Peppercoin Micropayments: How to make micropayments really work for e-Business!

Ronald L. Rivest
MIT Laboratory for Computer Science

(joint work with Prof. Silvio Micali)
Center for eBusiness luncheon 12/11/2002
Outline

- Need for micropayments
- Previous Micropayment schemes
- The “Peppercoin” scheme
  (English law says a peppercorn is smallest amount that can be paid in a contract)
    - Improve “micropayments via lottery tickets” to achieve:
      » Non-interactivity, via recipient signatures
      » User-fairness, via serial numbers
Payment Framework:

Payment System Provider (PSP), Bank

Authorization

Payment(s)

Deposit(s)

User Alice

Merchant Bob
Payment schemes

- Dominant today:
  - Credit cards
  - Subscriptions
  - Advertisements

- Other possibilities:
  - Electronic checks
  - Anonymous digital cash
  - Micropayments
What is a “micropayment”?

- A payment in the range 0.1¢ to $10.
- A payment small enough that processing it is relatively costly. (Processing one credit-card payment costs about 25¢!)
- *Processing cost* is the key issue for micropayment schemes.
The need for small payments

- “Pay-per-click” purchases on Web:
  - Music, video, information
- Mobile commerce ($20G by 2005)
  - Location-based info services, gaming, sodas, parking
- Infrastructure accounting:
  - Bandwidth
Why aren’t micropayments already here?

- The market need is still nascent.
- Rolling out a new payment system requires the coordination of many players.
- But fundamentally: COST! Previous micropayment schemes are too costly to implement.
Previous Work: Digital Cash

- Example: Chaum’s digital coins
- Emphasis on *anonymity*: Withdrawals use blind signatures
- Problem of double-spending handled by having doubler-spenders revealed (e.g. Brand’s protocol)
- No aggregation: every coin spent is returned to the PSP.
Previous Work: PayWord

- Rivest and Shamir ’96
- Emphasis on reducing public-key operations by using per user/merchant hash-chains instead:
  \[ x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow \ldots \rightarrow x_n \]
- User signs \( x_0 \) over to merchant and releases next \( x_i \) for next payment
- Session-level aggregation only.
Previous Work: Millicent

- Manasse et al. ‘95
- User buys merchant-specific scrip from PSP for each session.
- Requires PSP to be on-line for scrip purchase
- Session-level aggregation only
Dimensions to consider:

- Aggregation (*global*)
- PSP on-line or off-line? (*off-line*)
- Interactive vs. non-interactive (*non*)
- Computation Cost (*cheap*)
- User-fairness (*fair*)
- ... (many other issues, too)
Aggregation

- To reduce cost, *micropayments* should be aggregated into fewer *macropayments*.
- Possible levels of aggregation:
  - **None**: PSP sees every payment
  - **Session-level**: aggregate all payments in one user/merchant session
  - **Global**: Payments aggregated across users and merchants
- Can be **deterministic** or **statistical**.
On-line vs. Off-line

- **On-line PSP** authorizes each payment or session.  
  (Example: credit cards)

- **Off-line PSP** not needed to initiate session or make payment.  
  (Example: paper money used for taxi.)
Interactive vs. Non-interactive

- **Interactive:**
  Payment protocol is *two-way*:

- **Non-interactive:**
  Payment protocol is *one-way*
  (e.g. anti-spam payment in email):
Digital signatures are still relatively expensive --- but much cheaper than they used to be!

It now seems reasonable to base micropayments on digital signatures. (E.g. Java card in cell phone)

User and merchant are anyways involved with each transaction; digital signatures add only a few milliseconds.

On-line/Off-line signature can also help.
Ability to handle disputes

- User claims he didn’t approve payment
  Solution: use digital signatures

- User claims goods are poor quality or were never sent.
  Solution: let user complain to merchant directly.

- A micropayment PSP can’t afford to handle *any* such disputes!
Previous Work: Lottery Tickets

- “Electronic Lottery Tickets as Micropayments” – Rivest FC ’97 (similar to “Transactions using Bets” proposal by Wheeler ’96)
- Payments are probabilistic
- First schemes to provide global aggregation: payments aggregated across all user/merchant pairs.
Probabilistic Payments

Would you, as a merchant, prefer:

(a) 10 cents for each payment for sure,
or
(b) $10 for 1/100 payments, and
  $ 0 for 99/100 ?
Probabilistic Payments

Would you, as a merchant, prefer:

(a) 10 cents for each payment for sure,
or

(b) $10 for 1/100 payments, and $0 for 99/100?

What if processing charge was $0.25/payment?

(a) nets negative 15 cents/payment!
(b) nets 9.75 cents/payment (average)
“Lottery Tickets” Explained

- Assume all payments are for ten cents.
- Merchant gives user $y = \text{hash}(x)$
- User writes check: “Pay Merchant $10$ if two low-order digits of $\text{hash}^{-1}(y)$ are 75.” (Signed by user, with cert from PSP.)
- Merchant “wins” $10$ with probability $1/100$. Expected value of payment is 10 cents.
- Bank sees only 1 out of every 100 payments. (A plus for user privacy!)
Our “Peppercoin” Proposal

- Peppercoin improves lottery ticket scheme, making it:
  - Non-interactive (by using merchant signatures)
  - Fair to user: user never “overcharged” (by using serial numbers)
Peppercoin Solution

Peppercoin Fairness:
- User, merchant and bank cannot cheat
- Fair to user always (never overcharged)
- Fair to merchant and bank on average

Enable 100 Transactions at Cost of 1
Non-interactive

- Revised check:
  "Pay Merchant $10 if two low-order digits of the hash of Merchant’s digital signature on this check are 75."

- Merchant’s deterministic signature scheme unpredictable to user.

- Merchant can convince PSP to pay.
Optimization for less Signing

- “Pay Merchant $10 if the two low-order digits of the hash of Merchant’s digital signature on the date of this check are 75.”
- Merchant only signs once a day.
User Fairness: No “Overcharging”

- Concern: unlucky user might pay $10 for his first ten-cent payment!
- A payment scheme is *user-fair* if user never pays more than he would if all payments were deterministic ten-cent checks.
Achieving User-Fairness

- User must sequence number his payments: 1, 2, ...
- When merchant turns in winner with sequence number \(N\), user charged \(N - (\text{last } N \text{ seen})\) dimes

![Dimes illustration]

User charged thirty cents for ✓
User-Fairness (continued)

- Merchant is still paid $10 for each winning payment.
- Users severely penalized for using duplicate sequence numbers.
Variable-sized payments

- To make micropayment of size \( m \):
  - Chance of “winning” becomes \( \frac{m}{M} \)
  where \( M \) is the macropayment size.
- “Serial number” on user checks reveal cumulative amount spent so far.
Is revenue variance an issue?

**Theorem.** If new scheme reduces merchant fees by $R$ percent of transaction value, then merchant will be ahead (with probability $999,999/1,000,000$) after only \((5 / R)^2\) macropayments have been received.

**Example:** micro = 0.10, macro = $10, otherfee = 0.04, ourfee = 0.01, 
\[ R = (0.04-0.01)/0.10 = 0.30, \]
\[ (5/R)^2 = 277 \] or $2770 total value.
Conclusion

- Peppercoin micropayment scheme
  - Is *highly scalable*: bank supports *trillions* of micropayments by processing only *billions* of transactions
  - Provides *global* aggregation
  - Supports *off-line non-interactive* payments
  - Is *user-fair* and quite *private*
  - Uses digital signatures, but lightly.

- Start-up: [www.peppercoin.com](http://www.peppercoin.com)
(The End)