The Last Few Lectures...

Secret sharing:

• How to get two or more parties to share a secret in such a way that each individual cannot recover the secret from their share

Zero-knowledge protocols:

• How to get a party to prove to another that she knows a secret without revealing that secret

Today:

• How to compute with secrets
Oblivious Transfer

Suppose Alice has two messages $m_0$ and $m_1$

- Suppose Bob has a bit $b$ (= 0 or 1)
- Bob wants to have $m_b$

Constraints:
- Bob does not want Alice to know $b$
  - (Or, equivalently, which $m_b$ he wants)
- Alice does not want Bob to know both $m_0$ and $m_1$
1-2 Oblivious Transfer

(RSA-based version)

Alice generates an RSA key mod N  (public e, private d)

A  

msgs m₀, m₁  

B  

bit b
1-2 Oblivious Transfer

Alice generates an RSA key mod N (public e, private d) (RSA-based version)

A
msgs m₀, m₁
random x₀, x₁

N, e, x₀, x₁

B
bit b
1-2 Oblivious Transfer

(RSA-based version)

Alice generates an RSA key mod $N$  
(public $e$, private $d$)

Alice receives $N, e, x_0, x_1$ from Bob

Bob sends $b$ to Alice

random $k$

$q = k^e + x_b \pmod{N}$
1-2 Oblivious Transfer

(RSA-based version)

Alice generates an RSA key mod N  (public e, private d)

A
msgs m₀, m₁
random x₀, x₁

\[ t₀ = m₀ + (q-x₀)^d \]
\[ t₁ = m₁ + (q-x₁)^d \]

B
bit b
random k

q = \( k^e + x_b \) (mod N)
1-2 Oblivious Transfer

Alice generates an RSA key mod $N$ (public $e$, private $d$)

Bob computes $t_b-k$ ($= m_b$)

(A) msgs $m_0$, $m_1$
random $x_0$, $x_1$

$t_0 = m_0 + (q-x_0)^d$
$t_1 = m_1 + (q-x_1)^d$

(B) bit $b$
random $k$
$q = k^e + x_b \pmod{N}$
1-N Oblivious Transfer

• Alice has N values
• Bob has an index i
• Bob wants to get i-th value without Alice learning i
• Alice wants Bob to get only one value out of N

Related to private information retrieval

• Part of some databases’ privacy requirement
K-N Oblivious Transfer

- Alice has $N$ values
- Bob wants to get $K$ of those values without Alice learning which
- Alice wants Bob to get only those $K$ values

Two possibilities:
- messages requested simultaneously (non-adaptive)
- messages requested sequentially (adaptively)
  - can depend on previous requests
The Millionaires Problem

(Area A Yao, 1982)

Alice and Bob are both millionaires

• Alice has $I$ million dollars
• Bob has $J$ million dollars
• Alice and Bob both want to know who is richer
• But they don’t want the other to know how much money they have
• For simplicity, assume $1 \leq I, J \leq 4$
The Protocol

(RSA-based version)

Alice generates an RSA key mod N (public e, private d)

A
I

B
J
The Protocol

(RSA-based version)

Alice generates an RSA key mod N (public e, private d)
The Protocol

Alice generates an RSA key mod N (public e, private d)

\[ C = x^e \pmod{N} \]
The Protocol

(RSA-based version)

Alice generates an RSA key mod N (public e, private d)

A

I

M/2-bits random prime P

B

N, e

J

M-bits random x

C = x^e (mod N)

C-J+1 (mod N)

P, Z_1, ... Z_I,

Z_{I+1}+1, ..., Z_{4+1}
The Protocol

Alice generates an RSA key: $N$, public $e$, private $d$

$P$, random prime $P$

$C = x^e \pmod{N}$

$Z_1 = (C-J+1)^d \pmod{P}$

$Z_2 = (C-J+2)^d \pmod{P}$

$Z_3 = (C-J+3)^d \pmod{P}$

$Z_4 = (C-J+4)^d \pmod{P}$
The Protocol

(Alice generates an RSA key mod N)  (RSA-based version)

\[ N, e \]

\[ C-J+1 \pmod{N} \]

\[ P, Z_1, \ldots, Z_I, \]
\[ Z_{I+1}+1, \ldots, Z_{4+1} \]

\[ B \]

\[ J \]

M-bits random \( x \)

\[ C = x^e \pmod{N} \]

Bob receives

\[ P, R_1, \ldots, R_4 : \]

If \( R_J = x \mod P \)

then \( I \geq J \) (o/w I < J)
Secure Multiparty Computation

Given a publicly known function $F$ of $N$ inputs and producing $N$ outputs

$F(x_1,\ldots,x_n) = (y_1,\ldots,y_n)$

Suppose $N$ parties, each party $i$ with a private value $a_i$

- Goal: compute $F(a_1,\ldots,a_n) = (r_1,\ldots,r_n)$
- Each party $i$ wants to know $r_i$
- No party want others to learn their private value
Secure Multiparty Computation

Oblivious Transfer as a secure multiparty computation:

- Function $F(<m_0,m_1>,b) = (\text{nil},m_b)$
  - Alice has $<m_0,m_1>$, Bob has $b$
  - Bob wants $m_b$ (don’t care about what Alice wants)

Millionaires Problem as a secure multiparty computation:

- Function $F(I,J) = (\text{Alice},\text{Alice})$ if $I \geq J$
  - $= (\text{Bob},\text{Bob})$ if $I < J$
  - Alice has $I$, Bob has $J$
  - Alice and Bob want to know who’s richer
Other Examples

Statistical analyses with data stored across multiple databases

- Each database may be proprietary
- I.e., models of organic compounds across various bio-companies

Elections without a trusted third party

- Each elector gives his vote as input
- The function computed is vote tabulation (whatever it is)