

Lecture 16:

Distributing Secrets
&
Computing on Secrets

Problems: Centralized Secrets
& Centralized Authorities

- Any single computer can be broken into and information can be spied on
- Any single hardware component cannot be trusted to keep secrets
- Any single CA can be corrupted

How to Distribute Trust

- Distribute Information
 - No single point has all the information
 - Never store entire key in one place
- Distribute Power
 - No single point should be able to perform cryptographic computations on its own such as sign, decrypt, etc.

How to Share a Secret

- Secret $k < \text{Bound}$
- Share k among n people $P_1 \dots P_n$ such that:
 - Together they can recover k
 - Less than n people have no information on k
- SUM-sharing (k) :
 - Choose prime $p > \text{Bound}$
 - Choose at random x_1, \dots, x_{n-1} in Z_p
 - Set $x_n = k - (x_1 + \dots + x_{n-1}) \bmod p$
 - Give P_i share x_i
- Claim: Any set of $n-1$ shares gives no information on k .
- Proof: $\forall x, \text{Prob}(k = x) = 1/p$
(probability taken over random choices of the x_i 's)

t-out-n Secret Sharing

- Threshold Secret Sharing
 - Secret $k < \text{Bound}$
 - Shared among n players s.t.
 - t shares can be used to reconstruct k
 - $< t$ shares give no information about k
- To see how, lets recall some properties of polynomials

Representing a polynomial

- A degree- t polynomial is represented by its $(t+1)$ coefficients:
- $P(x) = c_t x^t + c_{t-1} x^{t-1} + \dots + c_1 x^1 + c_0$
- Coefficients: c_t, c_{t-1}, \dots, c_0

Polynomials over Finite Fields

- In principle, could work over any set with addition, multiplication, division defined.
- We will work with polynomials over Z_p
- Let $P(x) = c_0 + c_1x + c_2x^2 + \dots + c_{t-1}x^{t-1} \pmod p$
- All coefficients are from Z_p
- All arithmetic (+, *, -, \) is mod p
- For the rest of this lecture, sometimes will omit the mod p

The roots of a polynomial

$$P(x) = a_t x^t + a_{t-1} x^{t-1} + \dots + a_1 x^1 + a_0$$

Definition: r is a "root" of $P(x)$ if $P(r) = 0$

E.g. $P(x) = x^2 - 2x + 1$ roots = 1, 1

Theorem: Every degree t polynomial has at most t roots.

Proof: By induction on the degree

Corollary: $\forall P, Q$ polynomials of degree t s.t. $P \neq Q$,
 $P(a) = Q(a)$ for at most t distinct a 's.

Proof: $R(x) = P(x) - Q(x)$ is a t -degree polynomial.

$P(a) = Q(a) \Rightarrow R(a) = 0$, but R has at most t roots.

Alternative Representation of Polynomials

- Let $c_i \in \mathbb{Z}_p$, $P(x) = c_0 + c_1x + c_2x^2 + \dots + c_{t-1}x^t \pmod p$
- Theorem: Any $t+1$ pairs (a_i, b_i) s.t. $b_i = P(a_i)$ s.t. $a_i \neq a_j \ \forall i \neq j$ uniquely determine a polynomial of degree t
- Theorem: \exists an efficient procedure to compute coefficients c_i 's of P , from $t+1$ pairs (a_i, b_i) s.t. $b_i = P(a_i)$
- One method is Lagrange Interpolation

Lagrange Interpolation



- Let (a_i, b_i) s.t. $b_i = P(a_i)$ s.t. $a_i \neq a_j \ \forall i \neq j$ for $i=1, \dots, t+1$
- Let $\lambda_i(x) = \prod_{j \neq i} (x - a_j)(a_i - a_j)^{-1} \pmod p$
- Set $P(x) = \sum_{i=1}^{t+1} b_i \lambda_i(x)$
- Lemma: P is the unique t -degree polynomial such that $b_i = P(a_i)$ for all $i=1, \dots, t+1$

Two different representations

$P(x) = c_t x^t + c_{t-1} x^{t-1} + \dots + c_1 x^1 + c_0$
can be represented either by

1. $t+1$ coefficients

$c_t, c_{t-1}, \dots, c_2, c_1, c_0$

or

2. Its value at any $t+1$ points

$P(x_1), P(x_2), \dots, P(x_t), P(x_{t+1})$

- (e.g., $P(0), P(1), P(2), \dots, P(t+1)$.)

Secret Sharing

- Missile has random secret number S encoded into its hardware. It will not arm without being given S .
- n officers have memorized a private, individual "share".
- Any k out of n of them should be able to assemble their shares so as to obtain S .
- Any $\leq k-1$ of them should not be able to jointly determine any information about S .

t-out-of-n secret sharing scheme

- Let k be a random "secret" from Z_p
- Want to give shares S_1, S_2, \dots, S_n to the n players such that:
 - a) Given t of the S_i 's, can find out k .
 - b) Given $t-1$ S_i 's, any secret k is equally likely to have produced this set of S_i 's.

Shamir's: t-out-of-n Secret Sharing

Let k be a random "secret" from Z_p

Dealer:

- Picks $t-1$ random coefficients R_1, R_2, \dots, R_{t-1} from Z_p
- Let $P(x) = R_{t-1} x^{t-1} + R_{t-2} x^{t-2} + \dots + R_1 x^1 + k$
- $\forall 1 \leq j \leq n$, dealer gives player j 's his share $S_j = P(j)$

Fact:

- Reconstruction: Using Lagrange interpolation, can compute $P(0)=k$ from any t shares.
- Any $t-1$ shares give no information on $P(0)$

Problem 1: Faulty Players

- What if $t-1$ of the players cheat during reconstruction and give wrong share s.t. from there is no unique t -degree polynomial that fits S_1, \dots, S_n
- Claim: As long as fewer than $(n-t)/2$ corruptions, then can reconstruct the original polynomial $P(x)$
- Without finding out which S_i 's are corrupted.
- Corollary: If $n \geq 3t$ then can tolerate $t-1$ corrupted players during reconstruction

Problem 2: Faulty Dealer

- What if dealer cheats and gives n shares such that there is no unique $(t-1)$ -degree polynomial which fits the shares
- **t -out- n Verifiable Secret Sharing:**
 - Can detect faulty dealer
 - Robust: reconstruction can be accomplished as long as at most $t-1$ faulty players
 - No $t-1$ players can reconstruct secret.
- Feldman: Under Discrete Log Assumption
- Pederson: Information Theoretically Private

Feldman: t-out-n Verifiable Secret Sharing

- Pick prime $p=2q+1$, generator g s.t. $\text{Order}(g)=q$, let secret $s \in \mathbb{Z}_q$
- Dealer(s):
 - Pick random degree t polynomial $P(x) = \sum_{j=0}^{t-1} R_j x^j \in \mathbb{F}_q$ s.t. $P(0)=s$
 - Broadcast $A_j = g^{R_j} \pmod p$, $j=0, \dots, t-1$
 - \forall player i , send share $S_i = P(i)$
- \forall player i , verify $g^{S_i} = A_0 \{A_1\}^i \{A_2\}^{i^2} \dots \{A_{t-1}\}^{i^{t-1}} \pmod p$
 - If not, player i broadcasts a complaint, let $C = C \cup \{i\}$, dealer broadcasts S_i for complainer.
 - If more than t complaints, Dealer is disqualified.
- Reconstruction: Each player $i \notin C$, broadcasts its share S_i ; Players interpolate (i, S_i) ; let $s = P(0)$

Pederson: Verifiable Secret Sharing:

- Pick prime $p=2q+1$, generator g s.t. $\text{Order}(g)=q$, let secret $s \in \mathbb{Z}_q$, let $h = g^s \pmod p$
- Dealer(s):
 - Pick two random degree t polynomial $P(x) = \sum_{j=0}^{t-1} R_j x^j \in \mathbb{F}_q$ s.t. $P(0)=s$
 - $P'(x) = \sum_{j=0}^{t-1} R'_j x^j \in \mathbb{F}_q$ s.t. $P'(0) = \text{random}$
 - Broadcast $A_j = g^{R_j} h^{R'_j} \pmod p$, $j=0, \dots, t-1$
 - \forall player i ,
 - receives $S_i = P(i)$, $S'_i = P'(i)$,
 - check that $g^{S_i} h^{S'_i} = A_0 \{A_1\}^i \{A_2\}^{i^2} \dots \{A_{t-1}\}^{i^{t-1}} \pmod p$
 - If not, broadcasts a complaint, let $C = C \cup \{i\}$, dealer broadcasts S_i and S'_i .
 - If more than t complaints, Dealer is disqualified.
- Reconstruction: Each player $i \notin C$, broadcasts its share S_i ; Players interpolate $(i, S_i) \forall i$; let $s = P(0)$

More Facts about Polynomials

- Let $P(x), Q(x)$ be two polynomials of degree t
- The sum $R(x) = P(x) + Q(x)$ is a polynomial
 - R is degree t
 - $R(0) = P(0) + Q(0)$
 - $R(i) = P(i) + Q(i)$
- The product $R(x) = P(x)Q(x)$ is a polynomial
 - R is degree $2t$.
 - $R(0) = P(0)Q(0)$
 - $R(i) = P(i)Q(i)$
- Let $P'(x) = P(x) + R(x)$ for R random polynomial of degree t s.t. $R(0) = 0$. Then, P' is a random polynomial of degree t s.t. $P'(0) = P(0)$.

Mobile Adversary

- Suppose there exist t faulty players whose identity changes each time step.
- Proactive Secret Sharing:
 - Deal secret as in Shamir's secret sharing
 - Refresh shares each round, keeping the secret the same. How?
 - \forall round $j \forall i$ player i choose random polynomial R_i of degree t , $R_i(0) = 0$, sends player k , $S_{ik} = R_{ij}(k)$. Player k sets new $S_k = S_k + \sum_i \{S_{ik}\}$
 - Now: the new points (k, S_k) define a new random t -degree P polynomial s.t. $P(0) =$ original secret.
- Verifiable Secret Sharing changes accordingly

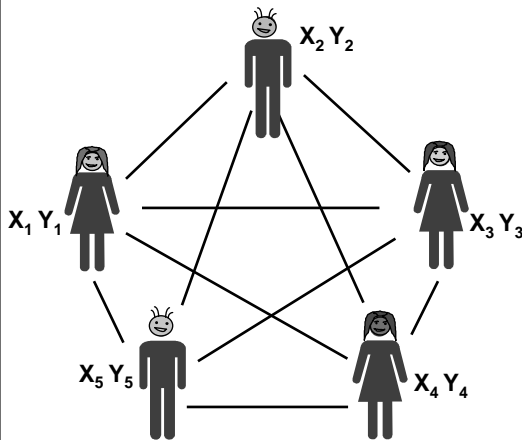
Distributed Computing on Shares

- P_x denotes random t -degree polynomial s.t. $P_x(0)=x$
- Input Stage: dealer SS's x_1, \dots, x_m among the n players (i.e $\forall i, j$: player j gets $P_{x_i}(j)$).
- Computing $a+b$:
 - $\forall j$, player j : SS's $\{R^j\}_0$ among the n players,
 - $\forall i$, player i : sets $P_{a+b}(i) = P_a(i) + P_b(i) + \sum_j R^j_0(i)$
- Computing $a*b$:
 - $\forall j$, player j : SS's $\{R^j\}_0$ among the n players,
 - $\forall i$, player i : sets $P_{a+b}(i) = P_a(i) * P_b(i) + \sum_j R^j_0(i)$
 - Problem: degree of polynomial is $2t$
 - Recall: $(i, P_{a*b}(i))$ for $i=1, \dots, t+1$ define t degree polynomial $P'_{a*b}(x)$ s.t. $P'_{a*b}(0)=a*b$.
 - Lagrange: $P'_{a*b}(j) = \sum_i \lambda_i(j) P_{a*b}(i)$.
- More generally: Any Arithmetic circuit over F_n

Distributed Computing on Shares

- P_x denotes random t -degree polynomial s.t. $P_x(0)=x$
- Input Stage: dealer SS's a, b among the n players (i.e $\forall i, j$: player j gets $P_a(j), P_b(j)$).
- Computing $a+b$:
 - $\forall j$, player j : SS's $\{R^j\}_0$ among the n players,
 - $\forall i$, player i : sets $P_{a+b}(i) = P_a(i) + P_b(i) + \sum_j R^j_0(i)$
- Computing $a*b$:
 - $\forall j$, player j : SS's $\{R^j\}_0$ among the n players,
 - $\forall i$, player i : sets $P_{a+b}(i) = P_a(i) * P_b(i) + \sum_j R^j_0(i)$
 - Problem: degree of polynomial is $2t$
 - Recall: $(i, P_{a*b}(i))$ for $i=1, \dots, t+1$ define t degree polynomial $P'_{a*b}(x)$ s.t. $P'_{a*b}(0)=a*b$.
 - Lagrange: $P'_{a*b}(j) = \sum_i \lambda_i(j) P_{a*b}(i)$.
- More generally: Any Arithmetic circuit over F_n

Multi Party Secure Computation

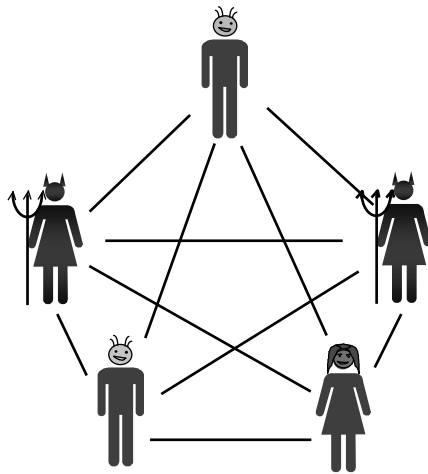


Each user/player has private information X_i

Want to do global computations keeping X_i private

$$(Y_1 \dots Y_n) = g(X_1 \dots X_n)$$

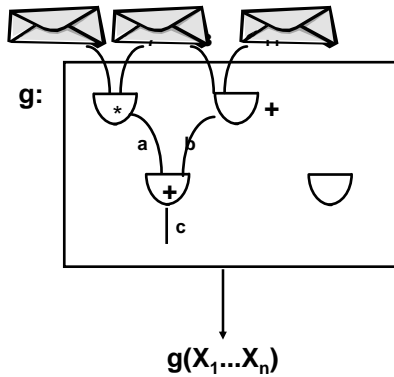
Who is the Adversary ?



Subsets of colluding players:

- Honest but Curious
- Malicious
- Mobile
- Globally coordinated attacks

Overview of General Method



Computing any function g is the combination of simple computations

*

- Input: each player secret shares his input
- Computation: Compute gate by gate
- Output: Reconstruct output

(n,t) - Threshold Cryptosystems

- n parties share the ability to perform cryptographic operations
- any t out of n can perform operation jointly
- Any $t-1$ cannot perform operation
- Example Cryptographic operations:
 - Digital Signatures
 - Decryption

(n,t)- threshold signatures

- Share secret key among n CA's
- $\geq t$ can sign
- $< t$ can not sign
- $< t$ cannot cause denial of service

- Example: DSS

Recall: El Gamal Signature Scheme

- Key Generation: On security parameter k choose $p=2q+1$, $g, g^x \pmod p$. The secret signing key is x and the public verifying key is (p,g,y) .
- Signing: output $\sigma=(r,s)$ s.t. $0 \neq r, s \neq p-1$ and $g^m = y^r r^s \pmod p$.
 - To compute (r,s) : choose a random t s.t. $0 \neq t \neq (p-1)$ and $\gcd(t, p-1) = 1$. Let $r = g^t \pmod p$ and s s.t. $g^m = y^r r^s = g^{xr+ts} \pmod p$.
- Verifying: Given public key (p,g,y) , signature (r,s) and message m , output 1 iff $g^m = y^r r^s \pmod p$.

Slight change: take $r = g^{t^{-1}} \pmod p$ and then $t(m-st) = s \pmod q$,

Distributed El Gamal Signatures

- **Generating a Secret Key:**
 - Player i : chooses a random $z_i \in \mathbb{Z}_q$
 - Party i : Feldman VSS's z_i broadcasting $A_j = g^{R_j}$ s.t. $P_{z_i}(x) = \sum_j R_j x^j \pmod p$ & giving player k , $P_{z_i}(k)$
 - Secret key $x = \sum_i z_i \pmod q$
- **Generating Matching Public Key $y = g^x \pmod p$:**
 - Player i broadcasts $y_i = g^{z_i} \pmod p$
 - All players verify that $\forall i, y_i = \prod_j (A_j)^{z_j} \pmod p$
 - Public key $y = \prod_i y_i \pmod p$
- **Signing m : Player i chooses t_i and Feldman VSS's t_i . Now, $r = g^{t_i^{-1}} \pmod p$, $s = (m + xr)t \pmod q$, but how can players share s , so that no player knows how to compute s or $t \pmod q$?**

Distributed El Gamal Signatures

- **Signing m : Player i chooses t_i and Feldman VSS's t_i . Now, $s = (m + xr)t \pmod q$, but how can players share s , so that no player knows how to compute s or $t \pmod q$?**
- **Idea for sharing $t^{-1} \pmod q$:**
 - Players will jointly generate a shared random secret $a \in \mathbb{Z}_q$, and 0 . Namely, player i has $a_i = P_a(i)$ and $b_i = R_0(i)$.
 - Players reconstruct $\mu = ta$ by broadcasting the values $\mu_i = t_i a_i + b_i$ and interpolating for μ . Then, each player let its share of t^{-1} , $u_i^{-1} = \mu^{-1} a_i \pmod q$
- **Idea for sharing $s \pmod q$:**
 - Player i broadcasts $s_i = t_i(m + x_i r) + R'_0(i) \pmod q$