CIS 700 Lecture 19: Secret Sharing

Problem: Nuclear codes.

Ensure 2 people present to enter code.

Soln: Give shares $x_1 = r$, $x_2 = r \oplus c$ $\quad x_1 \oplus x_2 = c$

x_1, x_2 uniformly random on own. c = actual code.

What about 3 people, all must be present?

Soln: $x_1 = r_1 \quad x_2 = r_2, \quad x_3 = c \oplus r_1 \oplus r_2 \quad x_1 \oplus x_2 \oplus x_3 = c$

x_1, x_2, x_3 random

$x_1 \oplus x_2, x_2 \oplus x_3, x_1 \oplus x_3$ random

What if only 1 person present?

$x_1 = x_2 = x_3 = c$

What if want 2 of 3 people present?

"threshold secret sharing"

Polynomial Interpolation:

Degree d polynomial uniquely determined by $d+1$ points (x_i, y_i)

Work over $\mathbb{F}_p$, $d < |\mathbb{F}_p|$.

Shamir secret sharing

Share secret s into n shares, recover with k / n shares.

Choose $p > n$.

1. Choose coeffs $f_1, \ldots, f_{k-1} \in \mathbb{F}_p$
2. Construct polynomial $f_{k-1} x^{k-1} + f_{k-2} x^{k-2} + \ldots + f_1 x + s = f(x)$
3. Handout secret shares $(x_1, f(x_1))\ (x_2, f(x_2)), \ldots, (x_n, f(x_n))$

   $(1, f(1)), (2, f(2)), \ldots (n, f(n))$

   ↓ Share 1 ↓ share 2 ↓ share n

Recovering shares:

Polynomial interpolation: Reconstruct unique polynomial passing through (x_i, y_i) for $\geq k$ points

(2)

Polynomial Interpolation = CRT

$f(x_i) = y_i \iff f(x) \equiv y_i \bmod (x - x_i)$

$\Leftarrow f(x) \equiv y_i \bmod (x - x_i)$

$f(x) \equiv y_i + h(x)(x - x_i)$

$f(x_i) = y_i + h(x_i)\cdot(x_i - x_i)$

$\Rightarrow f(x) \equiv y' + h(x)(x - x_i)$ (remainder mod $x - x_i$ = constant)

$f(x_i) = y' \Rightarrow y' = y_i$

$f(x) = y_1 \bmod (x - x_1)$
$f(x) = y_2 \bmod (x - x_2)$
$\vdots$
$f(x) = y_k \bmod (x - x_k)$

} CRT says unique solution

$f(x) \bmod \prod_i (x - x_i)$

$x^k + \dots$

$\Rightarrow f(x)$ uniquely determined $f_{k-1}x^{k-1} + \dots + f_0$

Constructive CRT reconstruction (Lagrange Interpolation)

Recall:

$x \equiv x_1 \bmod p_1$
$\vdots$
$x \equiv x_k \bmod p_k$

Construct b_i: $b_i \equiv 1 \bmod p_i$, $b_i \equiv 0 \bmod p_j$ $j \neq i$

Then $x = \sum x_i b_i$

Set $N_i = \prod_{i \neq j} p_i$, $q_i = N_i^{-1} \bmod p_i$, $b_i = N_i q_i$

Here: $p_i = (x - x_i)$

$N_i = \prod_{i \neq j} (x - x_j)$, $q_i = N_i^{-1} \bmod (x - x_i) = \prod_{i \neq j} \frac{1}{(x - x_j)} \bmod (x - x_i)$

$= \prod \frac{1}{(x_i - x_j)}$

$b_i = \prod \frac{(x - x_j)}{(x_i - x_j)}$

$f(x) = \sum_i y_i b_i = \sum_i y_i \prod_{i \neq j} \frac{(x - x_j)}{(x_i - x_j)}$

Really want $f(0) = \sum_i y_i \prod_{i \neq j} \frac{-x_j}{(x_i - x_j)}$

Pre-compute $b_i = \prod_{i \neq j} \frac{-x_j}{(x_i - x_j)}$

$\Rightarrow$ linear computation

Secrecy:

Assume opponent has $k-1$ shares. $(x_1 \dots, x_{k-1}$ obj$)$

For each candidate value (x_k, v) $v = 0 \dots p$

$\exists$ unique poly of deg k taking values $\{(x_i, y_i)\}, (x_k, v)$.

By construction we choose these uniformly at random.

$\Rightarrow$ information-theoretic security

Properties:

Shamir secret sharing homomorphic mod p:

$h(x) = f(x) + g(x)$

$h(x_i) = f(x_i) + g(x_i)$

$f(x) = f_{k-1}x^{k-1} + \dots + f_0$

$g(x) = g_{k-1}x^{k-1} + \dots + g_0$

$h(x) = (f_{k-1} + g_{k-1})x^{k-1} + \dots + (f_0 + g_0)$

$h(x) = f(x) \cdot g(x)$

$h(x_i) = f(x_i) \cdot g(x_i)$

$h(x) = f(x) \cdot g(x)$

$= (f_{k-1}x^{k-1} + \dots + f_0)(g_{k-1}x^{k-1} + \dots + g_0)$

$= h_{2k-2}x^{2k-2} + \dots + f_0 g_0$

Secure Multi-Party Computation:

"honest but curious model" = everyone follows protocol

Compute some function on inputs:

- Each party outputs f(inputs)
- No collusion of $< k$ parties can learn anything about inputs

To compute sum of c_1, c_2:

1. Secret share c_1, c_2 into k of n shares $s_{1,1}, \dots, s_{1,n}$ $s_{2,1}, \dots, s_{2,n}$
2. Each party outputs $s_{1,i} + s_{2,i}$

Product: $n \geq 2k-2$

1. Party i inputs shares $s_{1,i}, s_{2,i}$ computes $s_{1,i} \, s_{2,i} = d_i$
2. i secret shares d_i into k of n shares $t_{i,1}, \dots, t_{in}$
3. i sends share $t_{i,j}$ to party j
4. j computes $u_j = \sum t_{i,j} \cdot b_i$ $\quad b_i = \prod_{i \neq j} \frac{-x_j}{(x_j - x_i)}$ $\quad X_1 \cdot X_2 = \sum_{i=1}^{n} b_i \cdot d_i$

u_j are k of n shares of $c_1 \cdot c_2$: $c_1 \cdot c_2 = \sum_{i=0}^{2(k-1)} b_i \cdot d_i = \sum_{i=0}^{2k-2} b_i \cdot \sum_{j=0}^{k-1} b_j \cdot t_{ij} = \sum_{j=0}^{k-1} b_j \cdot \sum_{i=0}^{2(k-1)} b_i \cdot t_{ij}$

Application:

Compute any arithmetic circuit: