TOPIC: 6.857 - Computer & Network Security

DATE: 5/2/12

FILE UNDER:

PAGE: L22.1

Admin:

Talks start 5/9; papers due 5/16

Today:

"Electronic cash"

- Basics: Atoms/bits, tokens/accounts
- Electronic checks
- Desirable Properties
- Double-spending
- Coins & Peppercoin
- Anonymity

[Following notes from L22, 4/27/11]

"Electronic Money"

- What properties should it have?
- " " can " " ?

Atoms vs Bits

- What can "possessing value" (money) mean?
- How can we transfer value?

Easy to answer if we use (gold) atoms to represent value:

- gold atoms are hard to make
- only one person at a time can "own" an atom

Things get complicated if we want to use bits:

- easy to generate bits
- bits can be copied ⇒ double-spending becomes a problem!

(Token-based) Possession-based vs Account-based methods

- In a possession-based method, owning the representation ≡ owning the value
- In an account-based method, there is usually some TTP who "maintains accounts" (e.g. a "bank"); xactions cause value to be shifted from one acct to another.
- Most "bit-based" methods are account-based.

Simple example: Electronic checks

- Account-based: Bank has PK_B, SK_B
- User has PK_u, SK_u, cert on (U, PK_u) by bank
- Check = [cert (on PK_u, signed by SK_B) ; sign(SK_u, "Pay Bob $100, date, serial #")]
- Bank deposits check just once (using ser #)
- Usual problem of overdrawn acct (bad check)
- Bank knows xact details: payer, payee, amt, date
- Merchant " " "

This works.

What else is possible?

Can we make payments more like cash?

Desirable (?) Properties

- Non-forgeable (prevent fraud, inflation)
- Not double-spendable
- Reliability: can "back up" your $
- Exclusive ownership
- Transferability: A can pay B
- Transitivity: B can use A's payment to pay C
- Variable-denominations
- Divisibility & combinability
- Efficiency (esp. for small amts)
- On-line versus off-line transactions
- Scalability
- anonymity
- security
- Conversion to "ordinary" money

6.857 Rivest
L 22.5 4/27/11

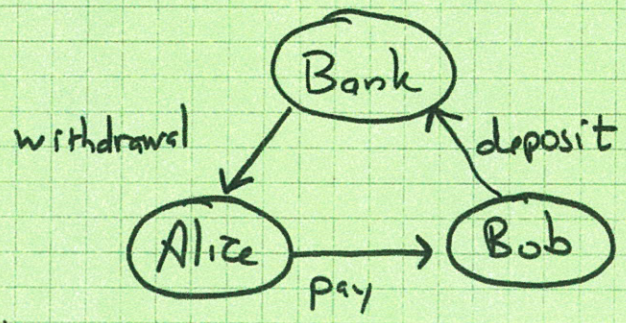
Double-spending

- essentially a "replay attack"
- if you can backup your $, then "restore" gives you your spent money back !?
- prevention seems really tough (unless you use atoms)
- detection requires convergence of spending records (e.g. at bank) and large databases (?)
- even if you can detect double-spending – what do you do?
  - roll back / deny transaction (2nd merchant to get same electronic coin can't deposit it)
  - punishing perpetrator may be impossible if we have (true) anonymity: payer is not identifiable
  - furthermore: is payer or payee the culprit? (can merchant "frame" consumer?)
  - deterrence may be hard... how to punish (pay fine from account?)

Some approaches:

Signed coin ID

Bank (TTP)
Alice (payer)
Bob (payee)



3 protocols to support:

1. withdrawal / authorization
   Alice becomes "able to pay"
   (e.g. cert issuance in check scheme)
2. payment
3. deposit

} life of a "coin"

① withdrawal:

- Bank gives Alice $\underbrace{R, \text{sign}(SK_B, R)}_{= \text{coin}}$ ← unforgeable object!

  R is coin ID

- Bank keeps R in database of unspent coins
- Bank debits Alice's acct for withdrawal

② payment:

Alice gives coin to Bob; Bob checks Bank sig

③ deposit:

Bob gives coin to Bank, Bank checks sig & R in DB
flags R as "spent"

- Not very efficient – bank has to sign each coin!
- Double-spending can be a problem!
- Check scheme better – merchant can't frame user!

Pepperoin (Micali & Rivest)

- "probabilistic payments":

  paying 10 ¢ (micropayment) $\equiv$ paying \$10 with probability 1/100 (macropayment, sometimes)
- based on electronic checks method
- Alice pays Bob 10 ¢ as follows:

  She gives Bob electronic check for \$10 that contains <u>condition</u>: "This check valid if and only if E is true" (where E holds with probability 1/100)
- Bob must be able to test if E is true
  - if so, he can deposit check
  - if not, he throws check away (but gives Alice her purchase)
  - he gets paid correctly <u>on the average</u> (law of large numbers)
- Alice should not be able to tell if E is true when she writes check (else she can filter checks...)
- Bank <u>should</u> be able to tell if E is true (so "bad checks" where E is false, don't get deposited.

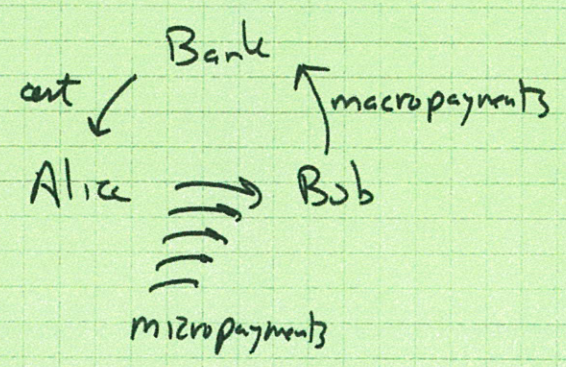
Peppercoin (cont.)

- Our recommendation for E
  - Bob has a <u>deterministic</u> signature scheme
  - E true $\equiv \left[\underbrace{\text{sign}(SK_B, \text{check})}_{\text{"countersignature" on check}} \bmod 100 = 0 \right]$
  - (can adjust odds based on value...)
  - Bob's signatures should be pseudorandom (unpredictable)
- Bank only sees 1/100 of checks Alice writes
  <u>ideal for micropayments</u>

- Merchant Bob may be ok with probabilistic payment; User Alice may not be (she may get charged for more than she paid)

  <u>Fix:</u> Each check also gives running total of all checks written; bank won't charge her more than that amt. (If she tries to cheat, it will get detected, e.g. when two checks in a row get converted to macropayments.) Note that bank acts as buffer here...

<u>Good efficiency</u>: work done is all between Alice & Bob



Anonymity

- Using blind sigs to achieve anonymity (Chaum)
- catching double-spenders (Brands)

Chaum's Anonymous Cash

- Bank signs all coins
- But we now use "blind signature" method, so bank doesn't see what it is signing (!)
- E.g. with RSA sig: $SK_B = (n, d)$ $PK_B = (n, e)$

$$\text{coin} = (m, \underbrace{m^d}_{\equiv \text{sign}(SK_B,\, m)}) \quad \text{where } m = \text{serial \#}$$

(formatted msg, ...)

How to get m^d from bank, without bank seeing m?

Homomorphic property of RSA:

Alice		Bank
random r	$\xrightarrow{m \cdot r^e}$	sign $(m \cdot r^e)^d = m^d \cdot r$
divide by r $\Rightarrow m^d$	$\xleftarrow{m^d \cdot r}$	

<u>Value</u> of coin is fixed (indep of m) since bank doesn't see m.

<u>Value</u> of coin depends only on PK.

- Double-spending a problem!

6.857 Rivest
L22.10 4/27/11

Catching double-spenders (Brands)

- coin withdrawal is anonymous (like Chaum)
- spending coin is anonymous
- <u>but</u> if Alice spends coin twice, her identity is revealed! (Bank can figure it out)

Ideas:

- Bank gives blind signature on coin
- User knows secret about coin
- secret tied to her identity
- payment protocol gives up "secret share" about coin secret to payee
- two secret shares ⇒ coin secret ⇒ user ID

(For details: see lecture notes L21 & L22 from 2009 6.857)

"Intergalactic Banking Model"??

How to make monetary system between star systems?

No TTP.

No use of force.

No atoms xferred - only bits.