1. Digital Cash

The advent of efficient telecommunication since the 19th century has facilitated the transfer of value and monetary units across long distances. However, even in the 21th century, such prevailing methods for such transfers involve the intermediation of these transfers via trusted third parties such as banks and credit card companies. Unlike the physical exchange of monetary units such as the exchange of paper bills or coins between two interacting individuals or parties, the involvement of these trusted third parties creates certain risks and potentially undesirable downsides:

- Trusted third party must be reachable when transaction occurs.
- Trusted third party is custodian of money in this process. This can be a problem if the party has problems with solvency or illicitly claiming ownership of the money.
- The trusted third party can censor or prevent transactions without approval of the payer or the payee.
- The trusted third party has extensive insights into the payment history and payment flows between payers and payees.

1.1 Blind Signatures

A key building block of digital cash systems that mitigate the downsides of money transfer via trusted third parties is the concept of a *digital signature*. A digital signature scheme enables each user to *sign* messages so that the digital signature can later be *verified* by anyone else.

In particular, a digital signature scheme specific to a user ALICE consists of a

- A *signing function* $s_{\text{ALICE}} : \mathcal{M} \longrightarrow \Sigma$ that maps messages $m \in \mathcal{M}$ from a finite *message space* $\mathcal{M}$ to an element $s_{\text{ALICE}}(m) \in \Sigma$ of a finite signature space Σ. This signing function is not publicly known, but only to the user ALICE.
- A *verification function* $v_{\text{ALICE}} : \mathcal{M} \times \Sigma \longrightarrow \{\text{TRUE}, \text{FALSE}\}$ which outputs

$$v_{\text{ALICE}}(m,\sigma) = \begin{cases} \text{TRUE}, & \text{if } \sigma = s_{\text{ALICE}}(m), \\ \text{FALSE}, & \text{if } \sigma \neq s_{\text{ALICE}}(m). \end{cases}$$

v_{Alice} is not only known by Alice, but to all users.

Digital signatures alone are, unfortunately, not sufficient to create a digital cash system that comes with privacy features. David Chaum [Cha83] proposed a version of such digital signatures schemes that additionally does not the signing party (Alice) to know the message m. Nevertheless, Alice (or any user person) will be able to check whether she has "signed" the message if she is later presented with both m and the derived signature. For this reason, signature schemes with such features are called *blind signature* schemes.

Chaum's blind signature scheme consists of the following building blocks. As previously, there is a signing function s_{Alice} that is only known to the signer Alice and a verification function v_{Alice} that is publicly known, but specific to Alice's signatures. Additionally, another user Bob can define and apply the following functions:

- A *blinding function* $b_{\text{Alice,Bob}} : \mathcal{M} \longrightarrow \mathcal{M}$ that maps a message $m \in \mathcal{M}$ to a *blinded message* $m' = b_{\text{Alice,Bob}}(m)$. This blinding function is not publicly known, but only to the user Bob.
- An *unblinding function* $u_{\text{Alice,Bob}} : \Sigma \longrightarrow \Sigma$ that maps a blinded signature σ to an unblinded signature $u_{\text{Alice,Bob}}(\sigma) \in \Sigma$. This unblinding function is not publicly known, but only to the user Bob.
- The signing function s_{Alice}, blinding function $b_{\text{Alice,Bob}}$ and unblinding function $u_{\text{Alice,Bob}}$ are related by the relationship

$$u_{\text{Alice,Bob}}(s_{\text{Alice}}(b_{\text{Alice,Bob}}(m))) = s_{\text{Alice}}(m)$$

  for all messages $m \in \mathcal{M}$. In words, this means that Alice's signature $s_{\text{Alice}}(m)$ of the message m coincides with the unblinded output $u_{\text{Alice,Bob}}(s_{\text{Alice}}(b_{\text{Alice,Bob}}(m)))$ of the signed blinded message $s_{\text{Alice}}(b_{\text{Alice,Bob}}(m))$.

1.2 Chaumian e-Cash

We present now the basic steps of an electronic cash system first pioneered by Chaum [Cha83]. It is designed to mimic the behavior of the exchange of physical cash between two interacting parties (here Bob and Carol). In particular, it should make the forgery of payments hard, prevent duplicate payments (so-called *double spending*), and provide a certain level of privacy for the parties involved. The intended privacy is in opposition to transactions facilitated by conventional digital payments systems, such as credit card transactions or ACH transfers.

We restrict ourselves to the setting that the digital cash corresponds to a fixed amount of money that underlies the digital cash, such as \$1. There are several ways to implement similarly other payments amounts (such as \$10, \$42.59 etc.): For example, one can define different signing/blinding functions for each round payment amount, or a "cut and choose" procedure in which only a random subset of the available e-cash serial numbers are signed, while the rest are required to be unblinded by the bank [GB08, Section 12.5.5].

In this section, we do not discuss a practical implementation of the signing, blinding, unblinding and verification functions. The most common one and the one proposed by Chaum is based on the RSA cryptosystem [GB08, Section 7.2.3]. The Chaumian e-cash protocol has three main components: the *withdrawal protocol*, the *payment protocol*, and the *deposit protocol*, detailed below.

Chaumian e-cash comes with a high level of privacy for the payer Bob, but with no significant privacy for the payee Carol.

Within the Bitcoin ecosystem, scaling solutions based on Chaumian e-cash issued by bitcoin banks or bitcoin federations have been recently studied with significant interest [Cas; Fed].

Chaumian e-Cash: *Withdrawal Protocol: Bob withdraws e-cash worth* $1 *from bank Alice.*

Input: Availability of $1 in account of Bob at bank Alice.
Output: Bob has e-cash note information (m, σ) available or False

```
1  Bob chooses a random message $m \in \mathcal{M}$.
2  Bob blinds message $m$ by computing $b_{\text{Alice,Bob}}(m)$.
3  Bob forwards $b_{\text{Alice,Bob}}(m)$ to bank Alice.
4  Bank Alice signs blinded message by computing $s_{\text{Alice}}(b_{\text{Alice,Bob}}(m))$.
5  Bank Alice sends $c = s_{\text{Alice}}(b_{\text{Alice,Bob}}(m))$ back to Bob.
6  Bank Alice debits account of Bob by $1.
7  Bob unblinds the received data by computing $\sigma = u_{\text{Alice,Bob}}(c)$.
8  Bob checks Alice's signature by applying the verification function
   $v_{\text{Alice}}(u_{\text{Alice,Bob}}(c), m)$.
9  if $v_{\text{Alice}}(u_{\text{Alice,Bob}}(c), m)$ = True then
10     return e-cash note information $(m, \sigma)$ to Bob.      ▹ Withdrawal successful
11 else
12     return False.              ▹ Withdrawal unsuccessful, signature not accurate
```

Chaumian e-Cash: *Payment Protocol: Bob sends e-cash worth* $1 *to Carol.*

```
1  Bob sends $m \in \mathcal{M}$ and $\sigma$ to Carol.
2  Carol checks Alice's signature by applying the verification function $v_{\text{Alice}}(m, \sigma)$.
3  if $v_{\text{Alice}}(m, \sigma)$ = True then
4      Carol accepts payments.
5  else
6      return False.              ▹ Payment unsuccessful, as signature not genuine
```

Chaumian e-Cash: *Deposit Protocol: Carol deposits e-cash worth* $1 *at bank Alice.*

Input: List of cleared e-cash notes $\mathcal{L}$ at bank Alice.
Output: Updated list of cleared e-cash notes $\mathcal{L}$ at bank Alice.

```
1  Bob sends $m \in \mathcal{M}$ and $\sigma$ to bank Alice.
2  Alice checks her own signature by applying the verification function $v_{\text{Alice}}(m, \sigma)$.
3  if $v_{\text{Alice}}(m, \sigma)$ = True then
4      if $m \in \mathcal{L}$ then
5          Alice denies payment.          ▹ Deposit unsuccessful, double spend detected.
6      else
7          Append $m$ to list of cleared e-cash notes $\mathcal{L}$: $\mathcal{L} = \mathcal{L} \cup \{m\}$.
8          Alice accepts deposit.
9          Alice credits account of Carol by $1.            ▹ Deposit successful.
10 else
11     return False.              ▹ Deposit unsuccessful, signature not genuine.
```

References

[Cas] Cashu. “Cashu, a free and open-source Chaumian ecash protocol built for Bitcoin”. `https://cashu.space` (cited on page 4).

[Cha83] David Chaum. “Blind signatures for untraceable payments”. In: *Advances in Cryptology: Proceedings of Crypto 82*. Springer. 1983, pages 199–203 (cited on page 4).

[Fed] Fedimint. “Fedimint, a modular open source protocol to custody and transact bitcoin in a community context, built on a strong foundation of privacy”. `https://fedimint.org/docs/intro` (cited on page 4).

[GB08] Shafi Goldwasser and Mihir Bellare. “Lecture Notes on Cryptography”. In: *Summer course 6.87s on cryptography at MIT, available at* `https://cseweb.ucsd.edu/~mihir/papers/gb.pdf` (2008) (cited on page 4).