Technology

Crypto CEO Dies Holding Only Passwords That Can Unlock Millions in Customer Coins

By Doug Alexander

February 4, 2019, 9:54 AM PST

- Without digital keys, clients lose access to coins, funds
- Board said last week that it was seeking creditor protection

In this article

0915448D
ERNST & YOUNG LTD
Private Company

Digital-asset exchange Quadriga CX has a $200 million problem with no obvious solution -- just the latest cautionary tale in the unregulated world of cryptocurrencies.

The online startup can't retrieve about C$190 million ($145 million) in Bitcoin, Litecoin, Ether and other digital tokens held for its customers, according to court documents filed Jan. 31 in Halifax, Nova Scotia. Nor can Vancouver-based Quadriga CX pay the C$70 million in cash they're owed.

CS 168: Blockchain and Cryptocurrencies

Digicash, Part 2: Anonymous Accountability

Prof. Tom Austin
San José State University

Review Lab

Properties of an ideal digital currency
(Okamoto and Ohta)

1. **Independence**
   - not dependent on any physical condition
   - can be transferred through computer networks
2. **Security**
   - cannot be copied and reused (double-spending)
   - cannot be forged
3. **Privacy**
4. **Offline Payment**
   - No need to be linked to a host to process payment
5. **Transferability**
6. **Divisibility**

Properties of an ideal digital currency
(Okamoto and Ohta)

1. **Independence**
   - not dependent on any physical condition
   - can be transferred through computer networks
2. **Security**
   - cannot be copied and reused (double-spending)
   - cannot be forged
3. **Privacy**
4. **Offline Payment**
   - **No need to be linked to a host to process payment**
5. **Transferability**
6. **Divisibility**

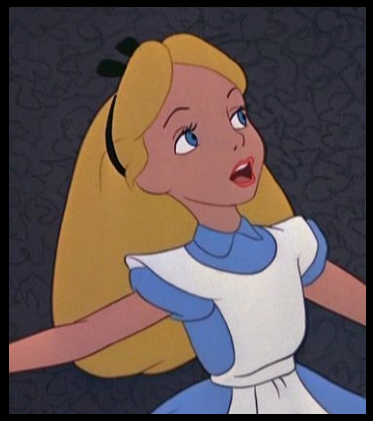
- Blinded signatures
  - anonymity
- Central clearinghouse
  - double-spending
- Supports online and offline payments
- Bankrupt in 1998

Problem using blind signatures

When the bank signs blindly, how do they know they are not being cheated?

E.g. – Trudy buys a $10 coin from the bank, but tricks the bank into signing $100 coin

Solution: Cut and Choose



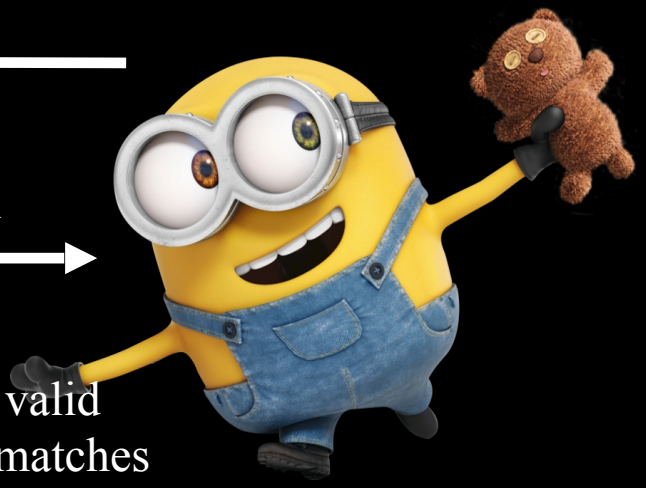
Alice

1) 100 blinded document hashes

2) Selected document (0-99)

3) Documents, blinding factors for all documents hashes NOT selected

4) Bob verifies:
- Doc. contents valid
- Blinded hash matches doc. & blinding factor



Bob

5) Signed selected document hash

Centralized Cryptocurrencies

- Our initial version:
  - provides anonymity
  - bank can still verify correct amount
- (See Schneier's "Applied Cryptography")

1. Alice prepares 100 anon $1k money orders.
2. She places each money order in an envelope.
3. Bank opens 99 orders & verifies the amount.
4. The bank (blindly) signs the last order and deducts $1k from Alice's account.
5. Alice takes the anonymous money order out of the envelope and spends it.
6. Merchant accepts money order after verifying bank's signature.
7. Merchant takes money order to the bank.
8. Bank verifies signature and pays money.

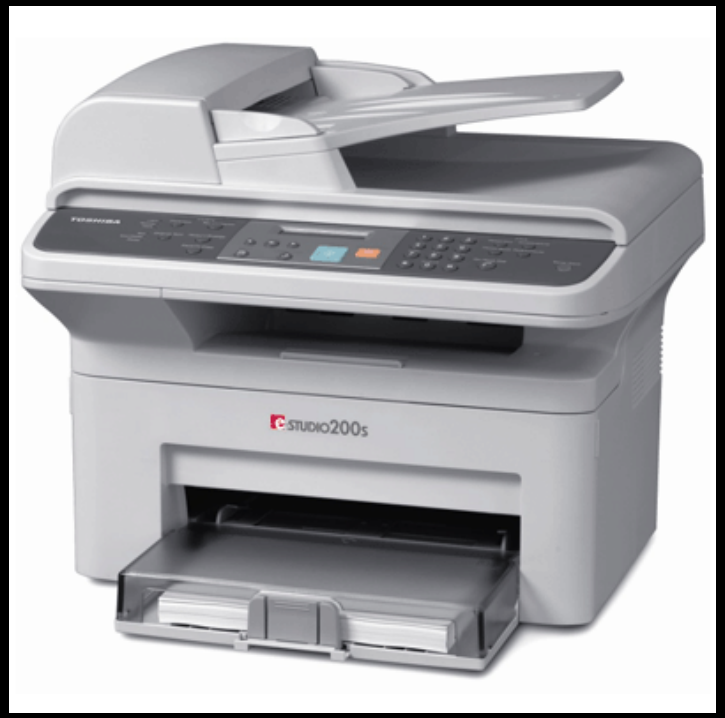
Discussion Question

We could get a protocol very similar to DigiCash without using blinded signatures.

Instead of sending blinded hashes, we could just send the hashed document.

What property would that approach *not* provide?

The previous currency still does not address the *double spending problem*.



The next version will solve that issue.

1. Alice prepares 100 anon $1k money orders **and places a random global identifier string on each**.
2. She places each money order in an envelope.
3. Bank opens 99 orders & verifies the amount.
4. The bank (blindly) signs the last order and deducts $1k from Alice's account.
5. Alice takes the anonymous money order out of the envelope and spends it.
6. Merchant accepts money order after verifying bank's signature.
7. Merchant takes money order to the bank.
8. Bank verifies signature, **checks that the identifier has not been previously used**, and pays money.

Online and Offline Payments

DigiCash supported 2 approaches

- Online (the easy one)
  - Merchant checks that GUID has not been redeemed with the bank.
  - **Prevents** double-spending.
  - Requires more network traffic.
- Offline
  - Bank is not available.
  - **Detects** double spending.
  - **Identifies cheaters** when merchant deposits coin.

Random Identity String (RIS)

- Each bill includes hashes of N copies of purchaser's identity.
  - Identity encrypted with one-time pad (OTP).
  - Provably unbreakable.
- Anonymity is preserved in normal cases.
- If a coin is double-spent, bank can identify cheater.

One-Time Pad Review

Key is as long as the original message

Plaintext: 0101 1010 0101 1011 0101

⊕ **Key:** 1011 0010 1101 1001 0001

Ciphertext: 1110 1000 1000 0010 0100

Minting coin

Same as in previous protocol, except

- Purchaser must share hashes of identity pairs
  - H(key) and H(ciphertext)
- During reveal phase, purchaser must share key and ciphertext for all unblinded coins

Payment Phase

Same as before, except:

- Merchant randomly selects either the left or right half of each identity pair.
- Client shares the selected half.
  - One half will not reveal identity.
- Merchant checks the hashes, and then accepts the coin.

Review lab code
(in-class)

Lab, Setup

1. Download `coin.js`, `driver.js`, and `utils.js` from the course website.

2. In the same directory, install the blind-signatures library:
   ```
   $ npm install blind-signatures
   ```

Lab, part 1:
Implement Merchant Logic

Update the `acceptCoin` method in `driver.js` to simulate the merchant's role.

Details in Canvas.

Deposit Phase

- Merchant redeems coin.
- As before, bank verifies that GUID is unused.
- *If GUID has previously been used*
  - Bank compares each position of the two identity strings.
  - If the client cheated, their identity is revealed.
  - If the identity strings are identical, the merchant is trying to double spend.

Lab, part 2:
Implement Bank Cheater Identification

When the bank receives the same coin twice, it uses the two identity strings to reveal the cheater.

Implement the `determineCheater` function in `driver.js` with this logic.

Details in Canvas.

HW1: Review assignment
(in class)