Lecture 15
November 19, 2001

- HW2
- Authentication & Identification
- Secret Sharing
- Blind signatures + e-cash

Fiat-Shamir ID Scheme

\[ p, q \text{ – large primes} \]
\[ n = pq \]
\[ t \text{ – security parameter} \]

Publics: \( n, t, ID = x^2 \mod n \)

Secrets: \( p, q \text{ – global} \), \( x \text{ – Alice} \)

Note: \( \gcd(x, n) = 1 \)

1. \( Alice: k \in [1, n[ \), \( w = k^2 \mod n \)
2. \( Bob: r \in [0,1] \)
3. \( Alice: y = kx^r \mod n \)
4. \( Bob: y^2 = ? = wID^t \mod n \)

Repeat t times!
Schnorr ID Scheme

- $p$ – large prime
- $q$ – large (prime) divisor of $p - 1$
- $g$ – generator
- $a = g^{(p-1)/q} \mod p$
- $t$ – security parameter (40?)

Publics: $p, q, g, a, t, \text{ID} = a^{-t} \mod p$
Secrets: $x$ (Alice)

Alice: $k \in [0, q[, \ w = a^k \mod p$
Bob: $r \in [1, 2^t]$ 
Alice: $y = k + xr \mod q$
Bob: $w = ? = a^y \text{ID}' \mod p$

---

Okamoto ID Scheme

- $p$ – large prime
- $q$ – large (prime) divisor of $p - 1$
- $g$ – generator
- $a_1 = g^{(p-1)/q} \mod p$ \hspace{8pt} $a_2 = a_i^c \mod p$
- $t$ – security parameter

Publics: $p, q, g, a_1, a_2, t, \text{ID} = a_1^{-x_1} a_2^{-x_2} \mod p$
Secrets: $x_1, x_2$ – Alice, $c$ – global

Alice: $k_1, k_2 \in [0, q[, \ w = a_1^{k_1} a_2^{k_2} \mod p$
Bob: $r \in [1, 2^t]$ 
Alice: $y_1 = k_1 + x_1 r \mod q$ \hspace{8pt} $y_2 = k_2 + x_2 r \mod q$
Bob: $w = ? = a_1^{y_1} a_2^{y_2} \text{ID}' \mod p$
Okamoto ID Scheme (contd)

If Eve can impersonate Alice then Eve and Alice can compute $c!$

Guillou-Quisquater (GQ) ID Scheme

$p, q$ – large primes
$n = pq$
$e$ – global 'encryption' key
Publics: $n, e, ID = (x^{-1})^e \mod n$
Secrets: $p, q$ – global, $x$ – Alice

Alice: $k \in [0, n[$, $w = k^e \mod n$
Bob: $r \in [0, e[$
Alice: $y = kx^r \mod n$
Bob: $w = ? = ID^r y^e \mod n$

Cert = \{"Alice", ID\}^{CA}

Error(s) in book!
**GQ Identity-based Scheme**

\[ p, q \text{ -- large primes} \]
\[ n = pq, \ (e, d) \text{ -- RSA key-pair} \]
\[ x = (h(\text{Alice})^{-1})^d \]
\[ \text{i.e., TTP signs } h(\text{Alice}) \]

\[ \text{Publics: } n, e, ID = h(\text{Alice}) \mod n \]
\[ \text{Secrets: } p, q, d \text{ -- global, } x = \text{Alice} \]

\[ \text{Alice: } k \in [0,n[, \ w = k^e \mod n \]
\[ \text{Bob: } r \in [0,e] \]
\[ \text{Alice: } y = kx^r \mod n \]
\[ \text{Bob: } w = ? = ID^r y^e \mod n \]

**Converting ID to Signature Scheme (Schnorr)**

\[ p \text{ -- large prime} \]
\[ q \text{ -- large (prime) divisor of } p - 1 \]
\[ g \text{ -- generator} \]
\[ a = g^{(p-1)/q} \mod p \]
\[ t \text{ -- security parameter (40?)} \]

\[ \text{Publics: } p, q, g, a, t, ID = a^{-x} \mod p \]
\[ \text{Secrets: } x \text{ (Alice)} \]

\[ \text{Alice: } k \in [0,q[, \ w = a^k \mod p \]
\[ \text{Bob: } r \in [1,2^t] \]
\[ \text{Alice: } y = k + xr \mod q \]
\[ \text{Bob: } w = ? = a^r ID^x \mod p \]
Secret Sharing

Why share secrets?

- Critical services: access by consent
- Replicate/backup valuable data
- In general, shared control...

E.g., 2 out of 3

Unanimous consent (t-out-of-t)

TTP is needed to generate and distribute the secret.

**SETUP:**

- \( m \) – large number
- TTP generates:

\[
\forall i \in [1, t-1] \quad S_i \in \mathbb{Z}_m \\
\text{and} \\
S_i = K - (S_1 + \ldots + S_{i-1})
\]

\( S_i \) is given to \( P_i \)

**RECONSTRUCTION:**

- Alice, Bob, Eve pool together:

\[
K' = S_1 + S_2 + S_3 = S_1 + S_1 + K - (S_1 + S_2) = K
\]

Note: why not just split the secret in a smaller chunks?
Threshold Scheme (Shamir’79) (t-out-of-n)

Need a TTP to set up the system!

SETUP:
- p - large prime, p > max(K, n), t < n
- TTP generates:
  ∀ i ∈ [1, n]  x_i ∈ Z_p
  ∀ i ∈ [1, t]  a_i ∈ Z_p
  ∀ i ∈ [1, n]  y_i = f(x_i) is given to P_i
  where:
  f(x) = a_0 x^0 + a_1 x^1 + ... + a_t x^t mod p
  a_0 = K
  publics: \{x_1, ..., x_n\}
  secrets: \{a_0, a_1, ..., a_{t-1}\}

RECONSTRUCTION:
t participants pool together:
  y_1 = a_0 x_1^0 + a_1 x_1^1 + ... + a_{t-1} x_1^{t-1}
  ...
  y_t = a_0 x_t^0 + a_1 x_t^1 + ... + a_{t-1} x_t^{t-1}
  t equations, t unknowns yield unique solution vector: <a_0, ..., a_{t-1}>

Shamir Threshold Schemes (example)

SETUP:
p = 17
n = 5, t = 3
TTP generates:
x_i = i for all i
a_i ∈ Z_p ∀ i ∈ [1, 2]
a_0 = K
y_i = f(x_i) is given to P_i
where:
f(x) = a_0 x^0 + a_1 x^1 + a_2 x^2 mod 17
y_1 = 8, y_2 = 13, y_3 = 9
y_4 = 5, y_5 = 11

RECONSTRUCTION:
Suppose P_1, P_3, P_5 pool their shares
y_1 = 8, y_3 = 10, y_5 = 11
a_0 + a_1 + a_2 = 8
a_0 + 3 a_1 + 9 a_2 = 10
a_0 + 5 a_1 + 8 a_2 = 11
a_0 = 13
a_1 = 10
a_2 = 2
Electronic Cash

Outline

- What is electronic cash?
- Why electronic cash?
- Issues:
  - Off-line overspending
  - Anonymity
- How does e-cash work?
- Adding trustee trace-ability
- The anonymous change problem
Motivation

Conventional Cash is:

• Counterfeitable
• Slow
• Costly
• Vulnerable
• Bad for Remote Transactions

Credit Cards, Bank Cards, Checks, and Phone/subway cards:

Easy Fraud
Little Privacy
**Off-line Electronic Cash refers to two-party payment**

- Low Communication Requirements

**By Contrast, On-line Payments Look Like This**

“OK”
Overspending: A problem with *off-line* e-cash

**Step 1:** The bad user copies his money

**Step 2:** The bad user gives copied cash to multiple people
The Bank is aware of trouble only later

Techniques to Contain Over-Spending

- Use tamper-resistant hardware to prevent over-spending (e.g., MONDEX in Europe)
- Trace over-spenders
- Blacklist over-spenders
- Put a bound on dollar-value of off-line transactions
Tamper-resistance is great – so far as it works

Resources Tradeoff