Secure Multiparty Computations

CS 6750 Lecture 11

December 3, 2009

Riccardo Pucella

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 m_0, m_1

B

bit b

1-2 Oblivious Transfer

(RSA-based version)

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

A	N, e, x_0, x_1 $\longrightarrow$	B
msgs m_0, m_1		bit b
random x_0, x_1		

1-2 Oblivious Transfer

(RSA-based version)

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

A		B
	N, e, x_0, x_1 →	
msgs m_0, m_1		bit b
random x_0, x_1	← q	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		B
	$\xrightarrow{N,\ e,\ x_0,\ x_1}$	
msgs m_0, m_1		bit b
random x_0, x_1	$\xleftarrow{q}$	random k
		$q = k^e + x_b \pmod N$
$t_0 = m_0 + (q - x_0)^d$		
$t_1 = m_1 + (q - x_1)^d$	$\xrightarrow{t_0,\ t_1}$	

1-2 Oblivious Transfer

(RSA-based version)

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

A		B
	N, e, x_0, x_1 →	
msgs m_0, m_1		bit b
random x_0, x_1	← q	random k
		$q = k^e + x_b \pmod N$
$t_0 = m_0 + (q - x_0)^d$	t_0, t_1 →	
$t_1 = m_1 + (q - x_1)^d$		Bob computes
		$t_b - k$
		$(= m_b)$

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

(Andrew 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　　　　　　　　　　　　　　　　　　　　B

I　　　　　　　　　　　　　　　　　　　　J

The Protocol

(RSA-based version)

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

A

B

$$A \xrightarrow{N,\ e} B$$

I

J

The Protocol

(RSA-based version)

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

A **B**

N, e →

I J

M-bits random x

$C = x^e \pmod N$

← $C-J+1 \pmod N$

The Protocol

(RSA-based version)

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

A		B
	N, e $\longrightarrow$	
I		J
	C–J+1 (mod N) $\longleftarrow$	M-bits random x
		$C = x^e \pmod N$
M/2-bits random prime P	$P, Z_1, \ldots Z_I, Z_{I+1}+1, \ldots, Z_4+1$ $\longrightarrow$	

The Prot

$$Z_1 = (C-J+1)^d \pmod P$$
$$Z_2 = (C-J+2)^d \pmod P$$
$$Z_3 = (C-J+3)^d \pmod P$$
$$Z_4 = (C-J+4)^d \pmod P$$

Alice generates an RSA k

A

N

I

$C-J+1$

om x

(mod N)

M/2-bits random prime P

$P, Z_1, \ldots Z_I,$
$Z_{I+1}+1, \ldots, Z_4+1$

The Protocol

(RSA-based version)

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

A		**B**
N, e $\longrightarrow$	
I		J
$\longleftarrow$ $C-J+1 \pmod N$	M-bits random x $C = x^e \pmod N$
M/2-bits random prime P	$P, Z_1, \ldots Z_I,$ $Z_{I+1}+1, \ldots, Z_4+1$ $\longrightarrow$	Bob receives $P, R_1, \ldots, R_4$: If $R_J = x \bmod P$ then $I >= J$ (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 $F(<m_0,m_1>,b) = (nil,m_b)$
  - 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 $F(I,J) = (Alice,Alice)$ if $I>=J$

  $= (Bob,Bob)$ 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)