



Electronic Cash

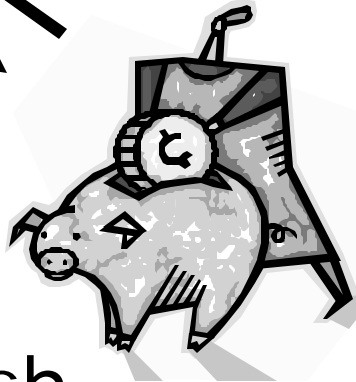
The Problem

Alice pays Merchant
Bob with ecash



Alice opens
account.

Withdraws ecash



Bob deposits
ecash in
his account



Ecash Requirements

- Maintain properties of physical cash
- Unforgeability
- Untraceability
 - Bank cannot link a deposit to a withdrawal
 - Required to maintain anonymity
- Double spending problem
 - A digital copy of ecash could be spent elsewhere



Blind Signature

- Recall RSA --signatures
 - (d, N) is private key and (e, N) public key
 - Signature: $m^d \bmod N$
 - Anyone can decrypt this with public key
 - Verify that only owner could sign it
- Blind signature
 - Sign a message without revealing the message to the signer



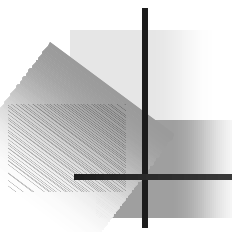
Blind signature

- Alice sends Bob $s = (r^e m) \bmod N$
 - r is a random number, not revealed to Bob
- Bob computes $t = s^d \bmod N$ and returns it to Alice
- Alice computes $t/r \bmod N = m^d \bmod N$
 - Obtains Bob signature on message m
 - Hasn't revealed m to Bob



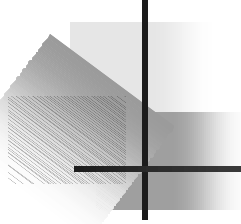
Ecash –blind signatures

- Alice and Bank work together to produce \$1
 - Bank signs the \$1 ecash certificate
 - Bank doesn't have information that this signed \$1 belongs to Alice
- Valid \$1 bill is a pair = (x, y)
 - $y = f(x)^d \text{ mod } N$
 - $f()$ is a one-way hash function



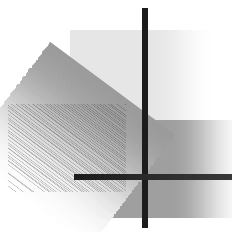
Ecash -protocol

- Alice withdraws \$1 by
 - Picks x , computes $f(x)$
 - Gets a blind signature on $f(x)$ from Bank
 - Alice sends bank $s = (r^e f(x)) \bmod N$
 - Bank sends Alice $t = s^d \bmod N$
 - Alice has $y = t/r = f(x)^d \bmod N$
- Alice pays Bob by sending (x, y)



Ecash --protocol

- Bob deposits (x,y) with the bank
- Bank can verify that $y^e = f(x)$
- Bank can check for double spending
 - Not on the list of previously deposited bills
- Bank cannot link this deposit to Alice
 - Blind signature on a random string s



Ecash –Forging

- Imagine \$1 bill = (x,y) , $y = x^d \bmod N$
- Alice can pick y
 - Then, compute $x = y^e \bmod N$
 - Now, has a feasible (x,y) pair
- By using, one-way hash function $f()$
 - Requires Alice to invert the one-way hash
 - Forging is difficult
 - Gets $f(x)$, not x , from $y^e \bmod N$



Multiple denominations

- Choose multiple key pairs
 - One for each denomination
- A second approach, choose different encryption exponents
 - $e = 3$ for \$1 bill, 5 for \$5 bill, 7 for \$10 bill
 - These exponents need to be mutually prime
 - If not, can forge bills



Ecash --offline

- The above scheme requires bank to be online all the time
- Not a problem now
- But, can we design a scheme that does not require bank to be online all the time?



Offline ecash

- Let bank detect double spending
 - Bank not online all the time to prevent
- If user doesn't double spend, remains anonymous
- If user double spends, user ID is revealed
 - Take suitable action, charge a fine, put in jail etc..

Offline ecash --format

Bank Name
Random Serial Number
One Dollar
$f(x_1), f(x_1 \oplus ID)$
$f(x_2), f(x_2 \oplus ID)$
....
$f(x_k), f(x_k \oplus ID)$



Offline ecash

- Alice generates k random numbers x_i for each bill
- Computes $f(x_i)$ and $f(x_i \text{ XOR ID})$
- Gets the bank to blind sign the bill with these numbers



Offline ecash

- Bank randomly picks N out of M bills it signs for Alice
- Asks Alice to “unblind” them to make sure Alice is actually including her ID in the bills
- Bank assumes that remaining $N-M$ bills include Alice’s ID



Offline ecash

- Alice pays Bob, the Merchant with signed cash
- Bob sends a challenge bit stream b_i
- If $b_i = 0$, Alice reveals x_i , $b_i = 1$, Alice reveals $(x_i \text{ XOR ID})$, for $i = 1, \dots, k$
- Bob can verify each response from Alice
 - Against the information in the \$1 bill



Offline ecash

- Bob sends ecash to Bank along with the challenge bit string and the revealed information from Alice –for deposit
- If Alice double spends, the challenge strings given by Merchants will differ in one bit position with a high probability
- Then, bank will have both x_i and $(x_i \text{ XOR ID})$
 - Revealing Alice's ID



Offline ecash

- Sequence number is needed
- Without it, Alice can double spend by permuting the $f(x_i)$, $f(x_i \text{ XOR ID})$ pairs



Ecash Additional requirements

- Unlinkability: Given two bills, bank cannot tell if they come from the same user
- Divisibility: Bill can be broken up and spend at multiple merchants
- Anonymity Revocation: Based on Judge's orders, to trace illegal transfers
 - Money laundering –owner tracing
 - Blackmail –Bill tracing