Introduction to Cryptography
CS 355

Lecture 36

Secure Multiparty Computation and Commitment Schemes

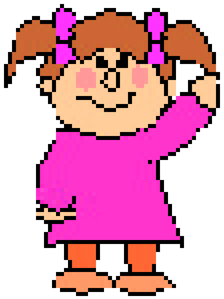
Secure Function Evaluation

- Also known as Secure Multiparty Computation
- 2-party SFE: Alice has x, Bob has y, and they want to compute two functions $f_A(x,y)$, $f_B(x,y)$. At the end of the protocol
  - Alice learns $f_A(x,y)$ and nothing else
  - Bob learns $f_B(x,y)$ and nothing else
  - Scrambled Circuit Protocol due to Andrew Yao
- n-party SFE: n parties each have a private input, and they join compute functions

Oblivious Transfer

- 1 out of 2 OT
  - Alice has two messages x_0 and x_1
  - At the end of the protocol
    - Bob gets exactly one of x_0 and x_1
    - Alice does not know which one Bob gets

Bellare-Micali 1-out-2-OT protocol



$x_0,\ x_1$

$c \leftarrow_R G_q$

g: generator of G_q

$b \in \{0,1\}$

$k \leftarrow_R Z_q$

$PK_b = g^k$

$PK_{1-b} = c/g^k$

c →

← $PK_0,\ PK_1$

$C_0 = [g^{r0},\ H(PK_0^{\,r0}) \oplus x_0]$,

$C_1 = [g^{r1},\ H(PK_1^{\,r1}) \oplus x_1]$ →

decrypts $C_b = [v_1, v_2]$ by computing $H(v_1^{\,k}) \oplus v_2$

Secret Sharing

- t-out-of-n secret sharing
  - divides a secret s into n pieces so that any t pieces together can recover n
- How to do n-out-of-n secret sharing?
- Shamir's secret sharing scheme
  - secret $s \in Z_p$
  - pick a random degree t-1 polynomial $f \in F_p[x]$ s.t. f(0)=s
  - user i gets $s_i = f(i)$
  - t users can interpolate f and find out b
  - t-1 shares reveal no information about s

Proactive Secret Sharing

- Suppose that s is shared in t-out-of-n
- User i has s_i=f(i)
- Proactive updates:
  - user 1 picks random degree t-1 polynomial s.t. g(t)=0
  - user 1 sends y_j=g(j) to user j
  - user j does $s_j^{new}=s_j^{old}+y_j$

BGW n-party SFE

- Use algorithmic circuits where operations are + and $\times$
- Each private input is shared among all participants
- Do computation with the shared value
  - e.g., given x and y both are shared by n parties, compute the shares of x+y and $x \times y$
- Secure when the majority of the parties are honest

Commitment schemes

- An electronic way to temporarily hide a value that cannot be changed
  - Stage 1 (Commit)
    - Sender locks a message in a box and sends the locked box to another party called the Receiver
  - State 2 (Reveal)
    - the Sender proves to the Receiver that the message in the box is a certain message

Security properties of commitment schemes

- Hiding
  - at the end of Stage 1, no adversarial receiver learns information about the committed value
- Binding
  - at the end of State 1, no adversarial sender can successfully reveal two different values in Stage 2

Commitment Using One-way Hash Functions

- Insecure version:
  - to commit to x, sender sends H(x) to the receiver
  - to open, sender sends x to the receiver
  - Why is this insecure?

- Improved version:
  - to commit to x, sender chooses a random string r, and send H(r || x) to receiver
  - to open commitment, sender sends r, x to receiver

The Pederson Commitment Scheme

- Public parameters: (p,g,h)
  - p: large prime (1024 bit)
  - g: a generator
  - h: another element such that $\log_g h$ is unknown
- Protocol
  - To commit to x, committer chooses random r and sends ($g^x h^r$ mod p) to the receiver.
  - To open, the committer sends x and r to the receiver
- Benefits:
  - one can prove many things about the committed value without opening it

Coming Attractions …

- Final Exam!