Advanced Tools from Modern Cryptography

Lecture 13

MPC: Honest-Majority + Active Corruption

UC-Secure Information-Theoretic MPC

- MPC protocols for general functions
- With no honest-majority (e.g., GMW paradigm)
  - Information-theoretic security possible, given OT
- With Honest Majority:
  - UC-security possible (with selective abort) if < n/2 parties corrupt
  - Can even get <u>guaranteed output delivery</u> and <u>perfect security</u> if < n/3 corrupt: **BGW Protocol** (Today)

Verifiable Protocol Execution

- We already saw passive secure BGW protocol
- So need to only implement a functionality F_{VPE} which carries out the protocol on behalf of all the parties
  - Progress? Seems like we still need MPC for general functions!
    - But easier: Every variable/computation in F_{VPE} is "owned" by some party

VPE Functionality

- F_{VPE} maintains a state for each party (image), and carries out "public" instructions (sent by a majority of parties) on these images
- F_{VPE} supports:
  - Uploading a variable to one's own image. The value being uploaded is private. (The operation itself is public.)
  - An addition or multiplication within an image
  - Transferring a variable from one image to another
  - Can at any point read a variable in one's own image
- Plan for implementing F_{VPE}: Every variable will be maintained as a <u>commitment</u> by its owner to the others

Commitment

- Simply do (n,t+1) secret-sharing of the message among all the n players (e.g., degree t Shamir secret-sharing)
  - To reveal, sender broadcasts all the shares and all the parties must agree. If the broadcast shares are valid, accept reconstruction. Else abort.
  - For $n-t \geq t+1$ (i.e., $t < n/2$), honest parties' shares already define a unique secret. Corrupt parties cannot force outputting a wrong value
- Problem 1: A single corrupt party can cause abort
- Problem 2: Does not ensure that there is a valid commitment! If commitments are not just opened, but computed on, problematic.

Commitment with Guaranteed Opening

- When $t < n/3$, can prevent adversary from causing abort at any point (unless a corrupt sender refuses to commit)
- Idea: Before accepting a commitment, do consistency checks to ensure that honest players' shares do define a valid polynomial.
  - Problem: Corrupt parties can claim inconsistency with honest players' shares ("dispute")
  - Idea: Let sender resolve disputes between two parties by publishing both their shares
  - Problem: Adversary sees more information by disputing.
  - Idea: Information published is already known to the adversary

Commitment with Guaranteed Opening

- Commitment: Instead of Shamir secret-sharing the message, use a bivariate polynomial $f(x,y)$. $f(x,0)$ is the sharing of the message (with $f(0,0)$ being the message) and party P_j gets $f(i,j)$ for all i.
  - i.e., Share the shares: each party gets a share of every share
  - P_j can check that it got a degree t polynomial, $f_j(x) := f(x,j)$
  - Will require $f(i,j) = f(j,i)$

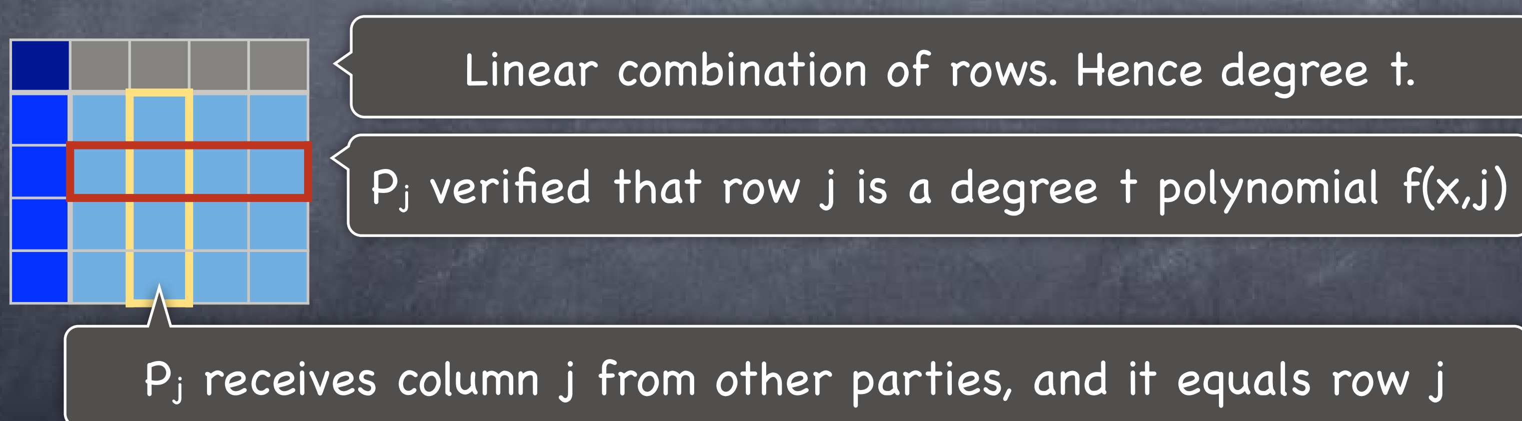
    $f(x,y) = \Sigma\ c_{p,q}\ x^p y^q$, with $c_{p,q} = c_{q,p}$ and $c_{0,0}$=msg
  - Consistency check between P_i and P_j by checking $f(i,j) = f(j,i)$. Disputing: If check fails, P_j announces $f(i,j)$ it got. Resolution by sender <u>broadcasting</u> $f(x,j)$ for P_j with whom it disagrees. (P_j assumed to update its shares using this.)
  - Repeat until no more disputes

Commitment with Guaranteed Opening

- If sender honest
  - Before any disputes, corrupt players (<t) learn nothing about the message
    - There is a bijection between sharings of m and sharings of 0, which preserves the view of the adversary
      - Consider degree t polynomial h(x) s.t. h(0)=1, and h(j)=0 for all corrupt P_j
      - Bijection maps $f(x,y)$ to $f(x,y) - m \cdot h(x)h(y)$
  - Messages revealed during dispute resolution are all messages known to the corrupt parties
  - During opening, allow sender to be inconsistent with < t players (they may be corrupt)

Commitment with Guaranteed Opening

- If sender corrupt:
  - Either sender aborts before all disputes settled,
  - Or, no dispute remaining among the honest players. Then { f(i,j) | i,j honest } is part of a valid sharing of f(0,0), and determines f(0,0) uniquely.



  - During opening, allowed sender to be inconsistent with t parties' shares. So now need remaining honest players to uniquely define the message: (n-t)-t > t, or n > 3t.

Recall VPE Functionality

- F_{VPE} maintains a state for each party (image), and carries out "public" instructions (sent by a majority of parties) on these images
- F_{VPE} supports:
  - Uploading a variable to one's own image. The value being uploaded is private. (The operation itself is public.)
  - An addition or multiplication within an image
  - Transferring a variable from one image to another
  - Can at any point read a variable in one's own image
- Plan for implementing F_{VPE}: Every variable will be maintained as a commitment by its owner to the others

A VPE Protocol

- Every variable maintained as a commitment by its owner to the others, where commitment is using the symmetric bivariate polynomial secret-sharing. Uploading: Commitment.
- Linear operations: If f, g shares of a, b, then $\alpha f+\beta g$ is a share of $\alpha a+\beta b$ (with the same dealer)
- Multiplication: Owner will send a fresh commitment of c and give a proof of $c=a\cdot b$, that can be verified collectively
    - Proof of $c=a\cdot b$: Degree d polynomials p, q with constant terms a, b, and a degree 2d polynomial r with constant term c, s.t. $p(i)\cdot q(i) = r(i)$ at 2d+1 positions. All coefficients are committed, and evaluations p(i), q(i), r(i) are computed (using linear operations) and revealed to party P_i. d=t+1 to keep a,b secret.

A VPE Protocol

- Every variable maintained as a commitment by its owner to the others, where commitment is using the symmetric bivariate polynomial secret-sharing. Uploading: Commitment.

- Linear operations: If f, g shares of a, b, then $\alpha f+\beta g$ is a share of $\alpha a+\beta b$ (with the same dealer)

- Multiplication: Owner will send a fresh commitment of c and give a proof of $c=a\cdot b$, that can be verified collectively

- Transfer: To transfer a committed variable a from P_i to P_j, P_i opens it to P_j and P_j recommits it and P_i, P_j cooperate to <u>prove equality</u>

  - To prove values a, b committed by P_i, P_j are equal, they commit to (identical) degree t polynomials p, q with constant terms a, b respectively, and open p(k),q(k) to P_k who checks p(k)=q(k)

Broadcast

- Our protocol relied on broadcast to ensure all honest parties have the same view of disputes, resolution etc.
- Concern addressed by broadcast: a corrupt sender can send different values to different honest parties
- Broadcast with selective abort can be implemented easily, even without honest majority
  - Sender sends message to everyone. Every party cross-checks with everyone else, and aborts if there is any inconsistency.
- If corruption threshold $t < n/3$, then it turns out that broadcast with guaranteed output delivery can be implemented
- If broadcast given as a setup, can do MPC with guaranteed output delivery for up to $t < n/2$