Electronic currencies advantages:
1. no need for trusted party
2. privacy of transactions.

But they are havens for criminals:
1. can be used to hide incomes to fiscal authority
2. criminals can do their business anonymously.

We would like to provide a balance between privacy and accountability
All transactions are public.
Privacy due to the use of pseudonyms instead of real identities.

- Alice signature
- Bob signature
- Bob: 5 bitcoins
- Charlie: 10 bitcoins
- Charlie signature
- Charlie signature
- Dan: 20 bitcoins
- Charlie: 5 bitcoins
- #3432 signature
- #5698 signature
- Charlie: 15 bitcoins
- #5698: 20 bitcoins
A declaration protocol

- Where the authority has access to all incomes.
  - This feature is required by most legislations.
  - Avoid that a compromised key signs too much illegal transactions.
- Where the authority can investigate on true identities but these investigations are recorded by custodians.
  - Allows the authority to do targeted investigations on who sends money to who.
  - Prevent mass-surveillance abuses (systematic investigation on all transactions).
Entities

Participants:

- **Payees**: the users that declare the money
  - Need to prove their public key is authentic before starting declarations
  - May be malicious
- **Authority**: it knows the incomes of payees and can perform investigations
  - Malicious but cautious: does only malicious actions that cannot be detected.
- **Custodians**: the link between each transaction and payee is stored by \( n \) of them.
  - They are certified by the authority (like banks, big institutions, non-governmental organizations)
  - they are all malicious but cautious w.r.t. the authority
  - at least one is honest w.r.t. the payee
- **Public logs**:
  - the Bitcoin transactions
  - temporary keys certified by custodians
  - transactions signed by temporary keys
Overview

Time is divided into periods:
- Longer periods allows more privacy for payees
- Shorter periods allows authority to investigate earlier

One instance of 4 phases for each period
- custodian choice phase
- custodian establishment phase
- payment phase
- acknowledgment phase

April 1st, 2015
Establishing temporary identities

A link provides a relation between:

- A public key $k_1$ that the payee certified to the authority to be linked to its true identity $R$.
- A public key $k_{n+1}$ whose corresponding signature is used to sign transactions.

Properties:
1. Links can be revealed when all custodians collaborate.
2. Links cannot be faked or altered surreptitiously.
Custodian establishment phase

The onion is opened by all custodians iteratively. 

\[ H_{i+1} = h(k_i, s_i, H_i) \] 
build a unique identifier of the link.
Custodian establishment phase

The onion is opened by all custodians iteratively. \( H_{i+1} = h(k_i, s_i, H_i) \) build a unique identifier of the link.
Custodian establishment phase

The onion is opened by all custodians iteratively. $H_{i+1} = h(k_i, s_i, H_i)$ build a unique identifier of the link.
Custodian establishment phase

The onion is opened by all custodians iteratively.

\[ H_{i+1} = h(k_i, s_i, H_i) \] build a unique identifier of the link.

\[ R \]

\[ A \]

\[ C_1 \]

\[ A \]

\[ C_2 \]

\[ C_3 \]
Custodian establishment phase

The onion is opened by all custodians iteratively. 

\( H_{i+1} = h(k_i, s_i, H_i) \) build a unique identifier of the link.
Payment phase

To validate a payment, a payee:

- checks that the previous transactions has been declared.
- looks on custodian’s logs for an identity $k_{n+1}$ the payee has issued.
- checks that the corresponding $H_{n+1}$ is the intended one.
- publishes a declaration sign $(sk_{n+1}, [H_{n+1}, a, t])$ where $t$ identifies the transaction and $a$ is a partial amount.

<table>
<thead>
<tr>
<th>$C_3$</th>
<th>$k_4^1$</th>
<th>$H_4^1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_3$</td>
<td>$k_4^2$</td>
<td>$H_4^2$</td>
</tr>
<tr>
<td>$C_3$</td>
<td>$k_4^3$</td>
<td>$H_4^3$</td>
</tr>
<tr>
<td>$C_3$</td>
<td>$k_4^4$</td>
<td>$H_4^4$</td>
</tr>
</tbody>
</table>

#3432 signature | #3443 : 10 bitcoins
#5698 signature | #5698 : 20 bitcoins
Acknowledgment phase

\[ H'_i = h(H'_{i+1}, s_i, m) \] allows to identify the link uniquely.

All declarations for a temporary identity are summed by \( C_n \): here \( 2 + 4 \).
Acknowledgment phase

\[ H'_i = h(H'_{i+1}, s_i, m) \] allows to identify the link uniquely.

\[ R \begin{array}{c} \text{A} \\ k_1 \end{array} \]

\[ \text{C}_1 \begin{array}{c} \text{A} \\ k_1 \text{ H}_0 \end{array} \]

\[ \text{C}_2 \begin{array}{c} \text{C}_1 \\ k_2 \text{ H}_1 \end{array} \)

\[ \text{C}_3 \begin{array}{c} \text{C}_2 \\ k_3 \text{ H}_2 \end{array} \)

\[ \text{C}_3 \begin{array}{c} \text{C}_3 \\ k_3 \text{ 6} \text{ H}'_2, \text{H}_2 \end{array} \)

\[ R \begin{array}{c} \text{R} \\ k_4 \text{ H}_3 \text{ 4} \text{ t}_2 \end{array} \]

\[ R \begin{array}{c} \text{R} \\ k_4 \text{ H}_3 \text{ 2} \text{ t}_1 \end{array} \)

The custodians propagate the declaration to the authority.
Acknowledgment phase

\[ H'_i = h(H'_{i+1}, s_i, m) \] allows to identify the link uniquely.

The custodians propagate the declaration to the authority.
Acknowledgment phase

\[ H'_i = h(H'_{i+1}, s_i, m) \] allows to identify the link uniquely.

The authority learns \( R \) has declared 6 for this period.
Acknowledgment phase

\( H'_i = h(H'_{i+1}, s_i, m) \) allows to identify the link uniquely.

The authority sends a receipt to the payee for verification.
Global view

MSC diagram:

- **Payee (R)**
- **Authority (A)**
- **C₁**
- **Cₙ₋₁**
- **Cₙ**
- **Public log**

**Phase 0 (ph. 0.0):**

- **O**

**Phase 1 (ph. 1.0):**

- **(enc(C₁, [V₀]), O₁)**

**Phase 1 (ph. 1.n):**

- **Vₙ**
- **sign(skₙ₊₁, [T])**
- **enc(Cₙ, [Vₙ₋₁])**

**Phase 2:**

- **Bₙ**

**Phase 3 (ph. 3.0):**

- **sign(skₙ₊₁, [T])**

**Phase 3 (ph. 3.n):**

- **F**
- **B₁**

- **B̃₁**
Final checks

Bad events that may happen:

- The authority receives a complaint from a payee.
  - An investigation is started to figure out whether the payee is lying or a custodian misbehaved.
- The total of all validated transactions for the period is not equal to the total of all incomes.
  - Potentially all transactions of the period have to be investigated but the malicious custodian will be found.
- A custodian never receives an acknowledgment.
  - Look for a faulty custodian.

Once these events has been solved, the authority knows that

- incomes declarations are correct,
- later investigation cannot be deceived.
Investigation property

The worst possible scenario: one custodian does not answer. Either because:

- it is deliberate
- it did not follow the protocol and cannot provide a correct answer.

This situation is unlikely as custodians are chosen from highly reliable entities.

Attacks which are prevented:

- Ability for payees to hide transactions,
- Ability for a malicious custodian to blame someone else,
- Ability for a custodian to choose at the time of investigation which link it provides.
Investigation procedure

Two investigation process:

- \( I_p(B_1) \) starts from the message \( B_1 = \text{sign}(C_1, [k_1, a, H_0, H'_0]) \) associated to the true identity \( R \).
- \( I_t(V_n) \) starts from the message \( V_n = \text{sign}(C_n, [k_{n+1}, H_n]) \) where \( k_{n+1} \) links to transactions.

**Process:** For \( i = 1 \) to \( n \), resp. for \( i = n \) to \( 1 \).

1. Present to \( C_i \) the message \( B_i = \text{sign}(C_i, [k_i, a, H_i, H'_i]) \), resp. \( V_i = \text{sign}(C_i, [k_{i+1}, H_i]) \).
2. \( C_i \) provides to the authority: \( E_i \) and \( B_{i+1} \) resp \( E_i, \tilde{B}_i \) and \( V_{i-1} \).
3. \( C_i \) maintains a log of the number of requests.
4. The authority checks the hashes are the intended ones, the next custodian is found from \( E_i \) resp. \( V_{i-1} \) signature.

At the end \( I_p(B_1) \) resp. \( I_t(V_n) \) either stops as a malicious custodians does not provide the correct messages or the authority get \( V_n \) resp. \( V_1 \) that contains the key used to sign transactions, resp. the key signed by \( R \).
Investigation property

Proposition

For any message $B_1$ received by the authority (and for which it knows the corresponding payee $R$), the authority can start an investigation procedure $I_p(B_1)$.

- If the procedure fails, then the authority can identify one custodian who has been malicious.
- If the procedure succeeds, $I_p(B_1)$ returns inside $V_n$ a temporary identity $k_{n+1}$ such that:
  - the investigation process cannot return another $k'_{n+1}$,
  - $R$ has necessarily been involved in the signature of $k_{n+1}$,
  - an investigation from $k_{n+1}$ can only return $R$.

A similar property holds for investigations $I_t$ starting from $k_{n+1}$. 
Which anonymity do we need?

The authority should not be able to link real identities to temporary identities

Example

Alice declares $T_1$ and Bob $T_2$ or Alice declares $T_2$ and Bob $T_1$ should not be distinguishable

- The authority would lose its reputation by leaving evidences of attacks.
- The authority has no interest in attacking one link and then losing its reputation: it can use the official process.

A successful attack is an attack that succeeds and *then* does not leave any evidence.
We formalize this anonymity property in the applied pi calculus. An attacker can always abort the process once it gets what it wants: Evidences are likely to be absence of the reception of an expected message. In the pi-calculus: the instance is pending and cannot reduce anymore. Pending means:

- has started: not under a bang
- has not finished: not null

**Definition**

A completed extended process is a process whose structure is

\[ \mu \tilde{k} (\|P_1\| \ldots \|P_n\|\{x_1/y_1\} \ldots \{x_m/y_m\}) \].

We denote by \( C(P) \), the fact that \( P \) is a completed process.
Example

Here $c$ and $d$ are public channels, $p$ and $p'$ public constants.

\[
\nu k. (\nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))|\text{in}(c, x); \text{out}(c, \text{dec}(x, k)))|\text{out}(d, \text{enc}(k, p')))
\]

\[
\nu k. (\nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))|\text{in}(c, x); \text{out}(c, \text{dec}(x, k)))|\text{out}(d, \text{enc}(k, p))
\]

\[
\nu k. \text{out}(d, \text{enc}(k, p)) \quad \text{→} \quad \nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))|\text{in}(c, x); \text{out}(c, \text{dec}(x, k))
\]

\[
\frac{\text{in}(c, \text{enc}(k, p))}{\nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))|\text{in}(c, x); \text{out}(c, \text{dec}(x, k))}
\]

\[
\frac{\text{out}(c, p)}{\nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))|\text{out}(c, \text{dec}(\text{enc}(k, p), k))}
\]

\[
\frac{\nu k z. \text{out}(c, \text{enc}(k, z))}{\nu z. (\text{out}(c, \text{enc}(k, z)); \text{in}(c, = z))}
\]

\[
\frac{\text{in}(c, = z)}{\nu k z. \text{out}(c, \text{enc}(k, z))}
\]

The attacker get $p$ (or $p'$) but the process cannot reduce to 0.
Definition
A completed extended process is a process whose structure is
\[ \mu \tilde{k} (\|P_1\| \ldots \|P_n\|\{x_1/y_1\} \ldots \{x_m/y_m\}) \].

We denote by \( C(P) \), the fact that \( P \) is a completed process.

Definition
\[\text{Labeled bisimilarity under a cautious environment} (\approx_c) \text{ is the largest}\]
\[\text{symmetric relation } R \text{ on closed extended processes such that } A R B \text{ implies:}\]

1. if \( A \rightarrow^* (\exists \alpha \rightarrow \rightarrow^*)^* A' \) for some \( A' \) such that \( C(A') \), \( A \approx_s B \);
2. if \( A \rightarrow A' \), then \( B \rightarrow^* B' \) and \( A' \mathcal{R} B' \) for some \( B' \);
3. if \( A \xrightarrow{\alpha} A' \), \( A' \rightarrow^* (\exists \alpha \rightarrow \rightarrow^*)^* A'' \) for some \( A'' \) such that \( C(A'') \) and \( \text{fv}(\alpha) \subseteq \text{dom}(A) \) and \( \text{bn}(\alpha) \cap \text{fn}(B) = \emptyset \), then \( B \rightarrow^* \xrightarrow{\alpha} \rightarrow^* B' \) and \( A' \mathcal{R} B' \) for some \( B \).

This definition is strictly weaker than labeled bisimilarity.
Anonymity property

Labeled bisimilarity under cautious environment holds for the protocol.

Proposition

Let $A$ and $B$ be two honest payees of the protocol that declare a payment, $C_h$ is an honest custodian at the $h^{th}$ position chosen by both $A$ and $B$. Then the protocol guarantees payees anonymity; that is

$$\nu sk_C.\text{payee}(A)\{ T_1/t \} | \text{payee}(B)\{ T_2/t \} | !C_h$$

$$\approx_c \nu sk_C.\text{payee}(A)\{ T_2/t \} | \text{payee}(B)\{ T_1/t \} | !C_h$$
Conclusion

Features of the protocol:

- Relatively independent from the base payment protocol
- Provides the incomes to the authority
- Investigations cannot be corrupted by custodians
- Links between payees and payers are not automatically known by the authority.

Drawbacks, future work:

- Custodians have to be highly reliable
- Number and amounts of declarations leak some information about payees
- Try to plug the protocol on Zerocoin
- Add zero knowledge proofs to reduce dependency to custodians
new sk1 : skey; let (k1 : pkey) = pk(sk1) in new s1 : bitstring;
new sk2 : skey; let (k2 : pkey) = pk(sk2) in new s2 : bitstring;
new sk3 : skey; let (k3 : pkey) = pk(sk3) in new s3 : bitstring;
new sk4 : skey; let (k4 : pkey) = pk(sk4) in
let(E : bitstring) = sign(skR, (pk(skR), C1, k1)) in
let(E1 : bitstring) = sign(sk1, (C1, k1, C2, k2, s1)) in
let(E2 : bitstring) = sign(sk2, (C2, k2, C3, k3, s2)) in
let(E3 : bitstring) = sign(sk3, (C3, k3, end, k4, s3)) in
out(c, encrypt(A, (E, encrypt(C1, (E1, encrypt(C2, (E2, encrypt(C3, E3))))))));

phase 13;
in(c, Vn : bitstring);

phase 20;
let(= k4, = hash((k4, s3, hash((k3, s2, hash((k2, s1, hash(k1))))))) = checksign(Vn);
out(c, sign((T, k4), sk4));

phase 33;
in(c, F : bitstring);
let(= k1, = hash((hash((hash((k4, s3)), s2)), s1))) = checksign(F, A) in
out(c, sign(skR, (k1, hash((hash((hash((k4, s3)), s2)), s1))))).
let authority(skA:skey) =
  in(c, O: bitstring);
let(E : bitstring, O1 : bitstring) = decrypt(O, skA) in
let(R : pkey, C1 : pkey, k1 : pkey) = getmess(E) in
let(x : bitstring) = checksign(E, R) in
checkmemory(k1);
phase 10;
let(V0 : bitstring) = sign(skA, (k1, hash(k1))) in
out(c, (V0, O1));
phase 32;
in(c, B1 : bitstring);
let(= k1, = hash(k1), H'1 : bitstring, m : amount, p : period)
  = checksign(B1, C1) in
phase 33;
out(c, sign(skA, (k1, H'1, m, p)));
in(c, G : bitstring);
let(= k1, = hash(k1), = H'1, = m, = p)
  = checksign(G, R) in
0.
let secondCustodian(skC : skey) =
phase 11;
in(c, (V1 : bitstring, 02: bitstring));
let(k2 : pkey, H2 : bitstring) = decrypt(V1, skC) in
checkmemory(k2);
phase 12;
let(E2 : bitstring, O3 : bitstring) = decrypt(O2, skC) in
let(C2: pkey, = k2, C3 : pkey, k3 : pkey, s2 : pkey) = checksign(E2, k2) in
let(H3) = hash((k3, s2, H2)) in
let(V2 : bitstring) = sign(skC, (k3, H3)) in
out(c, (V2, O3));
phase 30;
in(c, B3 : bitstring);
let(= k3, = H3, H'3 : bitstring, m : amount, p : period) = checksign(B3, C3) in
out(c, sign(skC, (k3, H3, H'3, m, p)));
phase 31;
let B2 = sign(skC, (k2, H2, hash((H'3, s2)), m, p)) in
out(c, B2);
in(c, rB2);
let(= k2, = H2, = hash((H'3, s2)), = m, = p) = checksign(rB2, C1) in
0.