Welcome!

The GNU Taler Payment System
Prof. Dr. Christian Grothoff
Age restriction in E-commerce

Problem:

Verification of minimum age requirements in e-commerce.

Common solutions:

1. ID Verification
2. Restricted Accounts
3. Attribute-based
Age restriction in E-commerce

Problem:
Verification of minimum age requirements in e-commerce.

Common solutions:

<table>
<thead>
<tr>
<th>Privacy</th>
<th>1. ID Verification</th>
<th>bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>2. Restricted Accounts</td>
<td>bad</td>
</tr>
<tr>
<td>Privacy</td>
<td>3. Attribute-based</td>
<td>good</td>
</tr>
</tbody>
</table>

Principle of Subsidiarity is violated
Age restriction in E-commerce

Problem:

Verification of minimum age requirements in e-commerce.

Common solutions:

<table>
<thead>
<tr>
<th>Solution</th>
<th>Privacy</th>
<th>Ext. authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ID Verification</td>
<td>bad</td>
<td>required</td>
</tr>
<tr>
<td>2. Restricted Accounts</td>
<td>bad</td>
<td>required</td>
</tr>
<tr>
<td>3. Attribute-based</td>
<td>good</td>
<td>required</td>
</tr>
</tbody>
</table>
Age restriction in E-commerce

Problem:

Verification of minimum age requirements in e-commerce.

Common solutions:

<table>
<thead>
<tr>
<th>Privacy</th>
<th>Ext. authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ID Verification</td>
<td>bad</td>
</tr>
<tr>
<td>2. Restricted Accounts</td>
<td>bad</td>
</tr>
<tr>
<td>3. Attribute-based</td>
<td>good</td>
</tr>
</tbody>
</table>

Principle of Subsidiarity is violated
Principle of Subsidiarity

Functions of government—such as granting and restricting rights—should be performed at the lowest level of authority possible, as long as they can be performed adequately.
Principle of Subsidiarity

Functions of government—such as granting and restricting rights—should be performed at the lowest level of authority possible, as long as they can be performed adequately.

For age-restriction, the lowest level of authority is:

Parents, guardians and caretakers
Our contribution

Design and implementation of an age restriction scheme with the following goals:

1. It ties age restriction to the **ability to pay** (not to ID’s)
2. maintains **anonymity of buyers**
3. maintains **unlinkability of transactions**
4. aligns with **principle of subsidiarity**
5. is **practical and efficient**
Assumption: Checking accounts are under control of eligible adults/guardians.
Age restriction
Assumptions and scenario

- Assumption: Checking accounts are under control of eligible adults/guardians.
- **Guardians commit** to an maximum age
Age restriction
Assumptions and scenario

▶ Assumption: Checking accounts are under control of eligible adults/guardians.
▶ Guardians commit to an maximum age
▶ Minors attest their adequate age
Assumptions and scenario

- **Assumption:** Checking accounts are under control of eligible adults/guardians.
- **Guardians commit** to an maximum age
- **Minors attest** their adequate age
- **Merchants verify** the attestations

Note: Scheme is independent of payment service protocol.
Assumptions and scenario

- Assumption: Checking accounts are under control of eligible adults/guardians.
- Guardians **commit** to an maximum age
- Minors **attest** their adequate age
- Merchants **verify** the attestations
- Minors **derive** age commitments from existing ones

---

Note: Scheme is independent of payment service protocol.
Age restriction
Assumptions and scenario

- Assumption: Checking accounts are under control of eligible adults/guardians.
- **Guardians** commit to an maximum age
- **Minors** attest their adequate age
- **Merchants** verify the attestations
- Minors derive age commitments from existing ones
- **Exchanges** compare the derived age commitments

Note: Scheme is independent of payment service protocol.
Age restriction
Assumptions and scenario

- Assumption: Checking accounts are under control of eligible adults/guardians.
- **Guardians commit** to an maximum age
- **Minors attest** their adequate age
- **Merchants verify** the attestations
- Minors **derive** age commitments from existing ones
- **Exchanges compare** the derived age commitments

Note: Scheme is independent of payment service protocol.
Formal Function Signatures

Searching for functions

Commit
Attest
Verify
Derive
Compare
Formal Function Signatures

Searching for functions with the following signatures

 Commit : \((a, \omega) \mapsto (Q, P)\) \\
 Attest \\
 Verify \\
 Derive \\
 Compare

Mnemonics: \\
\(\mathbb{O} = cOmmits\), \(Q = Q-mitment\) (commitment), \(P = Proofs\),
Formal Function Signatures

Searching for functions with the following signatures

Commit : \( (a, \omega) \mapsto (Q, P) \) \( \mathbb{N}_M \times \Omega \rightarrow \mathbb{O} \times \mathbb{P} \),

Attest : \( (m, Q, P) \mapsto T \) \( \mathbb{N}_M \times \mathbb{O} \times \mathbb{P} \rightarrow \mathbb{T} \cup \{ \bot \} \),

Verify

Derive

Compare

Mnemonics:
\( \mathbb{O} = cOmm\text{-}mitments, \quad Q = Q\text{-}mitment (\text{commitment}), \quad \mathbb{P} = P\text{roofs}, \quad P = P\text{roof}, \quad \mathbb{T} = aT\text{testations}, \quad T = aT\text{testation}, \)
Formal Function Signatures

Searching for functions with the following signatures

Commit : \((a, \omega) \mapsto (Q, P)\) 
\(N_M \times \Omega \to O \times P,\)

Attest : \((m, Q, P) \mapsto T\) 
\(N_M \times O \times P \to T \cup \{\bot\},\)

Verify : \((m, Q, T) \mapsto b\) 
\(N_M \times O \times T \to \mathbb{Z}_2,\)

Derive

Compare

Mnemonics:
\(O = cOmmits, \ Q = Q\text{-mitment} \ (\text{commitment}), \ P = P\text{roofs}, \ P = P\text{roof}, \)
\(T = aT\text{estations}, \ T = aT\text{estation},\)
Formal Function Signatures

Searching for functions with the following signatures

- **Commit**: 
  \[(a, \omega) \mapsto (Q, P)\]  
  \[\mathbb{N}_M \times \Omega \rightarrow \mathbb{O} \times \mathbb{P},\]

- **Attest**: 
  \[(m, Q, P) \mapsto T\]  
  \[\mathbb{N}_M \times \mathbb{O} \times \mathbb{P} \rightarrow \mathbb{T} \cup \{\bot\},\]

- **Verify**: 
  \[(m, Q, T) \mapsto b\]  
  \[\mathbb{N}_M \times \mathbb{O} \times \mathbb{T} \rightarrow \mathbb{Z}_2,\]

- **Derive**: 
  \[(Q, P, \omega) \mapsto (Q', P', \beta)\]  
  \[\mathbb{O} \times \mathbb{P} \times \Omega \rightarrow \mathbb{O} \times \mathbb{P} \times \mathbb{B},\]

- **Compare**

Mnemonics:
- \(O = \text{commitments}\)
- \(Q = \text{Q-commitment (commitment)}\)
- \(P = \text{Proofs}\)
- \(T = \text{testations}\)
- \(T = \text{aT-testation}\)
- \(B = \text{Blindings}\)
- \(\beta = \text{blinding}\).
Formal Function Signatures

Searching for functions with the following signatures

\[
\begin{align*}
\text{Commit} : & \quad (a, \omega) \mapsto (Q, P) \quad N_M \times \Omega \rightarrow \mathbb{O} \times P, \\
\text{Attest} : & \quad (m, Q, P) \mapsto T \quad N_M \times \mathbb{O} \times P \rightarrow T \cup \{\bot\}, \\
\text{Verify} : & \quad (m, Q, T) \mapsto b \quad N_M \times \mathbb{O} \times T \rightarrow \mathbb{Z}_2, \\
\text{Derive} : & \quad (Q, P, \omega) \mapsto (Q', P', \beta) \quad \mathbb{O} \times P \times \Omega \rightarrow \mathbb{O} \times P \times \mathbb{B}, \\
\text{Compare} : & \quad (Q, Q', \beta) \mapsto b \quad \mathbb{O} \times \mathbb{O} \times \mathbb{B} \rightarrow \mathbb{Z}_2, 
\end{align*}
\]

Mnemonics:
\( \mathbb{O} = \text{commitments}, \quad Q = Q\text{-mitment} \quad \text{(commitment)}, \quad P = \text{proofs}, \quad P = \text{proof}, \quad T = \text{attestations}, \quad T = \text{at testation}, \quad \mathbb{B} = \text{blindings}, \quad \beta = \beta\text{linging}. \)
Formal Function Signatures

Searching for functions with the following signatures

\[\text{Commit} : \quad (a, \omega) \mapsto (Q, P) \quad \mathbb{N}_M \times \Omega \to \mathbb{O} \times \mathbb{P},\]
\[\text{Attest} : \quad (m, Q, P) \mapsto T \quad \mathbb{N}_M \times \mathbb{O} \times \mathbb{P} \to T \cup \{\bot\},\]
\[\text{Verify} : \quad (m, Q, T) \mapsto b \quad \mathbb{N}_M \times \mathbb{O} \times T \to \mathbb{Z}_2,\]
\[\text{Derive} : \quad (Q, P, \omega) \mapsto (Q', P', \beta) \quad \mathbb{O} \times \mathbb{P} \times \Omega \to \mathbb{O} \times \mathbb{P} \times \mathbb{B},\]
\[\text{Compare} : \quad (Q, Q', \beta) \mapsto b \quad \mathbb{O} \times \mathbb{O} \times \mathbb{B} \to \mathbb{Z}_2,\]

with \(\Omega, \mathbb{P}, \mathbb{O}, T, \mathbb{B}\) sufficiently large sets.

Basic and security requirements are defined later.

Mnemonics:
\[\mathbb{O} = \text{commitments}, \quad Q = \text{Q-mitment} \quad (\text{commitment}), \quad \mathbb{P} = \text{Proofs}, \quad \mathbb{P} = \text{Proof},\]
\[T = \text{attestations}, \quad T = \text{aTtestation}, \quad \mathbb{B} = \text{blindings}, \quad \beta = \beta\text{linging}.\]
Age restriction
Naïve scheme

Zero-Knowledge Age Restriction for GNU Taler
Achieving Unlinkability

Simple use of Derive() and Compare() is problematic.
Achieving Unlinkability

Simple use of Derive() and Compare() is problematic.

- Calling Derive() iteratively generates sequence \((Q_0, Q_1, \ldots)\) of commitments.
- Exchange calls Compare\((Q_i, Q_{i+1}, \ldots)\)
Achieving Unlinkability

Simple use of Derive() and Compare() is problematic.

- Calling Derive() iteratively generates sequence \((Q_0, Q_1, \ldots)\) of commitments.
- Exchange calls \(\text{Compare}(Q_i, Q_{i+1}, \ldots)\)

\[\implies \text{Exchange identifies sequence}\]
\[\implies \text{Unlinkability broken}\]
Achieving Unlinkability

Define cut&choose protocol $\text{DeriveCompare}_\kappa$, using $\text{Derive()}$ and $\text{Compare()}$. 

**Sketch:**
1. $C$ derives commitments $(Q_1, \ldots, Q_\kappa)$ from $Q_0$ by calling $\text{Derive()}$ with blindings $(\beta_1, \ldots, \beta_\kappa)$
2. $C$ calculates $h_0 := H(H(Q_1, \beta_1) || \ldots || H(Q_\kappa, \beta_\kappa))$
3. $C$ sends $Q_0$ and $h_0$ to $E$
4. $E$ chooses $\gamma \in \{1, \ldots, \kappa\}$ randomly
5. $C$ reveals $h_\gamma := H(Q_\gamma, \beta_\gamma)$ and all $(Q_i, \beta_i)$, except $(Q_\gamma, \beta_\gamma)$
6. $E$ compares $h_0$ and $H(H(Q_1, \beta_1) || \ldots || h_\gamma || \ldots || H(Q_\kappa, \beta_\kappa))$ and evaluates $\text{Compare}(Q_0, Q_i, \beta_i)$.

Note: Scheme is similar to the refresh protocol in GNU Taler.
Achieving Unlinkability

Define cut&choose protocol \texttt{DeriveCompare}_\kappa, using Derive() and Compare().

Sketch:

1. \textsf{C} derives commitments \((Q_1, \ldots, Q_\kappa)\) from \(Q_0\) by calling Derive() with blindings \((\beta_1, \ldots, \beta_\kappa)\)
2. \textsf{C} calculates \(h_0 := H(H(Q_1, \beta_1)\| \ldots \| H(Q_\kappa, \beta_\kappa))\)
3. \textsf{C} sends \(Q_0\) and \(h_0\) to \textsf{E}
4. \textsf{E} chooses \(\gamma \in \{1, \ldots, \kappa\}\) randomly
5. \textsf{C} reveals \(h_\gamma := H(Q_\gamma, \beta_\gamma)\) and all \((Q_i, \beta_i)\), except \((Q_\gamma, \beta_\gamma)\)
6. \textsf{E} compares \(h_0\) and \(H(H(Q_1, \beta_1)\| \ldots \| h_\gamma \| \ldots \| H(Q_\kappa, \beta_\kappa))\) and evaluates \texttt{Compare}(Q_0, Q_i, \beta_i).

Note: Scheme is similar to the \textit{refresh} protocol in GNU Taler.
Achieving Unlinkability

With DeriveCompare$_\kappa$

- $\mathcal{E}$ learns nothing about $Q_\gamma$,
- trusts outcome with $\frac{\kappa - 1}{\kappa}$ certainty,
- i.e. $C$ has $\frac{1}{\kappa}$ chance to cheat.

Note: Still need Derive and Compare to be defined.
Refined scheme

Commit(a) → C

(\(Q, P_a\)) → \(\mathcal{G}\)

Derive\(\cdot\)\(\text{Comp}_{\kappa}(T_m, Q)\)

C → \(\varepsilon\)

\(\varepsilon\) → \(\mathcal{M}\)

(\(T_m, Q\)) → Verify(m, Q, T_m)

Attest(m, Q, P_a)
Basic Requirements

Candidate functions

(Commit, Attest, Verify, Derive, Compare)

must first meet basic requirements:

- Existence of attestations
- Efficacy of attestations
- Derivability of commitments and attestations
Basic Requirements

Formal Details

Existence of attestations

\[ \forall a \in \mathbb{N}_M : \text{Commit}(a, \omega) =: (Q, P) \implies \text{Attest}(m, Q, P) = \begin{cases} T \in T, & \text{if } m \leq a \\ \bot & \text{otherwise} \end{cases} \]

Efficacy of attestations

\[ \text{Verify}(m, Q, T) = \begin{cases} 1, & \text{if } \exists P \in P : \text{Attest}(m, Q, P) = T \\ 0 & \text{otherwise} \end{cases} \]

\[ \forall n \leq a : \text{Verify}(n, Q, \text{Attest}(n, Q, P)) = 1. \]

etc.
Security Requirements

Candidate functions must also meet security requirements. Those are defined via security games:

- Game: Age disclosure by commitment or attestation
  ⇔ Requirement: Non-disclosure of age

- Game: Forging attestation
  ⇔ Requirement: Unforgeability of minimum age

- Game: Distinguishing derived commitments and attestations
  ⇔ Requirement: Unlinkability of commitments and attestations

Meeting the security requirements means that adversaries can win those games only with negligible advantage.

Adversaries are arbitrary polynomial-time algorithms, acting on all relevant input.
Security Requirements

Simplified Example

Game $G^\text{FA}_\mathcal{A}(\lambda)$—Forging an attest:

1. $(a, \omega) \leftarrow \mathbb{N}_{M-1} \times \Omega$
2. $(Q, P) \leftarrow \text{Commit}(a, \omega)$
3. $(m, T) \leftarrow \mathcal{A}(a, Q, P)$
4. Return 0 if $m \leq a$
5. Return Verify$(m, Q, T)$

Requirement: Unforgeability of minimum age

$$\forall \mathcal{A} \in \mathcal{A}(\mathbb{N}_M \times \mathcal{O} \times P \rightarrow \mathbb{N}_M \times T): \Pr [G^\text{FA}_\mathcal{A}(\lambda) = 1] \leq \epsilon(\lambda)$$
Solution: Instantiation with ECDSA

To Commit to age (group) \( a \in \{1, \ldots, M\} \)
Solution: Instantiation with ECDSA

To Commit to age (group) \( a \in \{1, \ldots, M\} \)

1. Guardian generates ECDSA-keypairs, one per age (group):

\[
\langle (q_1, p_1), \ldots, (q_M, p_M) \rangle
\]
Solution: Instantiation with ECDSA

To Commit to age (group) \( a \in \{1, \ldots, M\} \)

1. Guardian generates ECDSA-keypairs, one per age (group):

\[ \langle (q_1, p_1), \ldots, (q_M, p_M) \rangle \]

2. Guardian then **drops** all private keys \( p_i \) for \( i > a \):

\[ \langle (q_1, p_1), \ldots, (q_a, p_a), (q_{a+1}, \perp), \ldots, (q_M, \perp) \rangle \]

- \( \vec{Q} := (q_1, \ldots, q_M) \) is the *Commitment*,
- \( \vec{P}_a := (p_1, \ldots, p_a, \perp, \ldots, \perp) \) is the *Proof*
Solution: Instantiation with ECDSA

To Commit to age (group) $a \in \{1, \ldots, M\}$

1. Guardian generates ECDSA-keypairs, one per age (group):
   \[
   \langle (q_1, p_1), \ldots, (q_M, p_M) \rangle
   \]

2. Guardian then **drops** all private keys $p_i$ for $i > a$:
   \[
   \langle (q_1, p_1), \ldots, (q_a, p_a), (q_{a+1}, \bot), \ldots, (q_M, \bot) \rangle
   \]
   \[
   \begin{align*}
   &\quad \tilde{Q} := (q_1, \ldots, q_M) \text{ is the } Commitment, \\
   &\quad \tilde{P}_a := (p_1, \ldots, p_a, \bot, \ldots, \bot) \text{ is the } Proof
   \end{align*}
   \]

3. Guardian gives child $\langle \tilde{Q}, \tilde{P}_a \rangle$
Instantiation with ECDSA

Definitions of Attest and Verify

Child has

- ordered public-keys $\vec{Q} = (q_1, \ldots, q_M)$,
- (some) private-keys $\vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp)$. 
Instantiation with ECDSA

Definitions of Attest and Verify

Child has

- ordered public-keys $\vec{Q} = (q_1, \ldots, q_M)$,
- (some) private-keys $\vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp)$.

To Attest a minimum age $m \leq a$:

Sign a message with ECDSA using private key $p_m$
Instantiation with ECDSA

Definitions of Attest and Verify

Child has
- ordered public-keys $\tilde{Q} = (q_1, \ldots, q_M)$,
- (some) private-keys $\tilde{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp)$.

To Attest a minimum age $m \leq a$:
Sign a message with ECDSA using private key $p_m$

Merchant gets
- ordered public-keys $\tilde{Q} = (q_1, \ldots, q_M)$
- Signature $\sigma$
Instantiation with ECDSA

Definitions of Attest and Verify

Child has

- ordered public-keys \( \vec{Q} = (q_1, \ldots, q_M) \),
- (some) private-keys \( \vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp) \).

To Attest a minimum age \( m \leq a \):
Sign a message with ECDSA using private key \( p_m \).

Merchant gets

- ordered public-keys \( \vec{Q} = (q_1, \ldots, q_M) \)
- Signature \( \sigma \)

To Verify a minimum age \( m \):
Verify the ECDSA-Signature \( \sigma \) with public key \( q_m \).
Instantiation with ECDSA
Definitions of Derive and Compare

Child has \( \vec{Q} = (q_1, \ldots, q_M) \) and \( \vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp) \).
Instantiation with ECDSA
Definitions of Derive and Compare

Child has \( \vec{Q} = (q_1, \ldots, q_M) \) and \( \vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp) \).

To Derive new \( \vec{Q}' \) and \( \vec{P}' \): Choose random \( \beta \in \mathbb{Z}_g \) and calculate

\[
\vec{Q}' := (\beta \ast q_1, \ldots, \beta \ast q_M),
\vec{P}' := (\beta p_1, \ldots, \beta p_a, \perp, \ldots, \perp)
\]

Note: \( (\beta p_i) \ast G = \beta \ast (p_i \ast G) = \beta \ast q_i \)

\( \beta \ast q_i \) is scalar multiplication on the elliptic curve.
Instantiation with ECDSA
Definitions of Derive and Compare

Child has $\vec{Q} = (q_1, \ldots, q_M)$ and $\vec{P} = (p_1, \ldots, p_a, \perp, \ldots, \perp)$.

To Derive new $\vec{Q}'$ and $\vec{P}'$: Choose random $\beta \in \mathbb{Z}_g$ and calculate

$$\vec{Q}' := (\beta \ast q_1, \ldots, \beta \ast q_M),$$
$$\vec{P}' := (\beta p_1, \ldots, \beta p_a, \perp, \ldots, \perp)$$

Note: $(\beta p_i) \ast G = \beta \ast (p_i \ast G) = \beta \ast q_i$

$\beta \ast q_i$ is scalar multiplication on the elliptic curve.

Exchange gets $\vec{Q} = (q_1, \ldots, q_M)$, $\vec{Q}' = (q'_1, \ldots, q'_M)$ and $\beta$

To Compare, calculate: $(\beta \ast q_1, \ldots, \beta \ast q_M) \overset{?}{=} (q'_1, \ldots, q'_M)$
Instantiation with ECDSA

Functions (Commit, Attest, Verify, Derive, Compare) as defined in the instantiation with ECDSA

- meet the basic requirements,
- also meet all security requirements.
  Proofs by security reduction, details are in the paper.
GNU Taler
https://www.taler.net

- Protocol suite for online payment services
- Based on Chaum’s blind signatures
- Allows for change and refund (F. Dold)
- Privacy preserving: anonymous and unlinkable payments
GNU Taler

https://www.taler.net

- Protocol suite for online payment services
- Based on Chaum’s blind signatures
- Allows for change and refund (F. Dold)
- Privacy preserving: anonymous and unlinkable payments

- Coins are public-/private key-pairs \((C_p, c_s)\).
- Exchange blindly signs \(\text{FDH}(C_p)\) with denomination key \(d_p\)
- Verification:

\[
1 \overset{?}{=} \text{SigCheck}(\text{FDH}(C_p), D_p, \sigma_p)
\]

\((D_p = \text{public key of denomination and } \sigma_p = \text{signature})\)
Integration with GNU Taler

Binding age restriction to coins

To bind an age commitment $Q$ to a coin $C_p$, instead of signing $\text{FDH}(C_p)$, $\mathcal{E}$ now blindly signs $\text{FDH}(C_p, H(Q))$

Verification of a coin now requires $H(Q)$, too:

$$1 \overset{?}{=} \text{SigCheck}(\text{FDH}(C_p, H(Q)), D_p, \sigma_p)$$
Integration with GNU Taler

Integrated schemes

- Commit(a)
- (Q, P_a)
- withdraw, using FDH(C_p, H(Q))
- deposit + H(Q)
- refresh + DeriveCompare_k
- purchase + (T_m, Q)
- Attest(m, Q, P_a)
- Verify(m, Q, T_m)
Instantiation with Edx25519

Paper also formally defines another signature scheme: Edx25519.

- Scheme already in use in GNUnet,
- based on EdDSA (Bernstein et al.),
- generates compatible signatures and
- allows for key derivation from both, private and public keys, independently.

Current implementation of age restriction in GNU Taler uses Edx25519.
Our solution can in principle be used with any token-based payment scheme

GNU Taler best aligned with our design goals (security, privacy and efficiency)

Subsidiarity requires bank accounts being owned by adults
  - Scheme can be adapted to case where minors have bank accounts
    - Assumption: banks provide minimum age information during bank transactions.
    - Child and Exchange execute a variant of the cut&choose protocol.

Our scheme offers an alternative to identity management systems (IMS)
Related Work

- Current privacy-perserving systems all based on attribute-based credentials (Koning et al., Schanzenbach et al., Camenisch et al., Au et al.)
- Attribute-based approach lacks support:
  - Complex for consumers and retailers
  - Requires trusted third authority

- Other approaches tie age-restriction to ability to pay (”debit cards for kids”)
  - Advantage: mandatory to payment process
  - Not privacy friendly
Conclusion

Age restriction is a technical, ethical and legal challenge. Existing solutions are

▶ without strong protection of privacy or
▶ based on identity management systems (IMS)

Our scheme offers a solution that is

▶ based on subsidiarity
▶ privacy preserving
▶ efficient
▶ an alternative to IMS
Next seminars

**Biel/Bienne**
Quellgasse 21, Aula

25.11.22 Experimental heart rate variability characterization Lars Brockmann, Assistant, Institute for Human Centered Engineering HuCE, BFH-TI

09.12.22 Parylene-based encapsulation technology for wearable or implantable electronic devices Dr. Andreas Hogg, CEO, Coat-X AG, La Chaux-de-Fonds

13.01.23 Care@Home mit technischer Unterstützung Prof. Dr. Sang-II Kim, Professor, Institute for Medical Informatics I4MI, BFH-TI

**Burgdorf/Berthoud**
Pestalozzistrasse 20, E 013

18.11.22 Flexible programming of Industrial Robots for Agile Production environments Laurent Cavazzana, Research scientist, Institute for Intelligent Industrial Systems I3S, BFH-TI

02.12.22 Wie gefährlich ist ein Unfall mit einem Cabriolet? Prof. Raphael Murri, Institutsleiter IEM, Institut für Energie- und Mobilitätsforschung IEM, BFH-TI

16.12.22 Systemtechnologie für die Mikrobearbeitung mit Hochleistungs-UKPLasern Prof. Dr. Beat Neuenschwander, Institutsleiter ALPS, Institute for Applied Laser, Photonics and Surface Technologies ALPS, BFH-TI