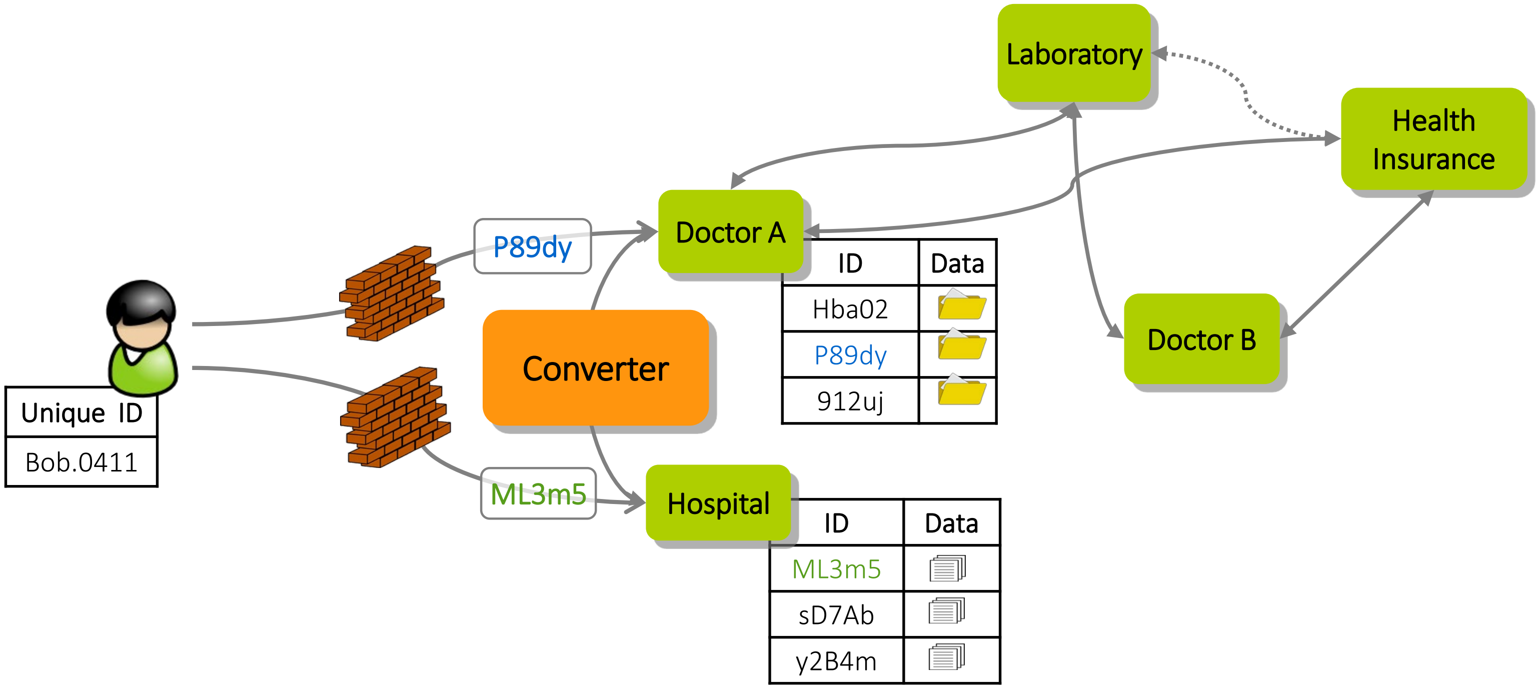
(Un)linkable & Auditable Pseudonyms | Security & Efficiency

- 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)
  - DDH

- Secure against actively corrupt users & servers, and honest-but-curious converter
  - (w/o audits even fully corrupt converter [CL15])

- Concrete instantiation ~50ms computational time per party for conversion

(Un)linkable & Auditable Pseudonyms



ID	Data
Hba02	📁
P89dy	📁
912uj	📁

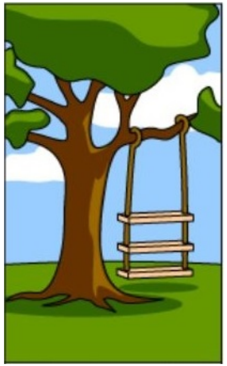
ID	Data
ML3m5	📄
sD7Ab	📄
y2B4m	📄

Controlled data exchange via central entity does not require a TTP !

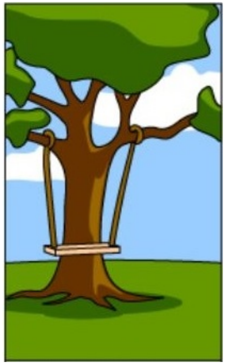
Research &
Consultancy

Understanding Requirements & Constraints

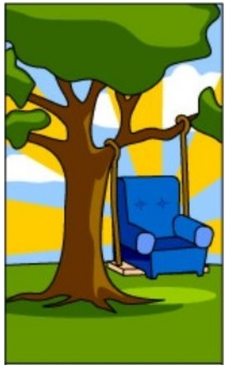
- Challenge: finding common language & clear understanding of problem and constraints



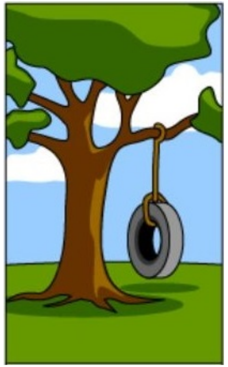
How the customer explained it



How the project leader understood it



How the Business Consultant described it



What the customer really needed

pictures by www.projectcartoon.com

- Often requirements are rather solutions – limits room for innovation
  - E.g., “requirement”: local pseudonyms must be encryption of hash of unique ID

Understanding Requirements & Constraints

- Green field projects: few legacy constraints

  but hard to get exact efficiency requirements

  - CBSS: 800 million requests/year

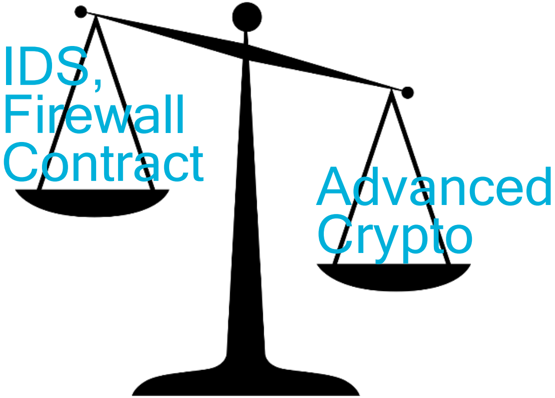
    11million citizens, ~72 requests pp; 2 million/day, 23/sec

  - Similar project: 1 million requests per minute

- „Crypto magic“ needs education and dissemination
  - In particular PETs are counter-intuitive
- Client needs to be comfortable in expressing „crazy“ requirements

How to sell crypto ?

- Selling argument is very different:
  - Research: Privacy is important, TTPs are bad
  - Industry: A TTP is expensive to realize
- Cryptography is costly – investment must pay off
  - Often crypto is not the most cost effective way to protect data
- Trust can be established via contracts & fines
  - Honest-but-curious vs active adversaries
  - Alternative: modest degradation



Where do we stand now

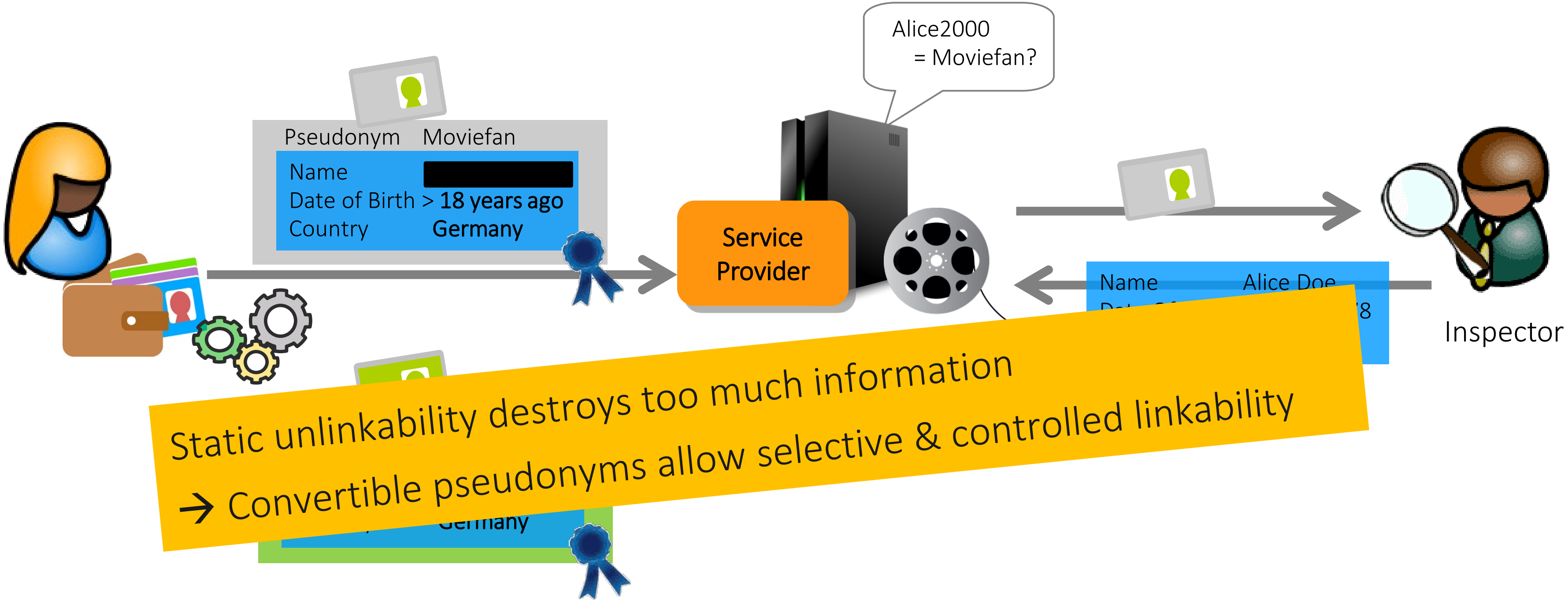
- **In theory:** a lot of interesting open problems! Dream big!
- **In practice:** don‘t dream big ;) But small steps matter as well!

Expectation management!

- Our solution is not used anywhere (yet), but:
  - Changed requirements in call for another nation-wide project
  - Led to a number of simple protocols needed in client projects → 2nd talk
  - Led to nice research papers ☺
  - Improved usability of other privacy-preserving technologies
  - Many open problems and research challenges
  - GDPR creates great practical demand for privacy-preserving mechanisms
    - Data minimisation, consent enforcement, auditability, ...

Anonymous Credentials / Group Signatures / DAA, ...

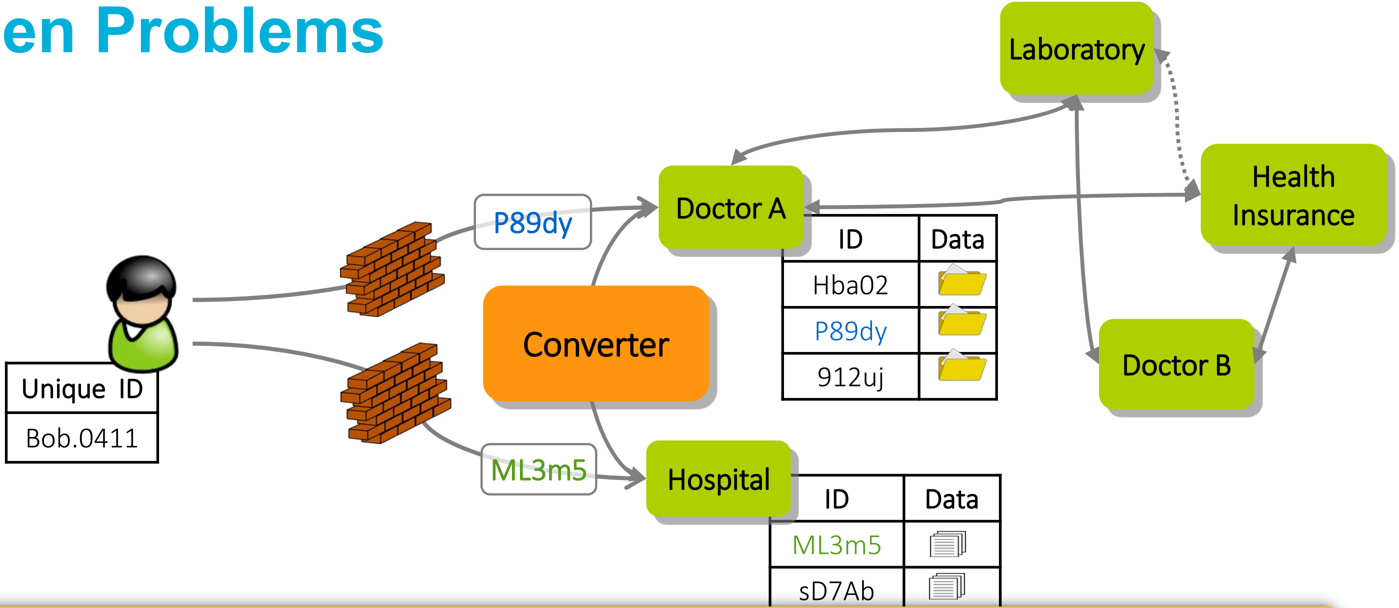
- Privacy-preserving authentication/signatures
  - Selective disclosure & unlinkable authentication
  - User-controlled linkability and/or opening authority



Static unlinkability destroys too much information

→ Convertible pseudonyms allow selective & controlled linkability

Open Problems



Unique ID
Bob.0411

ID	Data
Hba02	📁
P89dy	📁
912uj	📁

ID	Data
ML3m5	📄
sD7Ab	📄

- More fine-grained access control: user-specific policies
- Fair remuneration: users receive rewards for sharing of data
- Full system solution: ensure that data is not identifying either, yet all functionality is preserved
- …

Thanks! Questions?

anj@zurich.ibm.com