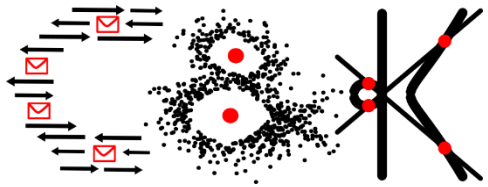
Fully-Featured Anonymous Credentials with Reputation System

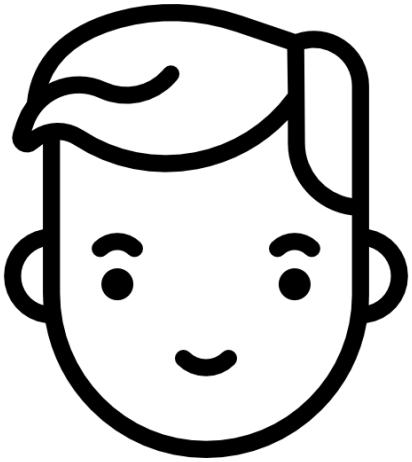
Kai Bemmann, Johannes Blömer, Jan Bobolz,
Henrik Bröcher, Denis Diemert, Fabian Eidens,
Lukas Eilers, Jan Haltermann, Jakob Juhnke,
Burhan Otour, Laurens Porzenheim, Simon Pukrop,
Erik Schilling, Michael Schlichtig, Marcel Stienemeier



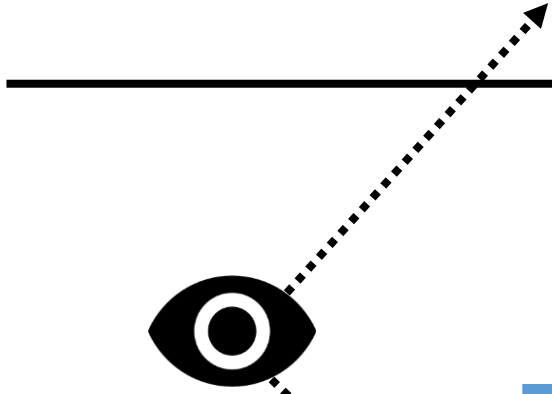
SFB 901
ON - THE - FLY COMPUTING

PADERBORN UNIVERSITY

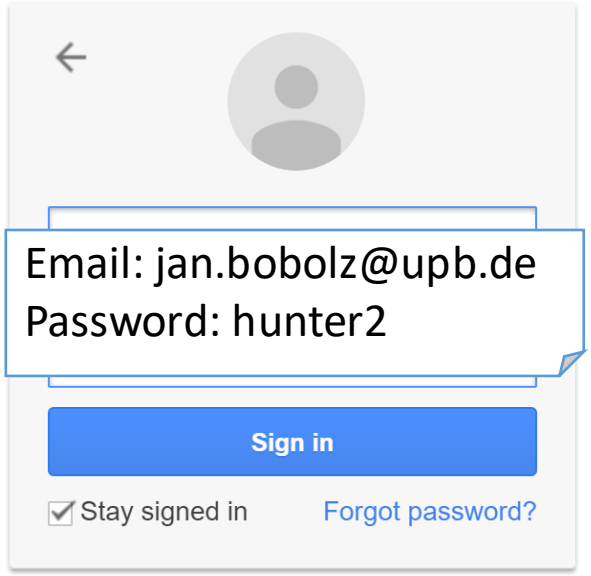
Access control: current state



User

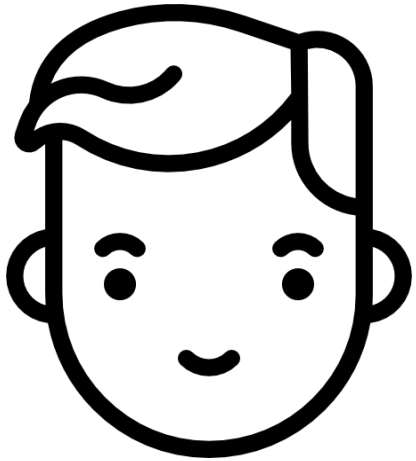


Service Provider



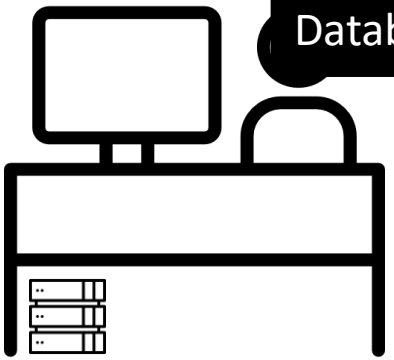
User	Password (*hashed*)
john@doe.com	password123
test@example.com	Lsrzr/?9s.yru3
jan.bobolz@upb.de	hunter2
h4xx0r@gmail.com	w1tn3ssh1d1ng
⋮	⋮

The problem(s)

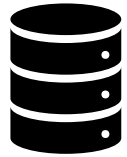


jan.bobolz@upb.de

jan.bobolz@upb.de



Database of payment data



jan.bobolz@upb.de

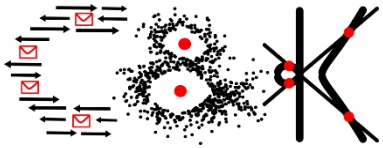
Database of movie tastes

The fix:

- Get rid of identifying information during authentication.
- Instead: Enforce access control through **anonymous credentials**.

CLARC

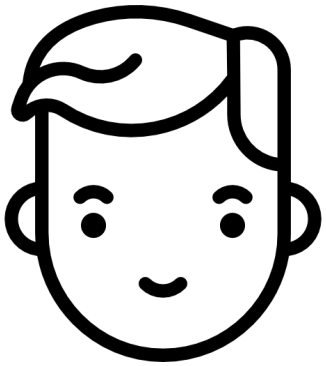
Cryptographic Library
for Anonymous Reputation and Credentials



SFB 901
ON - THE - FLY COMPUTING

PADERBORN UNIVERSITY

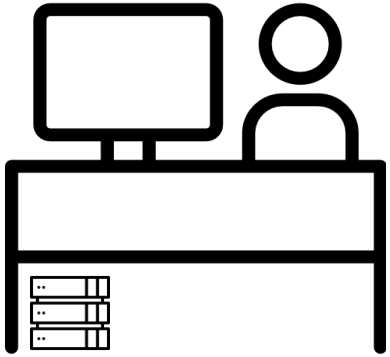
Overview of CLARC



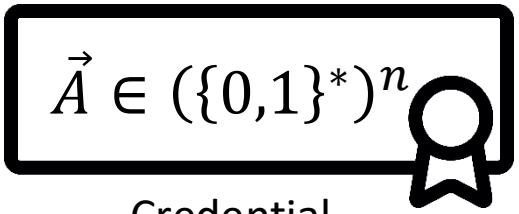
User

Pseudonym 1: request credential

Issue credential



Credential Issuer



Credential

Pseudonym 2: proof of credential with attributes fulfilling policy Φ

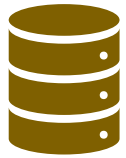
Service Provider

Issue rating token

Rating token

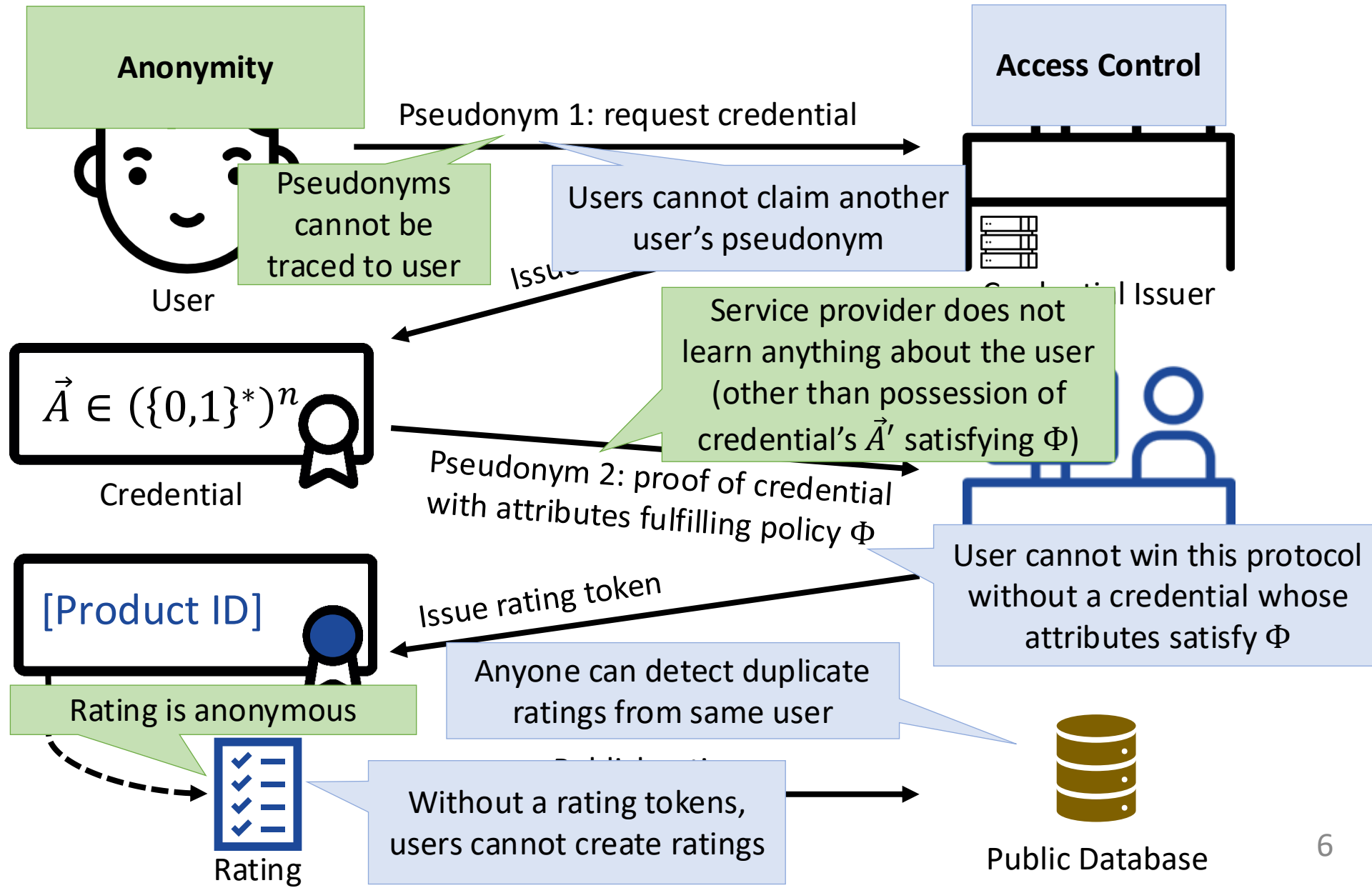
Rating

Publish rating

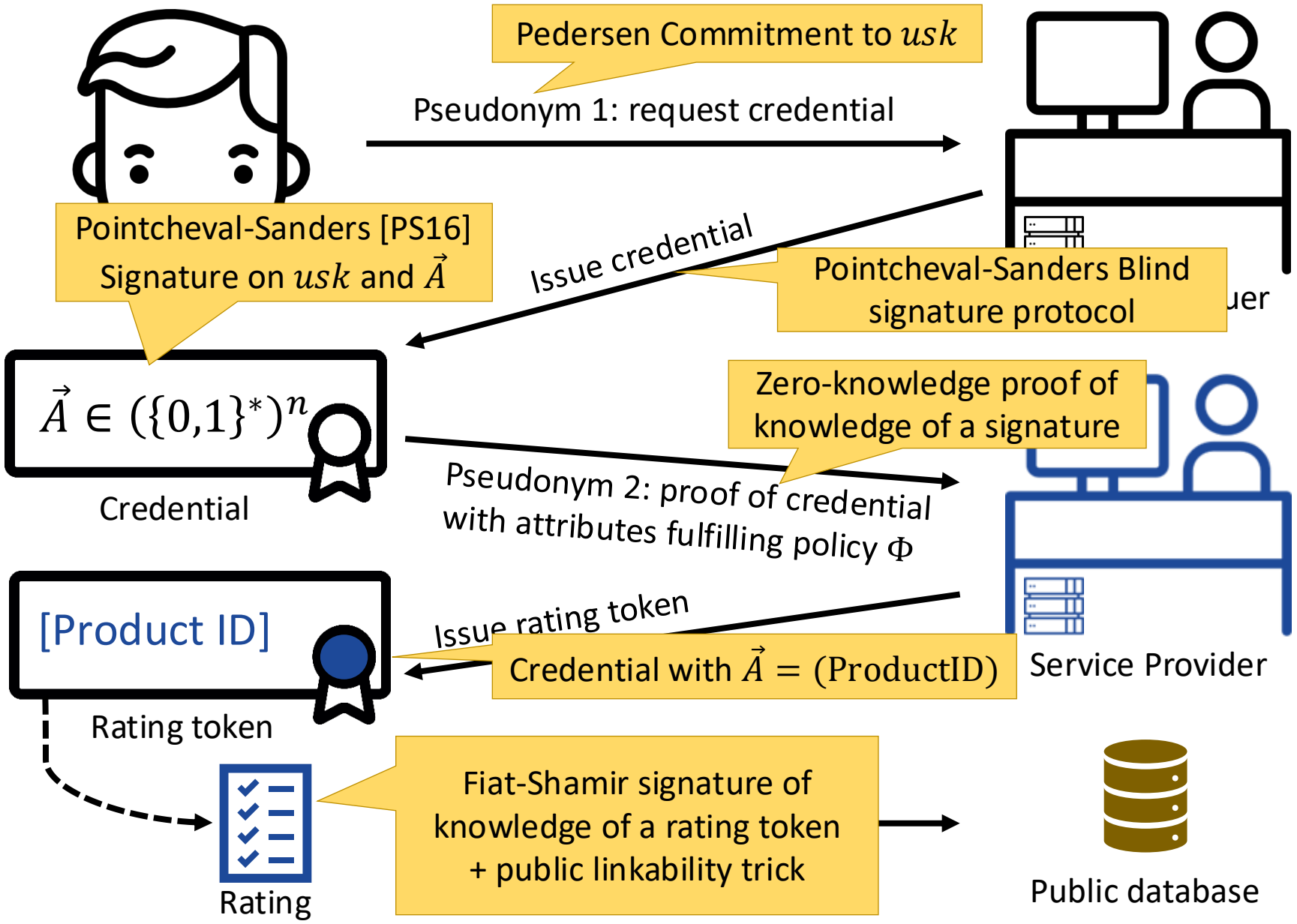


Public Database

Security


Anonymity
Access Control
Pseudonym 1: request credential
Pseudonyms cannot be traced to user
Users cannot claim another user's pseudonym
User
Issuer
$\vec{A} \in (\{0,1\}^*)^n$
Credential
Service provider does not learn anything about the user (other than possession of credential's $\vec{A}'$ satisfying Φ)
Pseudonym 2: proof of credential with attributes fulfilling policy Φ
User cannot win this protocol without a credential whose attributes satisfy Φ
[Product ID]
Issue rating token
Anyone can detect duplicate ratings from same user
Rating is anonymous
Without a rating tokens, users cannot create ratings
Rating
Public Database

Instantiation overview



[PS16] David Pointcheval and Olivier Sanders. Short Randomizable Signatures. In CT-RSA 2016

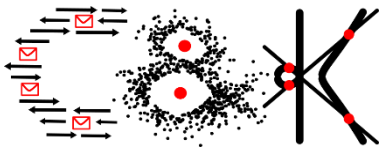
Implementation

PADERBORN UNIVERSITY

- Available at https://github.com/upbcuk/upb.crypto.clarc

- Implemented in Java, based on `upb.crypto` libraries for elliptic curves, hashing, secret sharing, etc.
  - https://github.com/upbcuk

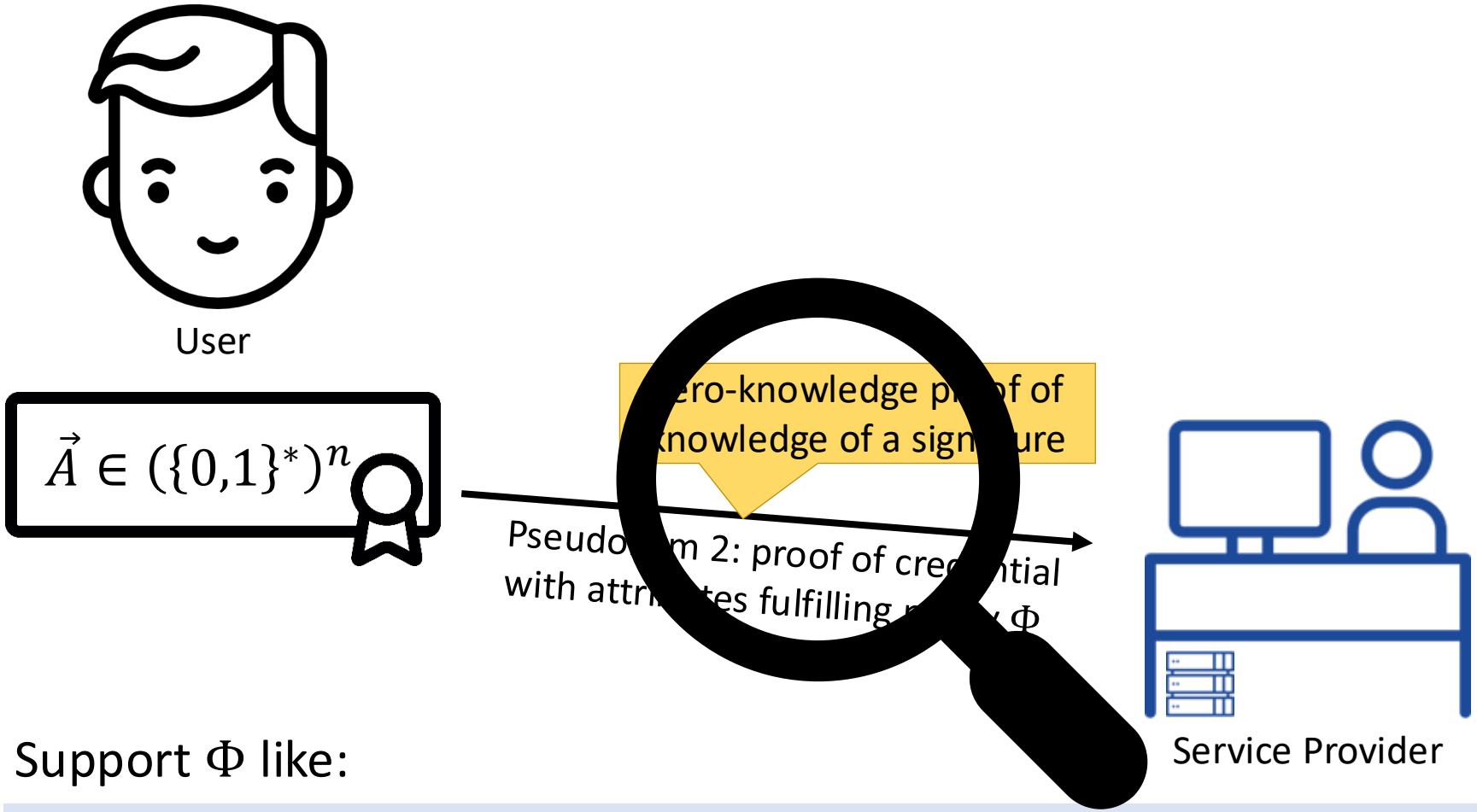
Credential show protocol



SFB 901
ON - THE - FLY COMPUTING

PADERBORN UNIVERSITY

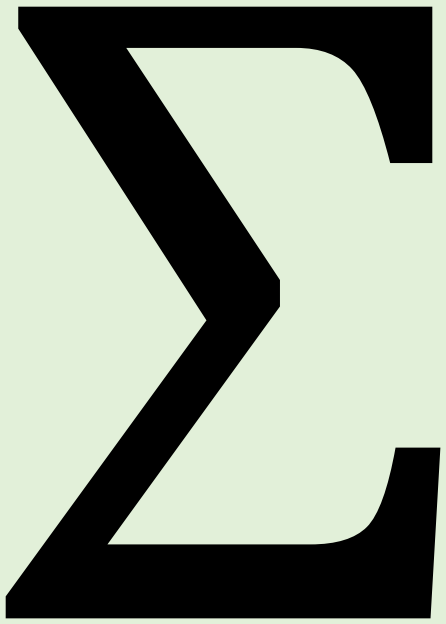
Protocol composition



Support Φ like:

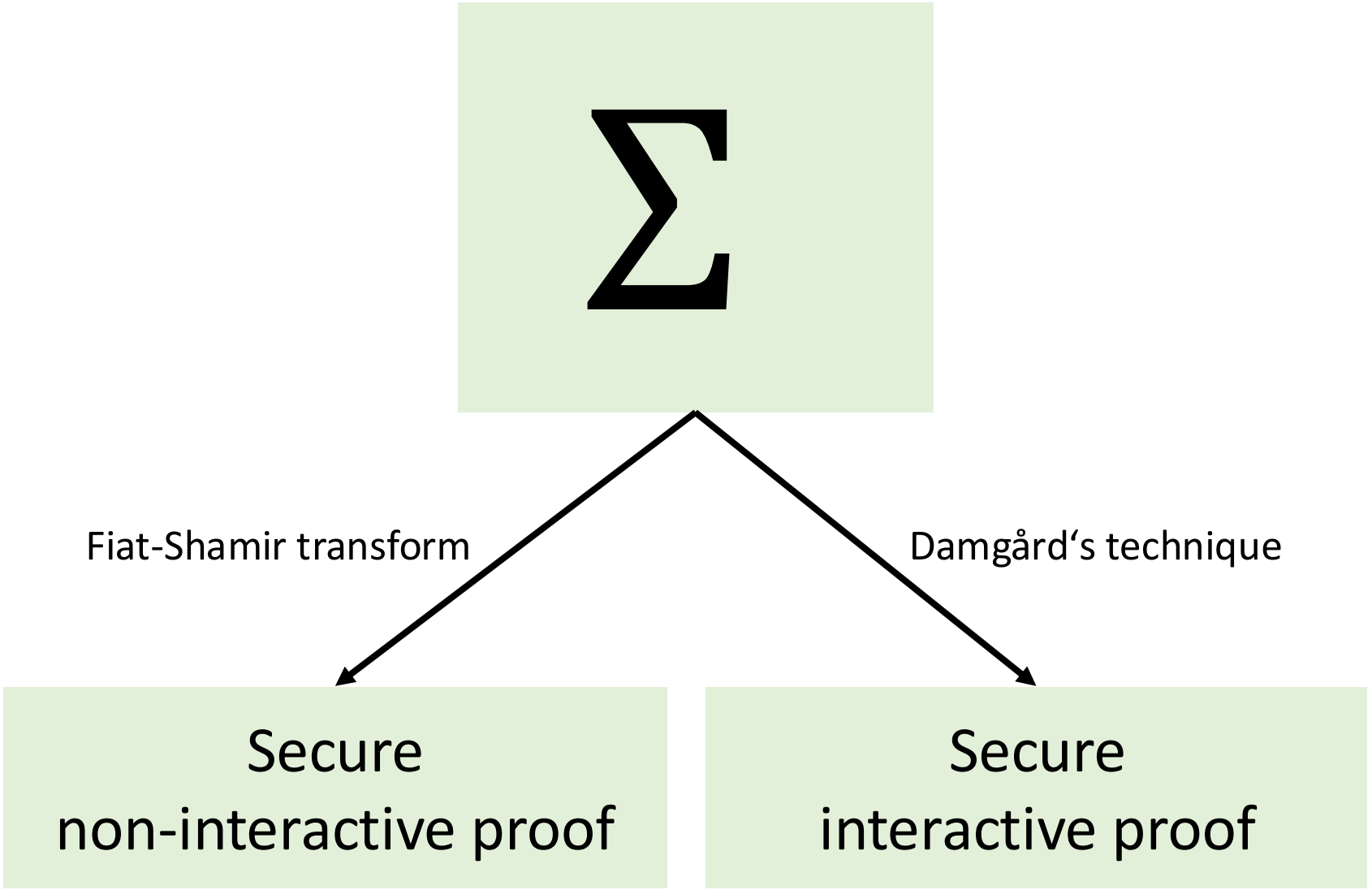
```
citizenship = Germany
∧ (status = student ∨ 0 ≤ age ≤ 17 ∨ age ≥ 66)
∧ residence ∈ {Germany, Austria, Switzerland}
```

Protocol composition



Techniques:
- Sigma protocols
- Schnorr-style protocols
- Accumulator membership proof [Nguyen05]
- Range proofs [Schoenmakers01]
- Proofs of Partial Knowledge [CDS94]

Protocol composition – last step



Performance

PADERBORN UNIVERSITY

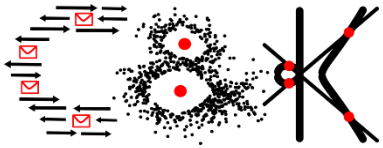
- Test on i7-8650U using bn256 as group/pairing [5]
- Noninteractive proofs of credential possession

Policy	Prove	Verify
`citizenship = Germany`	32 ms	28 ms

`citizenship = Germany`
`∧ (status = student ∨ 0 ≤ age ≤ 17 ∨ age ≥ 66)`
`∧ residence ∈ {Germany, Austria, Switzerland}`
	211 ms	170 ms

[5] mcl: A portable and fast pairing-based cryptography library. https://github.com/herumi/mcl

That‘s it. Questions?



SFB 901
ON - THE - FLY COMPUTING

PADERBORN UNIVERSITY

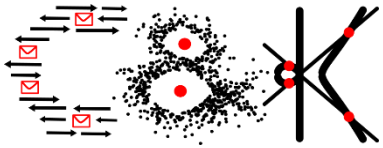
That‘s it. Any questions?

PADERBORN UNIVERSITY

- **CLARC**: Cryptographic Library for Anonymous Reputation and Credentials
- Combines **anonymous authentication** with **anonymous rating** of services
  - Removes the need to identify yourself to authenticate
- Allows proving **complex policies** about credential attributes
  - Compiles Sigma protocol for policy **at runtime**
- **Ratings are anonymous**, but we ensure:
  - Only users who used a service can rate it
  - Users **cannot rate twice**

- **Available** at github.com/upbcuk

Appendix



SFB 901
ON - THE - FLY COMPUTING

PADERBORN UNIVERSITY

Code

PADERBORN UNIVERSITY

github.com/upbcuk

Features that didn‘t make the presentation

PADERBORN UNIVERSITY

- **Anonymity escrow**: verifier can demand that user‘s anonymity can be rescinded by trusted third party
- **Revocation of users:** misbehaving users can be identified and put on a blacklist by a trusted party
- **Revocation of credentials**: a credential can be declared void by its issuer (e.g., if attributes should change)
- **Revocation of ratings**: a rating can be declared invalid by a trusted third party
- **Attribute-based ratings**: the rater can choose to prove possession of additional certified attributes (e.g., “expert reviewer”)
- **Anonymity between multiple credential issuers**: Prove possession of credential by one of multiple issuers.

How to express a policy

PADERBORN UNIVERSITY

```
policy(pp, true)
        .forIssuer(issuerPublicIdentity)
        .attribute(countryDef.getAttributeName()).isEqual("Germany")
        .and()
        .forIssuer(issuerPublicIdentity)
        .attribute(statusDef.getAttributeName()).isEqual("Student")
        .or()
        .attribute(ageDef.getAttributeName()).isInRange(0, 17)
        .or()
        .attribute(ageDef.getAttributeName()).isInRange(66, 200)
        .and()
        .forIssuer(issuerPublicIdentity)
        .attribute(residenceDef.getAttributeName())
          .isInSet("Germany", "Austria", "Switzerland")
        .build();
```

Public Linkability

- Public: g
- Choose $\zeta \leftarrow \mathbb{Z}_p$
- Compute and publish $H(\text{ProductID})^{\zeta+usk}$ and g^{ζ}
  - and prove within the signature of knowledge that those are well-formed
  - Basically ElGamal encryption of $H(\text{ProductID})^{usk}$ (but pairing makes comparison possible)

$\Rightarrow$ Can compute $e(H(\text{ProductID}), g)^{usk}$ and use that for linking.

Johannes Blömer, Jakob Juhnke, and Christina Kolb. 2015. Anonymous and Publicly Linkable Reputation Systems. In FC 2015 (LNCS),

Pointcheval Sanders signatures

- $sk = (x, y_1, \dots, y_n) \in \mathbb{Z}_p^{n+1}$
- $pk = (\tilde{g}, \tilde{g}^x, \tilde{g}^{y_1}, \dots, \tilde{g}^{y_n}) \in \mathbb{G}_2^{n+2}$

- Signatures: $\left(h, h^{x+\sum y_i m_i}\right)$ for random $h \leftarrow \mathbb{G}_1 \setminus \{1\}$

Protocol composition