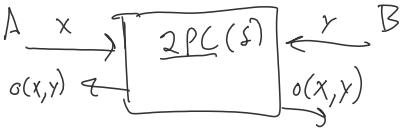
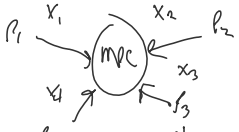
Secret Sharing

Tuesday, October 22, 2019 11:37 AM

MPC multiparty computation

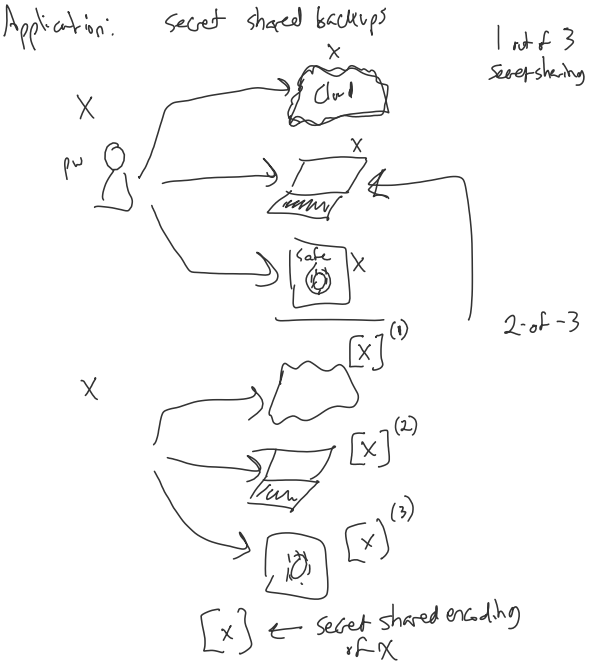




all get output $f(x_1, x_2, x_3, \ldots)$

Secret sharing

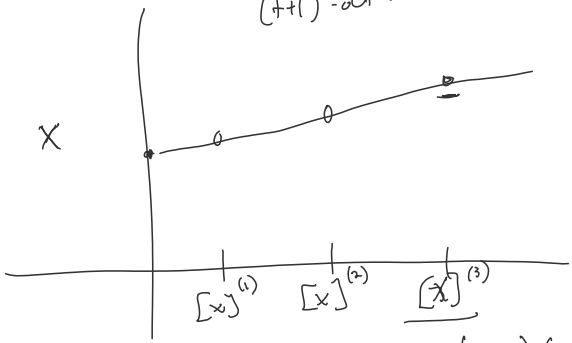
Application: secret shared backups



$[x]$ ← secret shared encoding of x

How?

Degree t polynomial for $(t+1)$-out-of-n secret sharing



- any $k-1$ shares reveal no info about x
- and any k shares can combine to decode x

Polynomials on finite fields

e.g. $f(x) = 5x^2 + 4x + 2$

$f: \mathbb{F} \to \mathbb{F}$

How many polynomials over $\mathbb{F} = \mathbb{Z}_7$?

Def'n equality for fxns: $f = g$ iff $\forall x,\ f(x) = g(x)$

2^7? $\quad |\mathbb{Z}_7|^{|\mathbb{Z}_7|} \quad 7^7$

Q. H...

A. ∞ ~~X~~

A degree-k poly is represented by... can be

$k+1$ coefficients

$f(x) = a_0 + a_1 x + \cdots a_k x^k$

Degree-bound vs. degree

~~def~~ $f(x) = \cdots 2x^2 + x + 5$ is degree-bound 8

$\cdots 0 \cdot x^{8} \cdots$

Degree-k bound

Polynomials form a group under addition

$f(x) : \quad a_0 \cdots a_k x^k$

$g(x) \quad b_0 \cdots b_k x^k$

$(g+f)(x) : (a_0+b_0) \cdots (a_k+b_k) x^k$

Do polynomials form a group under multiplication?

$(a_0 + a_1 x)(b_0 + b_2 x)$

foiling $a_0 b_0 + (a_1 b_0 x + a_0 b_1 x) + a_1 b_1 x^2$

$a^{|G|} = 1$

$x^7 = 1 \bmod 7$

$f(x) = 5x^7 + 4$

$= 5x^{14+1} + 4$

$g(x) = -2x^{k-1}$

Lagrange Interpolation.

Thm. Given any $k+1$ distinct points $(x_0, y_0), \ldots (x_k, y_k)$
we can find a degree-bound k polynomial
f s.t. $f(x_i) = y_i \quad \forall i \in 0 \ldots k$.

Lemma: Lagrange polynomials
Given $k+1$ points as above,
we can find degree-bound k polys
$p_i(x)$ s.t.

$$p_i(x_j) = \begin{cases} 1 & \text{if } i=j \\ 0 & \text{if } i \neq j \end{cases}$$

$p_i(x_0)\ p_i(x_1) \ldots \quad p_i(x_k)$

$y_0 \cdot p_0$: $y_0\ 0 \ldots 0$
$y_1 \cdot p_1$: $0\ y_1 \ldots 0$
$\vdots$
$\cdot p_k$: $0 \ldots \ldots y_k$

$$p(x) = \sum_i y_i \cdot p_i(x)$$

How to construct p_i

$$p_0(x) = \frac{(x-x_1)(x-x_2)\cdots(x-x_k)}{(x_0-x_1)(x_0-x_2)\cdots(x_0-x_k)}$$

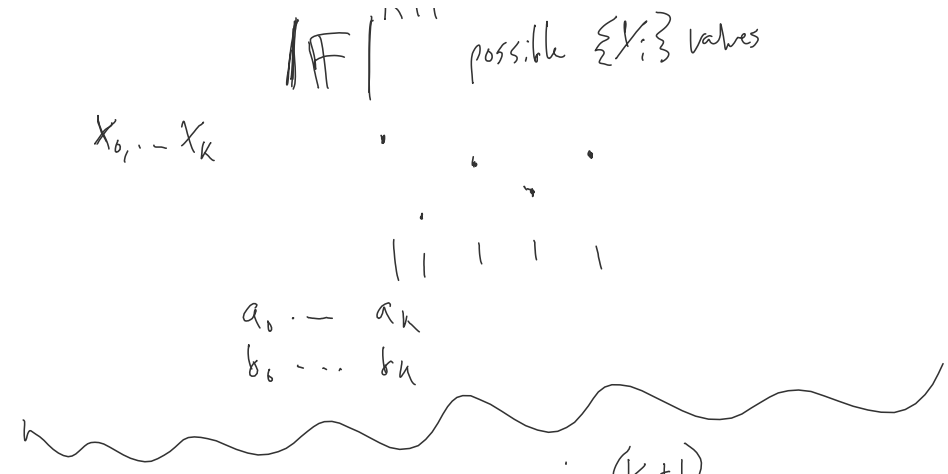
$$p_0(x_1) = p_0(x_2) = \cdots = 0$$
$$p_0(x_0) = 1$$

$$p_i(x) = \prod_{j=0, j\neq i} \frac{x - x_j}{x_i - x_j}$$

So, $p(x) = \left(\sum_i y_i \cdot \left(\prod_{j \neq i} \frac{x - x_j}{x_i - x_j} \right) \right)$

$p(x_i) = y_i$ for every $i \leq k$

$k+1$ points also represent
a degree-k polynomial
$k+1$



Claim: There are $p^{(k+1)}$ distinct degree-bound k polynomials over $\mathbb{Z}_p$.

Proof: Let $X_0, \ldots X_k$ be fixed distinct points in $\mathbb{Z}_p$

Let $Y_0, \ldots Y_k$

and $Y'_0, \ldots Y'_k$ be points in $\mathbb{Z}_p$ with at least 1 i s.t. $Y_i \neq Y'_i$.

By lagrange interp,

we can find degree k polys.

$f(x)$ s.t. $\forall i,\ f(X_i) = Y_i$

$f'(x)$ s.t. $\forall i,\ f'(X_i) = Y'_i$

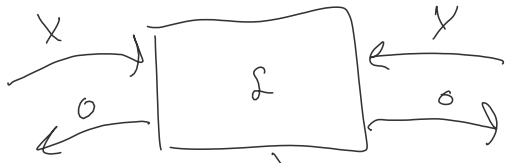
Now $f \neq f'$ because $f(X_i) \neq f'(X_i)$ for some i.

Since there are $p^{(k+1)}$ choices of $Y_0, \ldots Y_k$, there are at least $p^{(k+1)}$ polys of

degree bound k.

At most $p^{(k+1)}$ choices of $a_0, \ldots, a_k$ coefficients. ∎

Computing on S.S. data.



$o = f(x, y)$

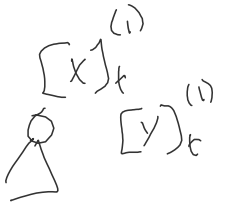
$$[o] \leftarrow MPC_f([x], [y])$$

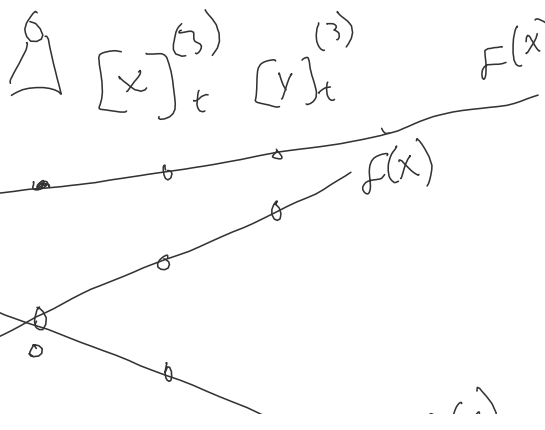
— Linear operations are trivial

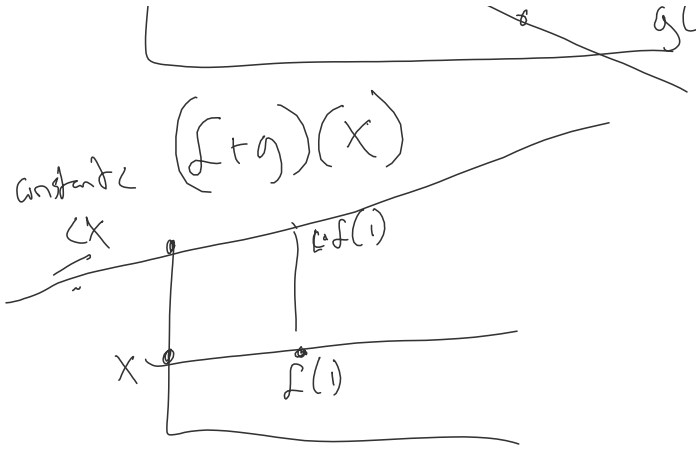
— locally computable

$$[x]_t \quad [y]_t$$

Goal $[x+y]_t$







Multiplication... harder.

$$[x]_t \cdot [y]_t = ? \; [x \cdot y]_t ?$$

$(f \cdot g)(x)$

$f(x) = a_1 x + a_0$ — $a_1 b_1 x^2 \ldots$

$g(x) = b_1 x + b_0$

Multiplication with Beaver Triples.

Input: $[x]_t$ $[y]_t$

Goal: $[xy]_t$

Assumption: (Precompute) $[a]_t, [b]_t, [ab]_t$ for uniform $b \xleftarrow{\$} \mathbb{F}_p$, $a \xleftarrow{\$} \mathbb{F}_p$

$[D]_t := [x]_t - [a]_t$

$[E]_t := [y]_t - [b]_t$

$D \leftarrow \text{open}([D]_t)$

$\forall X_1, X_2$ { $a \xleftarrow{\$} \mathbb{F}_p$, $h(\cdot) \xleftarrow{\$}$ deg. k poly s.t. $h(0) = a$, $f(\cdot), g(\cdot)$ deg. h $f(0) = X_1$ } $\sim (X_1 - h)(X)$ $\sim (X - a)$

$\vdots$

{ same, but $(X_2 - h)$

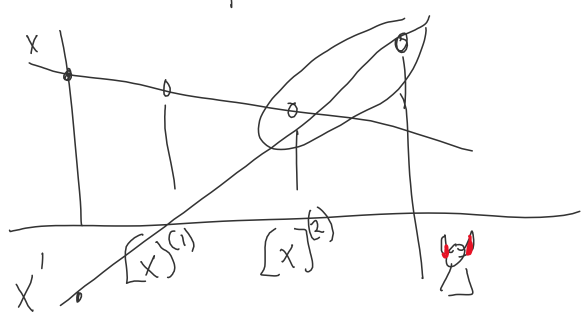
$E \leftarrow \text{Open}([E]_t)$ $\sim (y-b)$

$$[xy]_t = D\cdot[y]_t + E\cdot[x] + [ab]_t - D\cdot E$$

Why? $(x-a)\cdot y + (y-b)\cdot x + a\cdot b - (x-a)(y-b)$

$$xy = xy - ay + xy - bx + ab - xy - ab + xb + ay$$

Robust interpolation,



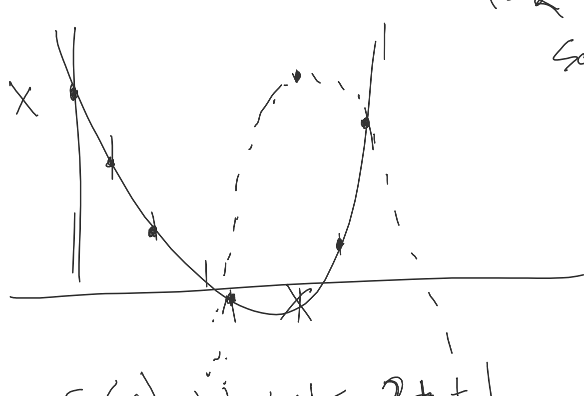
Suppose at most t parties are Byzantine (malicious, or adversary) (provide wrong values, do not follow protocol).

Wait to receive shares from parties such that $2t+1$ shares lie on a degree t polynomial.

Then the decoded polynomial is the correct one.

$t=2$

Say $n=7$



If $f(x)$ intersects $t+1$ points, at most t of which may come from corrupt parties, then $F(x)$ intersects at least $t+1$ honest parties.

These $t+1$ points uniquely determine a degree-t polynomial. So it's the correct one.

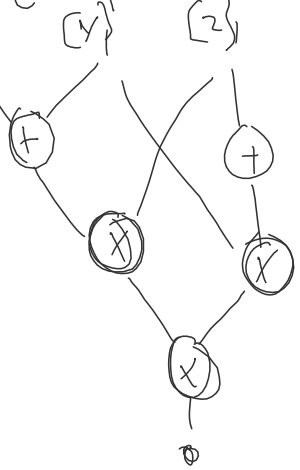
If $n \geq 3t+1$,

we can robustly decode deg-t poly's, even if t are malicious, and guarantee output

---

Arithmetic Circuits

f:



Evaluate layer by layer.

Faster protocol for squaring

Input: $[x]$

output: $[x^2]$

Precompute: $[a], [a^2]$

$$D = \text{Open}([x] - [a])$$

$$[x^2] = D^2 + 2D[x] + [a^2] - D^2$$

$$x^2 = 2(x-a)\overline{x} + a^2 - (x'-a)^2$$

$$x^2 = (2x^2) - (2ax) + (a^2) - (x^2) - (a^2) + (2ax)$$

Given 7-bit uniform random numbers, $[0,127]$

generate a random sample in $[0,100)$

$\{$ $x \xleftarrow{\$}$ 7-bit number
return $z = X \% 100$ $\}$

$\forall n.\ P_r[z = n] = \begin{cases} 2/128 & n < 28 \\ 1/128 & \text{otherwise} \end{cases}$

Better Solution: rejection sample

$$ZK\{(x, y, r', r''): \underline{m}\ C_1 = g^x h^r,\ C_2 = g^y h^{r''},\ \cancel{x - y \neq 0},\ m \cdot (x-y) = 1\}$$

$0 \in |\mathbb{Z}_p^+| = p$

$0 \notin |\mathbb{Z}_p^*| = p-1$

$\forall z \neq 0,\ \exists m \in \mathbb{Z}_p$ s.t. $m \cdot z = 1 \bmod p$

$\nexists m \in \mathbb{Z}_p$ s.t. $m \cdot z = 1 \bmod p$

$$ZK\{(x, y, r', r'', r'''): \underset{m, r''''}{\ } \bullet\ C_1 = g^x h^r \ \bullet\ C_2 = g^y h^{r'} \ \bullet\ C_3 = g^m h^{r''}\}$$

$$(C_1/C_2)^m = g h^{r'''}$$

$$(C_1/C_2)^m (h^{-1})^{(r''')} = g$$

$$(g^{x-y} h^{r-r'})^m \cdot h^{-r''''} = g$$

$$\longrightarrow g^{m(x-y)} h^{(r-r')m - r''''} = g$$

$$\bullet\ g = (C_1/C_2)^m h^{r''''}\}$$

$$ZK\{(x, r, r'): C_1 = g^x h^r,\ C_2 = g^x h^{r'}$$

$k_x \xleftarrow{\$} \mathbb{Z}_q$

$k_r \xleftarrow{\$}$

$k_{r'} \xleftarrow{\$}$

$K_1 = g^{k_1} h^{k_r}$

$K_2 = g^{k_1} h^{k_{r'}}$ $\xrightarrow{K_1, K_2}$

$\xleftarrow{c}$

$s_x = k_x + cx$

$s_r = k_r + c_r$

$s_{r'} = k_{r'} + c_{r'}$ $\xrightarrow{s_x, s_r, s_{r'}}$ $\cdot\ K_1 C_1^c = g^{s_x} h^{s_r}$

$\cdot\ K_2 C_2^c = g^{s_x} h^{s_{r'}}$

$\rightarrow\ g^{k_1} h^{k_r} \left(g^x h^r\right)^c = g^{k_x + cx} h^{k_r + cr}$

$g^{k_1 + cx} h^{k_r + cr} =$

(after running twice) $\frac{s_x - s_{x_2}}{c_1 - c_2}$

Constant round products.

Input : $[x_1], \ldots [x_k]$

Output : $[x_1 \cdot x_2 \cdot \cdots x_k]$

$= \left[\prod_i x_i\right]$

Try 1: $[x_1] \cdot [x_2]$ — Beaver mult.

$\Downarrow$

$[x_1 \cdot x_2] \cdot [x_3]$

$\Downarrow$

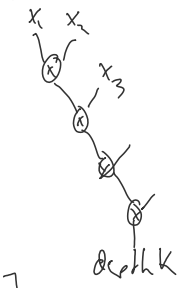
$[x_1, \ldots x_3]$

$\ddots$

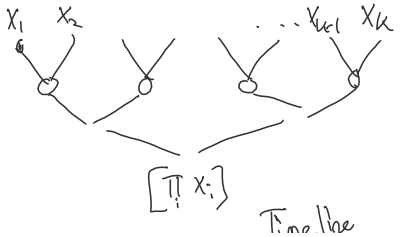
$\cdot [x_k]$

$\Downarrow$

$[\prod x_i]$ ✓

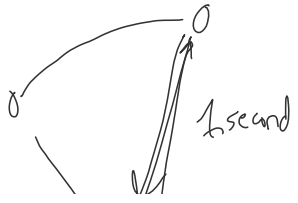


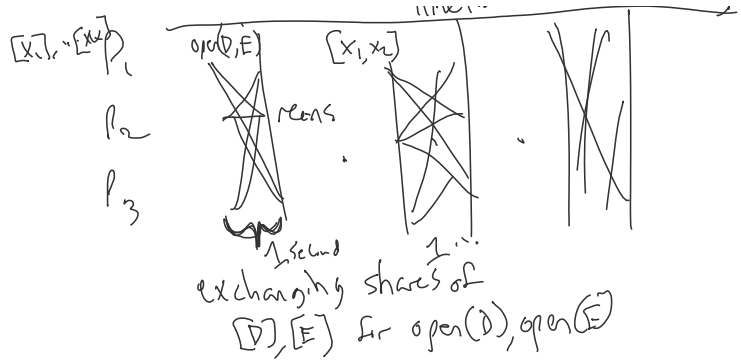
Try 2:



depth $\log k$ ✓

Timeline ——→





GC generate

encrypted gates → whole garbled circuit

all at once

Constant round

Input: $[x_1], \ldots, [x_k]$ (we know all $x_i \neq 0$)

Preompute: $[r_0], [r_1], \ldots [r_k]$

random sharings

Steps:

beaver mults

$$[a_1] := [r_0]^{-1} \cdot [x_1] \cdot [r_1]$$

$$[a_2] := [r_1]^{-1} [x_2] [r_2]$$

$$[a_k] := \ldots$$

note: $\prod a_i = (r_0^{-1}(x_1 \cdots x_k) \cdot r_k$

$a_1 \leftarrow \text{Open}([r_0^{-1} \cdot x_1 \cdot r_1])$

$\vdots$

$a_k \leftarrow \text{Open}([r_{k-1}^{-1} \cdot x_k \cdot r_k])$

local field element mults

$$[r_0] \cdot (a_1 \cdot a_2 \cdots a_k) \cdot [r_k]^{-1}$$

leave mult

DFT on finite fields

- Batch evaluation or interpolation
- Roots of unity

  $\omega \in \mathbb{Z}_p^*$ is an n'th root of unity if

  $$\omega^n \equiv 1 \bmod p$$

  - Every element in $\mathbb{Z}_p^*$ is $(p-1)$ root of unity

    $$x^{p-1} \equiv 1 \bmod p$$

  - Primitive n'th root of unity

    n is the smallest value s.t.

    $$\omega^n \equiv 1 \bmod p$$

  - Suppose $n = 2^k$

    $2^k \mid p-1$

    and ω be a 2^k-th root of unity

    and f is a degree $n-1$ bound polynomial

    $f = a_0 + a_1 x + \cdots a_{n-1} x^{n-1}$

$$DFT_{\omega,n}(f) = \left(f(\omega^0), f(\omega^1), f(\omega^2) \cdots f(\omega^{n-1})\right)$$

As a matrix:

powers $x_0 = \omega^0$ M_{ij}

$$\begin{bmatrix} & & \\ & & \end{bmatrix} \begin{bmatrix} (\omega^0)^0 & \omega^0 & (\omega^0)^0 \\ & & \end{bmatrix}$$

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{bmatrix} \begin{bmatrix} (\omega^0)^1 & \omega^1 & (\omega^2)^1 & \cdots \\ \vdots & \vdots & \vdots & \\ (\omega^0)^{n-1} & \omega^{n-1} & & \omega^{(n-1)(n-1)} \end{bmatrix} = \begin{bmatrix} f(\omega^0) & f(\omega^1) & f(\omega^2) & \cdots \end{bmatrix}$$

$$M_{i,j} = (\omega^i)^j$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \omega^4 & \cdots & \omega^{2(n-1)} = \omega^{2n-2} = \omega^{n-2} \\ 1 & \omega^2 & \omega^4 & & & & \\ \vdots & & & & & & \end{bmatrix}$$

$$\omega = e^{(2\pi i)/n}$$

$$\omega^n = 1 \in \mathbb{C}$$