A Generic Construction of an Anonymous Reputation System and Instantiations from Lattices

Johannes Blömer, Jan Bobolz, and Laurens Porzenheim

Asiacrypt 2023 - December 5 2023

PADERBORN UNIVERSITY

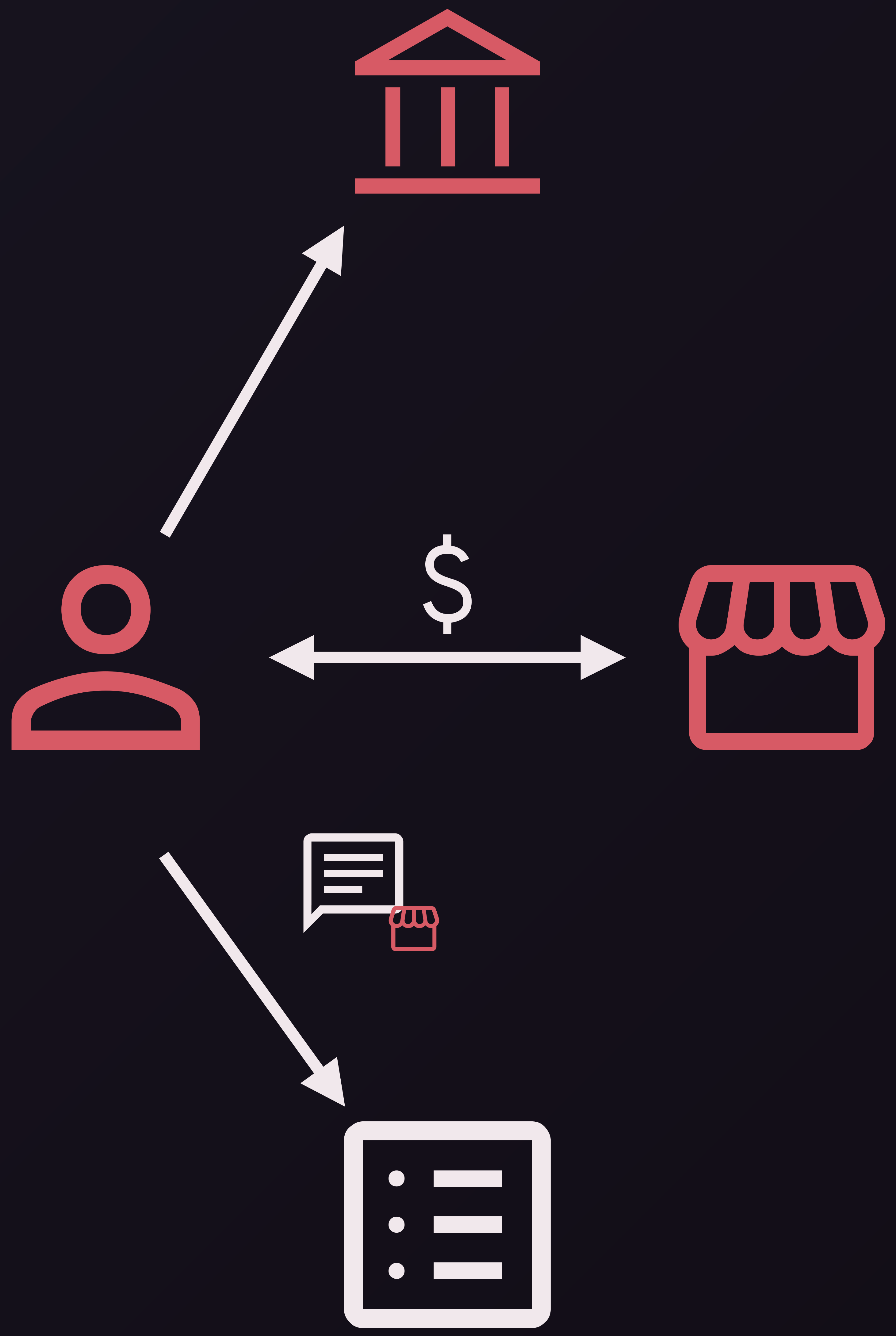
What is an anonymous reputation system?

Reputation Systems

- System where users can rate sellers

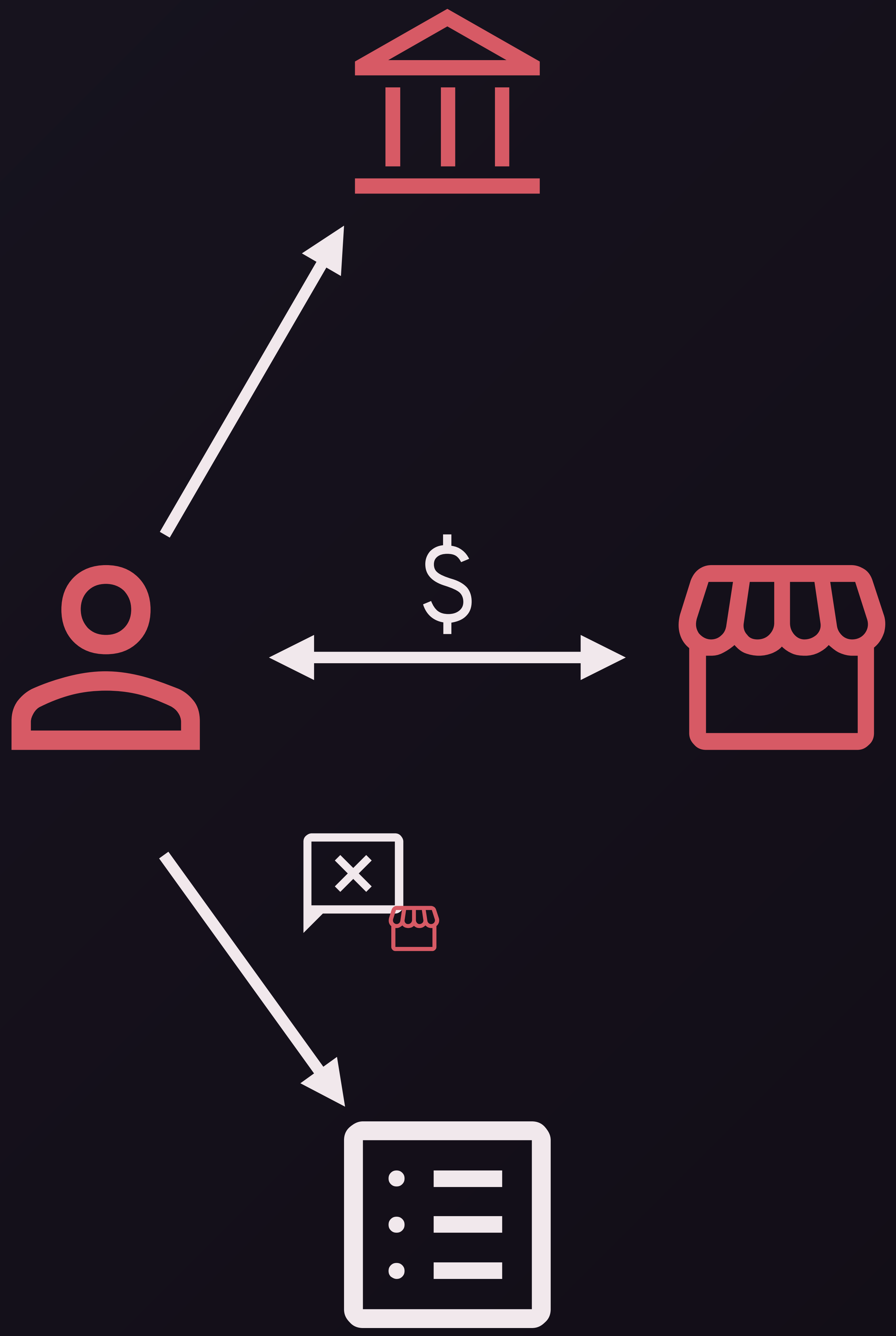
Reputation Systems

- System where users can rate sellers
- Currently centrally handled (e.g. Amazon)



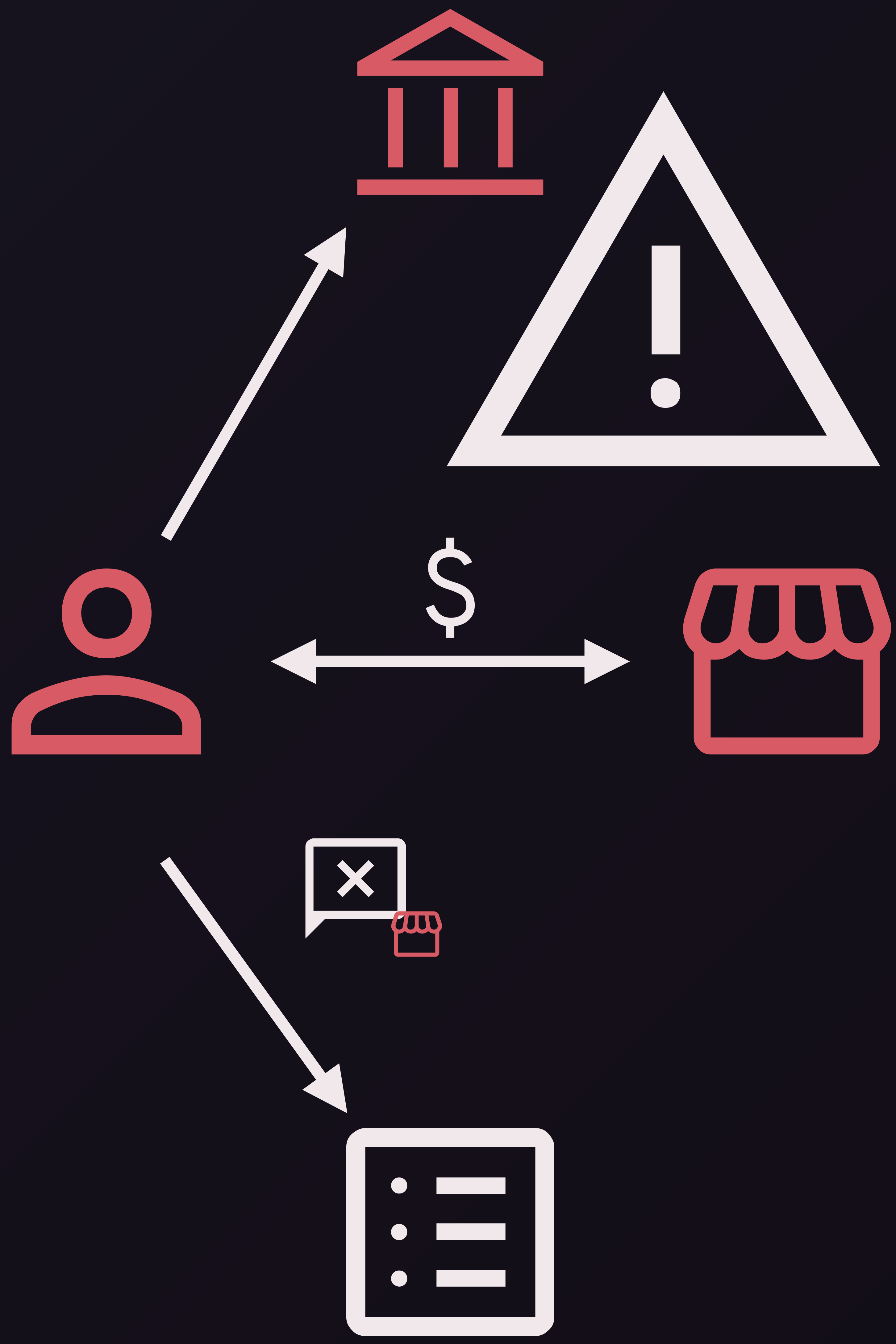
Reputation Systems

- System where users can rate sellers
- Currently centrally handled (e.g. Amazon)



Reputation Systems

- System where users can rate sellers
- Currently centrally handled (e.g. Amazon)
- Users can be retaliated against



Reputation Systems

- System where users can rate sellers
- Currently centrally handled (e.g. Amazon)
- Users can be retaliated against

PADERBORN UNIVERSITY

Reputation Systems

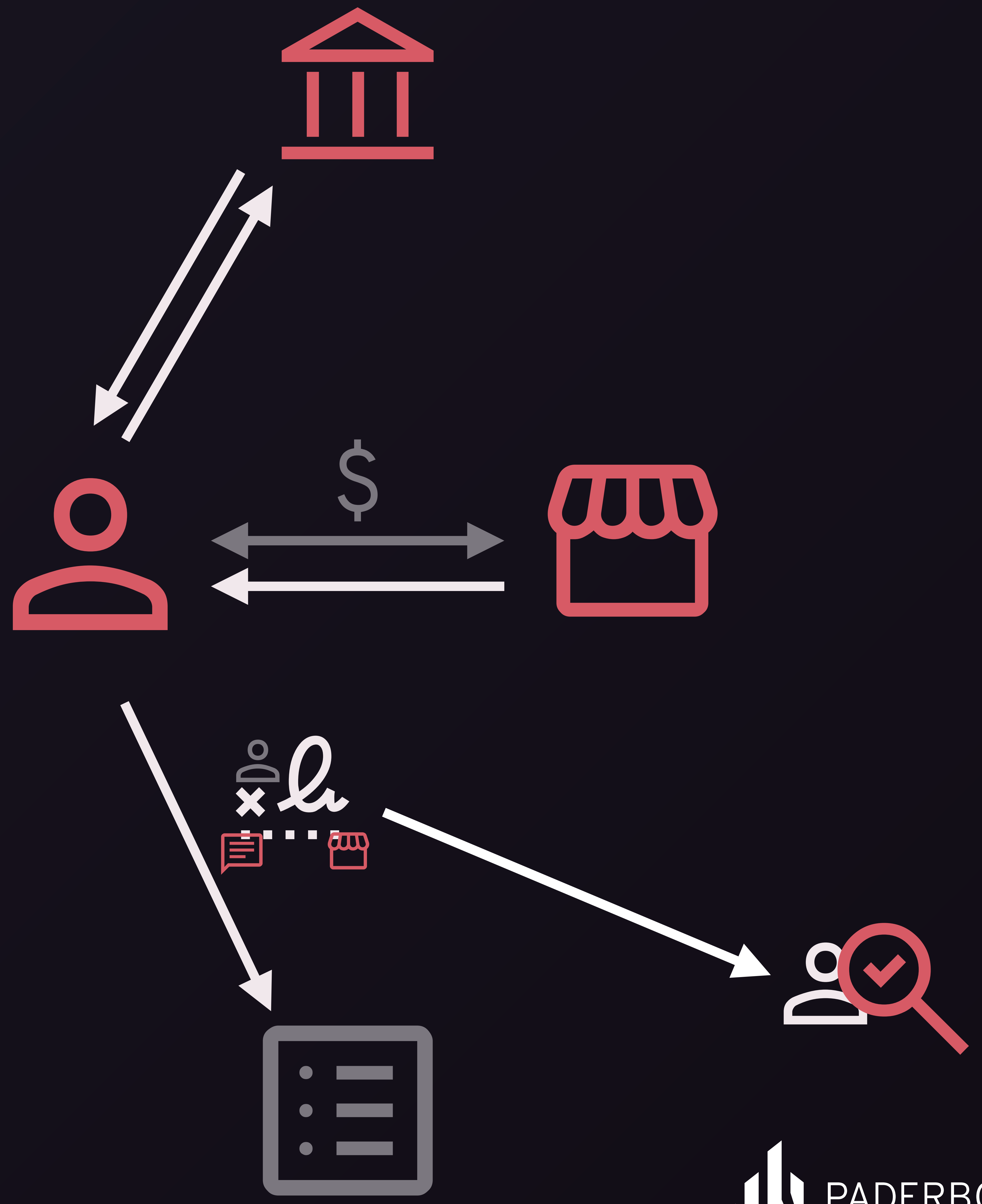
- System where users can rate sellers
- Currently centrally handled (e.g. Amazon)
- Users can be retaliated against

→ **Need to protect anonymity of users**

PADERBORN UNIVERSITY

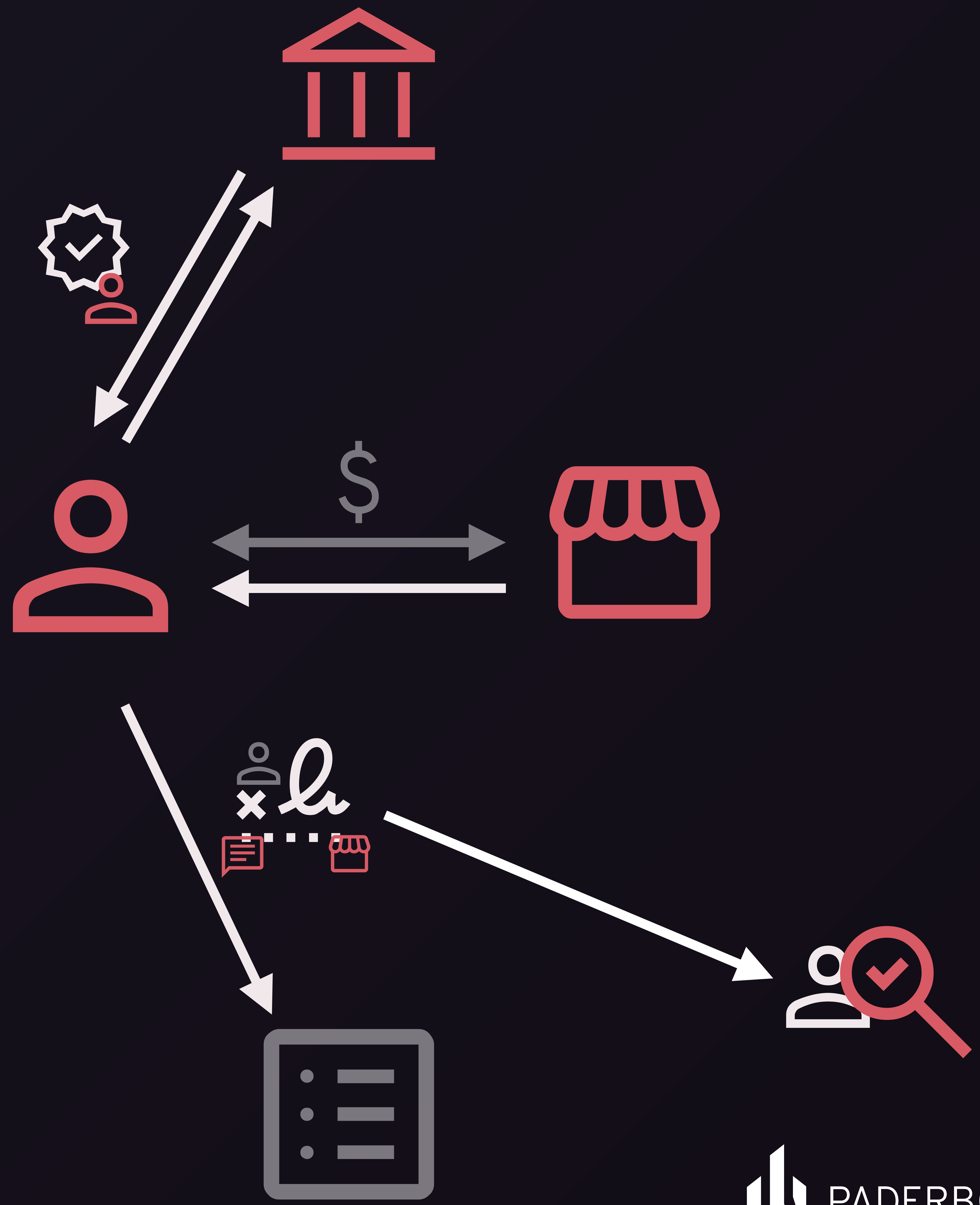
Security Goals

- Users are anonymous



Security Goals

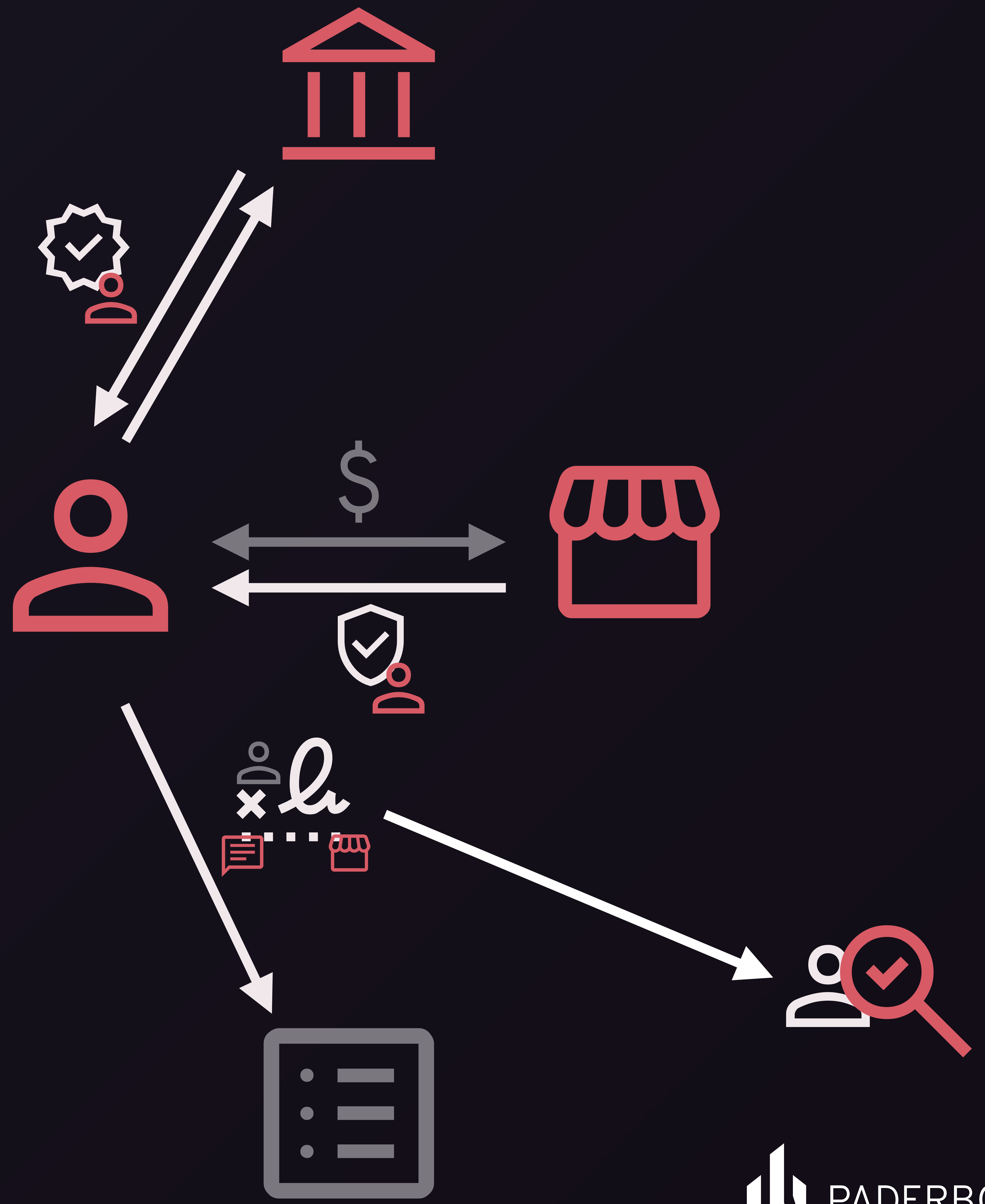
- Users are anonymous
- Only users registered with group manager can rate (join security)



PADERBORN UNIVERSITY

Security Goals

- Users are anonymous
- Only users registered with group manager can rate (join security)
- Only permitted users can rate permitting seller (traceability) once (linkability)



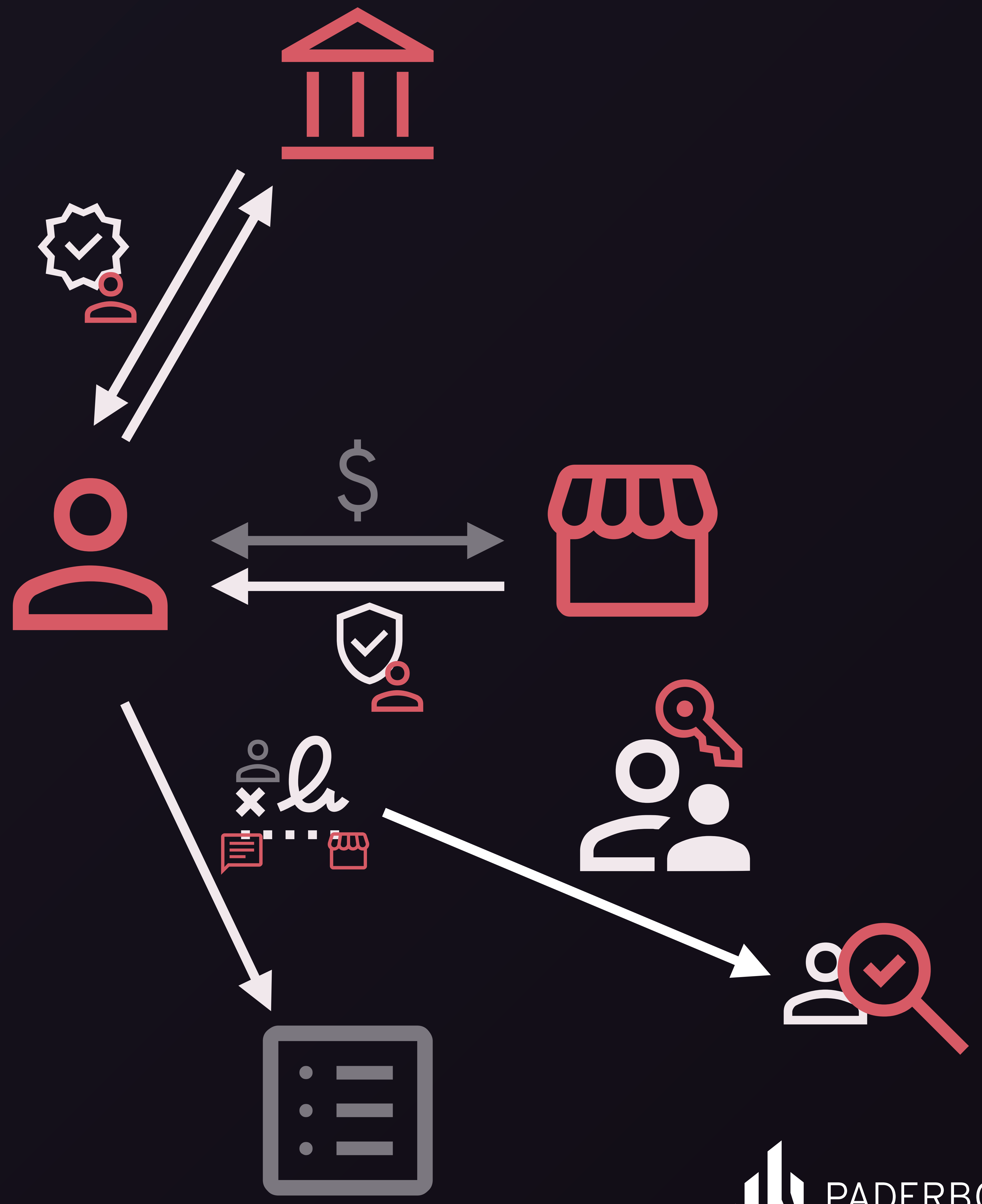
Security Goals

- Users are anonymous
- Only users registered with group manager can rate (join security)
- Only permitted users can rate permitting seller (traceability) once (linkability)
- Users can be correctly tracked by the opener

PADERBORN UNIVERSITY

Security Goals

- Users are anonymous
- Only users registered with group manager can rate (join security)
- Only permitted users can rate permitting seller (traceability) once (linkability)
- Users can be correctly tracked by the opener
- Users cannot be blamed for ratings they did not create (non-frameability)



PADERBORN
UNIVERSITY

How to model security?

PADERBORN UNIVERSITY

Too Many Oracles

- Security models of [BJK15] and [EKS18] exist
- Many oracles
- Many sets to keep track of e.g. corrupted users
- Can be hard to read

PADERBORN UNIVERSITY

Too Many Oracles

- Security models of [BJK15] and [EKS18] exist
- Many oracles
- Many sets to keep track of e.g. corrupted users
- Can be hard to read

→ New security model:

- uses maximally corrupted parties
- same security notions as [BJK15][EKS18]

PADERBORN UNIVERSITY

Anonymity

Goal: adversary does not know which user created a rating

- Two honest users, honest opener, rest is corrupted
- Similar to IND-CPA, distinguishing between users
- Adversary still has some oracles to interact with honest users, opener

Join Security

- Goal: adversary is not able to create ratings that open to non-registered users
- Group manager is honest, opener is honest but curious, rest is corrupted
- Similar to EUF-CMA

Traceability

- Goal: adversary is not able to create ratings that open to non-permitted users
- Seller is honest, opener is honest but curious, rest is corrupted
- Similar to EUF-CMA

Non-Frameability

- Goal: adversary cannot create rating that opens to honest user or links to signature of honest user
- One honest user, opener is honest but curious, rest is corrupted

Linkability

- Goal: linking and opening are consistent
- Opener is honest but curious

Generic Construction

Building Blocks

Building Blocks

- Similar to group signatures:
  - Signatures
  - Encryption
  - NIZK

Building Blocks

- Similar to group signatures:
  - Signatures
  - Encryption
  - NIZK
- Linking Indistinguishable Tags

Linking Indistinguishable Tags

PADERBORN
UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \to sk$
- $\mathsf{Tag}(sk, \mu) \to t$
- $\mathsf{Vrfy}(sk, \mu, t) \to b$

PADERBORN UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \to sk$
- $\mathsf{Tag}(sk, \mu) \to t$
- $\mathsf{Vrfy}(sk, \mu, t) \to b$
- $f(sk) \to pk$
- $\mathsf{Link}(\mu, t_0, t_1) \to b$

PADERBORN UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \to sk$
- $\mathsf{Tag}(sk, \mu) \to t$
- $\mathsf{Vrfy}(sk, \mu, t) \to b$
- $f(sk) \to pk$
- $\mathsf{Link}(\mu, t_0, t_1) \to b$

Security Properties:

PADERBORN UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \to sk$
- $\mathsf{Tag}(sk, \mu) \to t$
- $\mathsf{Vrfy}(sk, \mu, t) \to b$
- $f(sk) \to pk$
- $\mathsf{Link}(\mu, t_0, t_1) \to b$

Security Properties:

- Unforgeability: Cannot create tag that links to tag of honest user

PADERBORN UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \rightarrow sk$
- $\mathsf{Tag}(sk, \mu) \rightarrow t$
- $\mathsf{Vrfy}(sk, \mu, t) \rightarrow b$
- $f(sk) \rightarrow pk$
- $\mathsf{Link}(\mu, t_0, t_1) \rightarrow b$

Security Properties:

- Unforgeability: Cannot create tag that links to tag of honest user
- Indistinguishability: of which key was used to create a tag

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \rightarrow sk$
- $\mathsf{Tag}(sk, \mu) \rightarrow t$
- $\mathsf{Vrfy}(sk, \mu, t) \rightarrow b$
- $f(sk) \rightarrow pk$
- $\mathsf{Link}(\mu, t_0, t_1) \rightarrow b$

Security Properties:

- Unforgeability: Cannot create tag that links to tag of honest user
- Indistinguishability: of which key was used to create a tag
- Linkability: Cannot create tags with the same key and message that do not link

PADERBORN UNIVERSITY

Linking Indistinguishable Tags

- $\mathsf{Kg}(1^\lambda) \to sk$
- $\mathsf{Tag}(sk, \mu) \to t$
- $\mathsf{Vrfy}(sk, \mu, t) \to b$
- $f(sk) \to pk$
- $\mathsf{Link}(\mu, t_0, t_1) \to b$

Security Properties:

- Unforgeability: Cannot create tag that links to tag of honest user
- Indistinguishability: of which key was used to create a tag
- Linkability: Cannot create tags with the same key and message that do not link
- Invertability: Cannot compute sk from tags

PADERBORN UNIVERSITY

Put Together

PADERBORN UNIVERSITY

Put Together

- User . Kg : Use LIT . Kg $\rightarrow usk, upk = f(usk)$
- Register : Sign upk with group manager secret
- Join : Sign upk with seller secret

Put Together

- User . Kg : Use LIT . Kg $\rightarrow usk, upk = f(usk)$
- Register : Sign upk with group manager secret
- Join : Sign upk with seller secret

- Sign$(usk, \mu, \ldots)$:
  - Encrypt upk twice $\rightarrow c, c'$
  - $t \leftarrow$ Tag(usk, pk_{seller})

$$\text{NIZK}_\mu \left\{ usk, sigs, upk : \begin{array}{r} \text{LIT}.f(usk) = upk, \\ sigs \text{ for } upk, \\ c, c', t \text{ generated} \\ \text{honestly} \end{array} \right\}$$

PADERBORN UNIVERSITY

Put Together

- User . Kg : Use LIT . Kg $\rightarrow usk, upk = f(usk)$
- Register : Sign upk with group manager secret
- Join : Sign upk with seller secret
- Open : Decrypt c
- Link : Use LIT . Link

- Sign$(usk, \mu, \ldots)$:
  - Encrypt upk twice $\rightarrow c, c'$
  - $t \leftarrow$ Tag(usk, pk_{seller})

$$\text{NIZK}_\mu \left\{ usk, sigs, upk : \begin{array}{r} \text{LIT}.f(usk) = upk, \\ sigs \text{ for } upk, \\ c, c', t \text{ generated} \\ \text{honestly} \end{array} \right\}$$

Put Together

- User . Kg : Use LIT . Kg $\to usk, upk = f(usk)$
- Register : Sign upk with group manager secret
- Join : Sign upk with seller secret
- Open : Decrypt c
- Link : Use LIT . Link

- Sign$(usk, \mu, \ldots)$:
  - Encrypt upk twice $\to c, c'$
  - $t \leftarrow$ Tag(usk, pk_{seller})

$$\text{NIZK}_\mu \left\{ usk, sigs, upk : \begin{array}{r} \text{LIT} . f(usk) = upk, \\ sigs \text{ for } upk, \\ c, c', t \text{ generated} \\ \text{honestly} \end{array} \right\}$$

Theorem: If the LIT is secure, the encryption CPA secure, the signature EUF-CMA secure and the NIZK has zero-knowledge, simulation-soundness, and straight-line extractability, the construction is secure.

PADERBORN UNIVERSITY

Instantiation from Lattices

PADERBORN UNIVERSITY

Instantiation from Lattices

- LIT: [EKS18], modified to Module Learning with Errors

PADERBORN UNIVERSITY

Instantiation from Lattices

- LIT: [EKS18], modified to Module Learning with Errors
- Encryption: Primal Regev

PADERBORN UNIVERSITY

Instantiation from Lattices

- LIT: [EKS18], modified to Module Learning with Errors
- Encryption: Primal Regev
- Signature: [BLNS23] or (modified) [DM14], [JRS23]

Instantiation from Lattices

- LIT: [EKS18], modified to Module Learning with Errors
- Encryption: Primal Regev
- Signature: [BLNS23] or (modified) [DM14], [JRS23]
- NIZK: [LNP22] with Katsumata's transform [Kat21]

Instantiation from Lattices

- LIT: [EKS18], modified to Module Learning with Errors
- Encryption: Primal Regev
- Signature: [BLNS23] or (modified) [DM14], [JRS23]
- NIZK: [LNP22] with Katsumata's transform [Kat21]

Lemma: If Module Learning with Errors and Module Short Integer Solution are hard, the instantiation is secure in the random oracle model.

PADERBORN UNIVERSITY

Contributions

- New security model using maximally corrupted parties
  - Stronger than [EKS18] in some parts, weaker because more trust in opener required, no revocation
- First generic construction
- Lattice-based instantiation in random oracle model
  - More efficient than [EKS18]
- Pairing-based instantiation in random oracle model

Open Questions

- Updates of ratings
- Revocation
- More efficient direct constructions
- (Generic) construction with weaker requirement than straight-line extractabily for the NIZK

Thank you for the attention!

Sources:

- [BJK15]: Johannes Blömer, Jakob Juhnke, and Christina Kolb. “Anonymous and publicly linkable reputation systems.”
- [BLNS23]: Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Alessandro Sorniotti. “A Framework for Practical Anonymous Credentials from Lattices.”
- [DM14]: Léo Ducas, and Daniele Micciancio. “Improved short lattice signatures in the standard model.”
- [JRS23]: Corentin Jeudy, Adeline Roux-Langlois, and Olivier Sanders. “Lattice signature with efficient protocols, application to anonymous credentials.”
- [EKS18]: Ali El Kaafarani, Shuichi Katsumata, and Ravital Solomon. "Anonymous reputation systems achieving full dynamicity from lattices.”
- [Kat21]: Shuichi Katsumata. "A new simple technique to bootstrap various lattice zero-knowledge proofs to QROM secure NIZKs."

PADERBORN UNIVERSITY

LIT Construction

- Kg(1^λ) : $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \hat{\mathbf{e}} \leftarrow \chi^m$
- Tag(sk, μ) : $\mathbf{A} \leftarrow \mathscr{RO}(\mu), \mathbf{e} \leftarrow \chi^m, \mathbf{t} = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$
- Vrfy(sk, μ, t) : Recompute $\mathbf{t}$
- $f(sk)$: $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, pk = \mathbf{s}^t\mathbf{A} + \hat{\mathbf{e}}^t$
- Link(μ, t_0, t_1) : Check $\|\mathbf{t}_0 - \mathbf{t}_1\| \leq \beta$

PADERBORN UNIVERSITY

Syntax Model

- $\mathsf{Setup}(1^n)$: *The ppt algorithm outputs some public parameters* pp. *We implicitly assume that all algorithms have* pp *as additional input.*
- $\mathsf{KeyGen}_M(1^n)$: *The ppt algorithm outputs a pair of group manager secret and public key* $(\mathsf{gmsk}, \mathsf{gmpk})$.
- $\mathsf{KeyGen}_O(1^n)$: *The ppt algorithm outputs a pair of opening secret and public key* $(\mathsf{osk}, \mathsf{opk})$.
- $\mathsf{KeyGen}_I(1^n)$: *The ppt algorithm outputs a pair of issuer secret and public key* $(\mathsf{isk}, \mathsf{ipk})$.
- $\mathsf{KeyGen}_U(1^n)$: *The ppt algorithm outputs a pair of user secret and public key* $(\mathsf{usk}, \mathsf{upk})$.
- $\mathsf{Join}(\mathsf{gmpk}, \mathsf{usk}), \mathsf{Register}(\mathsf{gmsk}, \mathsf{upk})$: *At the end of their interaction of these interactive ppt algorithms,* Join *outputs a registration token* ρ.
- $\mathsf{Request}(\mathsf{gmpk}, \mathsf{ipk}, \mathsf{usk}, \rho), \mathsf{Issue}(\mathsf{gmpk}, \mathsf{isk}, \mathsf{upk})$: *At the end of the interaction of these interactive ppt algorithms,* $\mathsf{Request}$ *outputs a rating token* τ.
- $\mathsf{Sign}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{usk}, \rho, \tau, \mathsf{rtng})$: *The ppt algorithm outputs a signature* σ.
- $\mathsf{Vrfy}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{rtng}, \sigma)$. *The ppt algorithm outputs a bit* b.
- $\mathsf{Open}(\mathsf{gmpk}, \mathsf{osk}, \mathsf{ipk}, \mathsf{rtng}, \sigma)$: *The ppt algorithm outputs some* upk.
- $\mathsf{Link}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, (\mathsf{rtng}', \sigma'), (\mathsf{rtng}'', \sigma''))$: *The ppt algorithm outputs a bit* b.

Non-Frameability

$$\mathsf{NFrame}_{\Pi,\mathcal{A}}(n)$$

$$\mathsf{pp} \leftarrow \mathsf{Setup}(1^n)$$
$$\mathcal{Q} = \emptyset$$
$$(\mathsf{osk}, \mathsf{opk}) \leftarrow \mathsf{KeyGen}_O(1^n)$$
$$\mathsf{gmpk} \leftarrow \mathcal{A}(\mathsf{osk})$$
$$(\mathsf{usk}_0, \mathsf{upk}_0) \leftarrow \mathsf{KeyGen}_U(1^n)$$
$$\rho_0 \leftarrow \mathsf{Join}(\mathsf{gmpk}, \mathsf{usk}_0) \leftrightarrow \mathcal{A}(\mathsf{upk}_0)$$
$$(\mathsf{ipk}, \mathsf{rtng}, \sigma) \leftarrow \mathcal{A}^{\mathsf{Req}(\mathsf{gmpk},\cdot,0),\mathsf{SigO}(\mathsf{gmpk},\mathsf{opk},\cdot,0,\cdot)}()$$
$$\mathsf{upk} \leftarrow \mathsf{Open}(\mathsf{gmpk}, \mathsf{osk}, \mathsf{ipk}, \mathsf{rtng}, \sigma)$$
$$\text{If } \mathsf{Vrfy}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{rtng}, \sigma) = 0, \text{ return } 0$$
$$\text{If } (\mathsf{ipk}, \mathsf{rtng}, \cdot) \in \mathcal{Q}, \text{ return } 0$$
$$\text{If } \mathsf{upk} = \mathsf{upk}_0, \text{ return } 1$$
$$\text{If } \exists(\mathsf{ipk}, \mathsf{rtng}', \sigma') \in \mathcal{Q} : \mathsf{Link}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, (\mathsf{rtng}, \sigma), (\mathsf{rtng}', \sigma')) = 1, \text{ return } 1$$

Linkability

$$\mathsf{PLinkable}_{\Pi,\mathcal{A}}(n)$$

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^n)$

$(\mathsf{osk}, \mathsf{opk}) \leftarrow \mathsf{KeyGen}_O(1^n)$

$(\mathsf{gmpk}, \mathsf{ipk}, (\sigma_j, \mathsf{rtng}_j)_{j \in \{0,1\}}) \leftarrow \mathcal{A}(\mathsf{osk})$

If $\exists j \in \{0,1\} : \mathsf{Vrfy}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{rtng}_j, \sigma_j) = 0$, return 0.

If $\mathsf{Open}(\mathsf{gmpk}, \mathsf{osk}, \mathsf{ipk}, \mathsf{rtng}_0, \sigma_0) \neq \mathsf{Open}(\mathsf{gmpk}, \mathsf{osk}, \mathsf{ipk}, \mathsf{rtng}_1, \sigma_1)$, return 0.

If $\mathsf{Link}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, (\mathsf{rtng}_0, \sigma_0), (\mathsf{rtng}_1, \sigma_1)) = 0$, return 1.

Generic Construction

- $\mathsf{Setup}(1^n)$: *Run* $\mathsf{pp} \leftarrow \Pi_{\mathrm{NIZK}}.\mathsf{Setup}(1^n)$.
- $\mathsf{KeyGen}_M(1^n)$: *Run* $(\mathsf{gmsk}, \mathsf{gmpk}) \leftarrow \mathsf{KeyGen}_\Sigma(1^n)$.
- $\mathsf{KeyGen}_O(1^n)$: *Run* $(\mathsf{sk}_{\mathsf{Enc}}, \mathsf{pk}_{\mathsf{Enc}}) \leftarrow \mathsf{KeyGen}_{\mathsf{Enc}}(1^n)$ *and* $(\mathsf{sk}'_{\mathsf{Enc}}, \mathsf{pk}'_{\mathsf{Enc}}) \leftarrow \mathsf{KeyGen}_{\mathsf{Enc}}(1^n)$. *Set* $(\mathsf{osk}, \mathsf{opk}) = (\mathsf{sk}_{\mathsf{Enc}}, (\mathsf{pk}_{\mathsf{Enc}}, \mathsf{pk}'_{\mathsf{Enc}}))$ *and forget* $\mathsf{sk}'_{\mathsf{Enc}}$.
- $\mathsf{KeyGen}_I(1^n)$: *Run* $(\mathsf{isk}, \mathsf{ipk}) \leftarrow \mathsf{KeyGen}_\Sigma(1^n)$.
- $\mathsf{KeyGen}_U(1^n)$: *Choose* $\mathsf{usk} \leftarrow \mathsf{KeyGen}_{\mathsf{LIT}}(1^n)$ *and compute* $\mathsf{upk} = f(\mathsf{usk})$.
- $\mathsf{Join}(\mathsf{gmpk}, \mathsf{usk}), \mathsf{Register}(\mathsf{gmsk}, \mathsf{upk})$: *The group manager signs* $\rho \leftarrow \mathsf{Sign}_\Sigma(\mathsf{gmsk}, \mathsf{upk})$ *and sends* ρ *to the user. If* $\mathsf{Vrfy}_\Sigma(\mathsf{gmpk}, \mathsf{upk}, \rho)$, *the user outputs it.*
- $\mathsf{Request}(\mathsf{gmpk}, \mathsf{ipk}, \mathsf{usk}, \rho), \mathsf{Issue}(\mathsf{gmpk}, \mathsf{isk}, \mathsf{upk})$: *The issuer signs* $\tau \leftarrow \mathsf{Sign}_\Sigma(\mathsf{isk}, \mathsf{upk})$ *and sends* τ *to the user. If* $\mathsf{Vrfy}_\Sigma(\mathsf{ipk}, \mathsf{upk}, \tau)$, *the user outputs it.*
- $\mathsf{Sign}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{usk}, \rho, \tau, \mathsf{rtng})$: *Compute* $c = \mathsf{Enc}(\mathsf{pk}_{\mathsf{Enc}}, \mathsf{upk}; r)$. *Compute* $c' = \mathsf{Enc}(\mathsf{pk}'_{\mathsf{Enc}}, \mathsf{usk}; r')$. *Compute* $l = \mathsf{Tag}(\mathsf{usk}, \mathsf{ipk}; r_t)$. *Output* $\sigma = (c, c', l, \pi)$, *where*

$$\begin{aligned}\pi = \mathrm{NIZK}\{&\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{pk}_{\mathsf{Enc}}, \mathsf{pk}'_{\mathsf{Enc}}, c, c', l;\\ &\mathsf{upk}, \mathsf{usk}, \rho, \tau, r, r' \;; \mathsf{upk} = f(\mathsf{usk}) \wedge\\ &\mathsf{Vrfy}_\Sigma(\mathsf{gmpk}, \mathsf{upk}, \rho) = 1 \wedge\\ &\mathsf{Vrfy}_\Sigma(\mathsf{ipk}, \mathsf{upk}, \tau) = 1 \wedge\\ &c = \mathsf{Enc}(\mathsf{pk}_{\mathsf{Enc}}, \mathsf{upk}; r) \wedge\\ &c' = \mathsf{Enc}(\mathsf{pk}'_{\mathsf{Enc}}, \mathsf{usk}; r') \wedge\\ &\mathsf{Vrfy}_{\mathsf{LIT}}(\mathsf{usk}, \mathsf{ipk}, l) = 1\}(\mathsf{rtng})\end{aligned}$$

- $\mathsf{Vrfy}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, \mathsf{rtng}, \sigma)$: *Verify* π *for the corresponding statement.*
- $\mathsf{Open}(\mathsf{gmpk}, \mathsf{osk}, \mathsf{ipk}, \mathsf{rtng}, \sigma)$: *Verify* π *for the corresponding statement. If* π *is valid, output* $\mathsf{upk} = \mathsf{Dec}(\mathsf{osk}, c)$.
- $\mathsf{Link}(\mathsf{gmpk}, \mathsf{opk}, \mathsf{ipk}, (\mathsf{rtng}', \sigma'), (\mathsf{rtng}'', \sigma''))$: *Verify* π', π'' *for the corresponding statements. If* π', π'' *are valid, output* $\mathsf{Link}_{\mathsf{LIT}}(\mathsf{ipk}, l', l'')$.