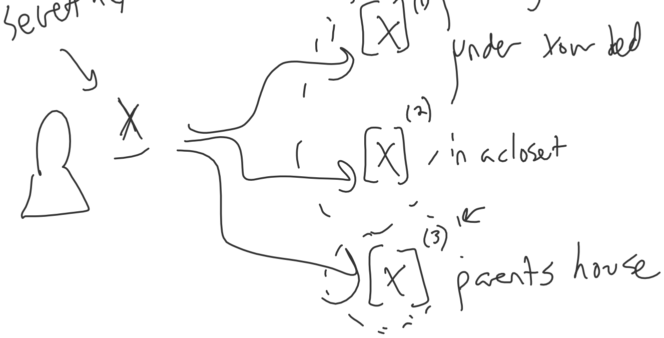
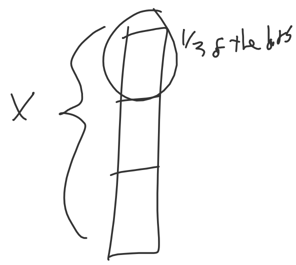
Polynomials and Secret Sharing

Thursday, October 18, 2018 2:08 PM

Secret Sharing

Secret Sharing Backups





2-of-3 secret sharing
- any 2 shares can be combined to get X
- any 1 share reveals no info about X

How do we do this?

Polynomials and Interpolation

Used to polynomials over $\mathbb{R}$

Here we're using polynomials over $\mathbb{F}_p$ finite fields

e.g. $f(\underline{x}) = 5x^2 + 4x + 2$

degree of f is 2

power of the leading term

variable

univariate polynomial

$y^2 = \underline{x}^3 + \underline{7}$ secp256k1 elliptic curve

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

A deg k poly is represented by k+1 coefficients

$$f(x) = a_k x^k + a_{k-1} x^{k-1} + \dots + a_1 x + a_0$$

$$f(x) = \sum^{k} a_i x^i$$

$\sum_{i=0}$

- Degree vs. degree-bound

$0x^3 + \underline{0}x^2 + 5x + 7$

- Are polynomials a group?

  - equality of polynomials

    $f = g$ iff $\forall x \in \mathbb{F}_p,\ f(x) = g(x)$

  - addition:

    $$(f+g)(x) = \underline{f(x)} + \underline{g(x)}$$

equivalently:

$$\text{eval}(a_0, a_1, \ldots a_k, x) = \sum_i a_i x^i$$

$$+$$

$$\text{eval}(b_0, \ldots b_k, x) =$$

$$\text{eval}(a_0 + \underline{b_0}, a_1 + b_1, \ldots a_k + b_k, x)$$

add the coefficients

id: $f(x) = 0$

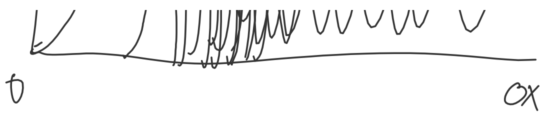
- Do degree-k polys form a group? (degree-bound) ✓

- Polynomials form a ring:

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

$$(a_0 + a_1 x + a_2 x^2)(b_0 + b_1 x + b_2 x^2)$$

$$a_0 b_0 + (a_1 b_0 + b_1 a_0) x + \ldots\ a_2 b_2 x^4$$

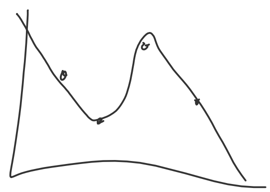


0xfffff... order of secp256k1

$f(x) = x^2$

Lagrange interpolation

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



degree-k lagrange polynomial

Format this takes is:

$$f(x) = \sum_{i=0}^{k} \underline{y_i} \cdot \underline{p_i}(\underline{x})$$

Lemma: Lagrange polynomials

Given $k+1$ distinct values $(x_0, \ldots x_k)$

We can find degree-k polynomials $p_i(x)$

such that $\forall x_j,\ p_i(\underline{x_j}) = \begin{cases} 1 & \text{iff } i=j \\ 0 & i \neq j \end{cases}$

	(x_0)	(x_1)	$\ldots$	(x_k)
p_0	1	0	0	0
p_1	0	1	$0 \ldots$	0
$\vdots$		$\ddots$		$\vdots$
p_k	$\ldots$	$\ldots$	0	1

How do we construct p_i?

Start with $p_0(x)$ $\quad p_0(x_0) = 1,\ p_0(x_i) = 0$ if $i \neq 0$ $x_1, \ldots x_k$

$$p_0(x) = 1 \frac{(x - x_1) \cdot (x - x_2) \cdot \ldots \cdot (x - x_k)}{(\underline{x_0} - x_1) \cdot (\underline{x_0} - x_2) \cdot \quad (\underline{x_0} - x_k)}$$

$$P_1(x) \quad \frac{(x-x_0)}{(x_1-x_0)} \cdot 1 \cdot \frac{(x-x_2)}{(x_1-x_2)} \cdots$$

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

Consequence: $(k+1)$ coefficients $\longleftrightarrow$ $(k+1)$ points $((x_i, y_i), \ldots)$

How to do k-of-n secret sharing of secret value $s \in \mathbb{F}_p$

1. Choose a random degree-$(k-1)$ polynomial f
   Such that $f(0) = s$

   $f(x) = a_{k-1} x^{k-1} + \ldots a_1 x + s \qquad (a_0 = s)$

   draw each a_i for $1 \leq i \leq k-1$ as $a_i \xleftarrow{\$} \mathbb{F}_p$

2. Let the shares be:
   $(1, f(1)), (2, f(2)), \ldots, (n, f(n))$

   Send each share $(i, f(i))$ to node i

   or store each share $(i, f(i))$ at safehouse i

3. To reconstruct given any k shares $(x_i, f(x_i))$

   $$s = f(0) = \sum_{x_i} f(x_i) \left(\prod_{x_j \neq x_i} \frac{0 - x_j}{x_i - x_j} \right)$$

- Polynomial evaluation $\quad a_0 + a_1 \cdot x \ldots \ldots a_k x^k$

  Horner's rule for poly evaluation

- Robust:

  Goal: tolerate up to f invalid/malicious shares.

  Still using: degree-(f) polynomial.

How many shares would you need.

could away? $s' = \underline{s} \mid 000^a$ $\quad y_0, y_1, y_2 \ (y_3 + s'|000)$

$f+2$ shares if f collude,

Degree-t

if $t > 2f$ then even if f collude

$\underline{3f+1}$ shares degree f polynomial

1 — any $(f+1)$ values uniquely determine a polynomial of degree f

2 — if we find $(f+1)$ shares, such that reconstructed poly ϕ' coincides with $(2f+1)$ shares then we know $\phi' = \phi$ ← original polynomial

- why? At most f have errors. At least $\underline{f+1}$ are good shares

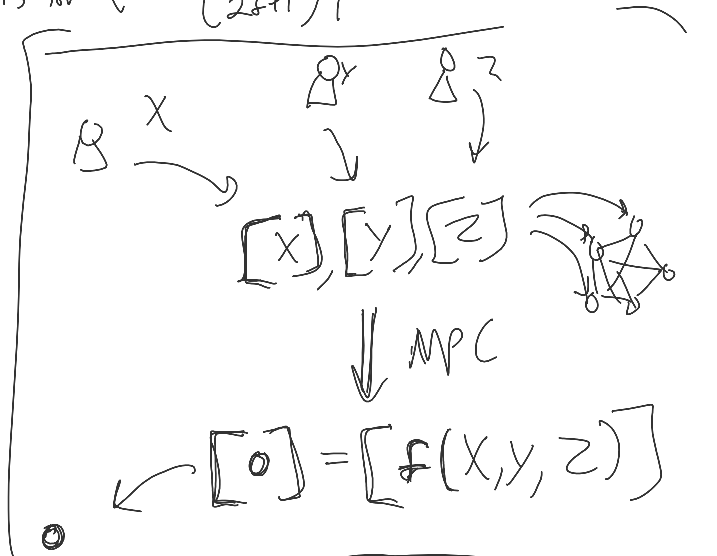
3 — if I have any $3f+1$ shares, some subset of $2f+1$ shares are good implying 2 is solvable $\binom{3f+1}{2f+1}$ possible subsets

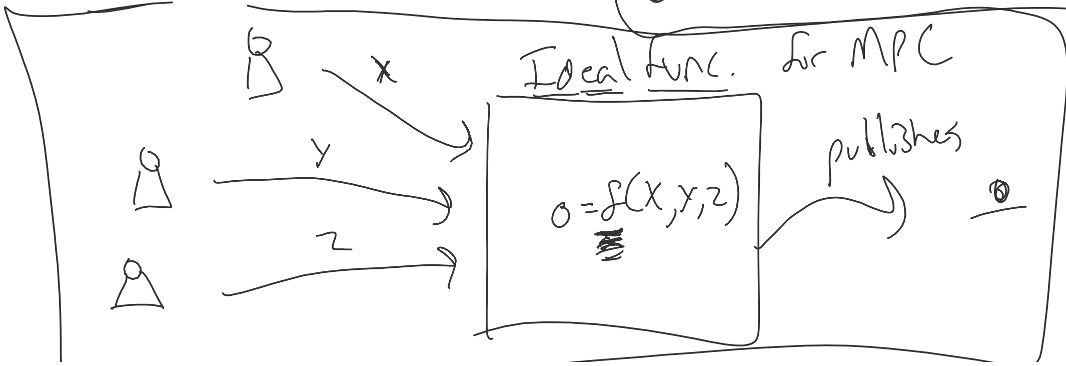
Next time!

secret sharing

$X \rightsquigarrow [x]^{(i)}$

$X \leftsquigarrow [x]$





Computing on Secret Shared data

Input: $[x]_t, [y]_t$ ← degree-t polynomial (bound) $(t+1)$-of-N secret sharing

$P^1 \ldots P^N$

$[x]^{(1)}, [y]^{(1)} \quad \ldots \quad [x]^{(N)} [y]^{(N)}$

Output:

$[x+y]_t$

$[x]^{(1)} + [y]^{(1)}$

$f(1) = [x]^{(1)}$
$g(1) = [y]^{(1)}$
$\Downarrow$
$(f+g)(1) = [x+y]^{(1)} = [x]^{(1)} + [y]^{(1)}$

Algorithm:

$[x+y]_t = [x]_t + [y]_t$

Input: $[x]$
Output: $c \cdot [x]$ ← constant

Linear operations of secret shared data are trivial — just compute locally

Multiplication is harder

$[xy]_t \neq ([x]_t \times [y]_t)$

$[xy]_{2t} = [x]_t \times [y]_t$

4t
8t

$N = 2t+1$

Bearer Multiplication:

- Assume we already have pre-shared random values $[a]_t, [b]_t, [ab]_t$, $a \xleftarrow{\$} \mathbb{F}_p$, $b \xleftarrow{\$} \mathbb{F}_p$
- Input: $[x]_t, [y]_t$
- Goal: $[xy]_t$

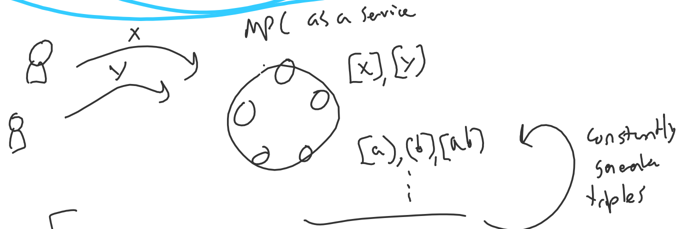
$[D] = ([x] - [a])$
$[E] = [y] - [b]$

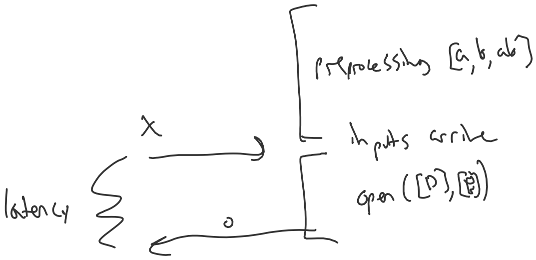
$\left.\begin{array}{l} D \leftarrow \text{open}([D]) \\ E \leftarrow \text{open}([E]) \end{array}\right\}$ Communication

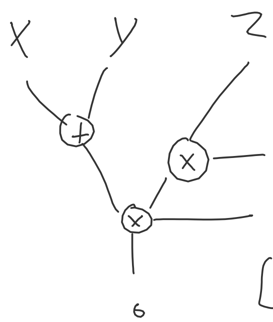
$$[xy] = D\cdot[y] + E[x] + [ab] - DE$$

$$(x-a)[y] + (y-b)[x] + [ab] - (x-a)(y-b)$$

$$\boxed{xy} - ay + xy - bx + ab - xy + ay + bx - ab$$





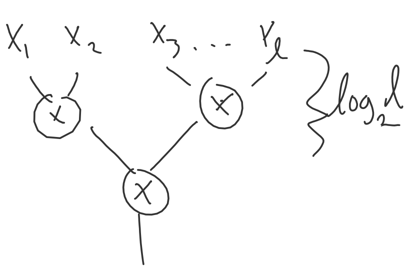


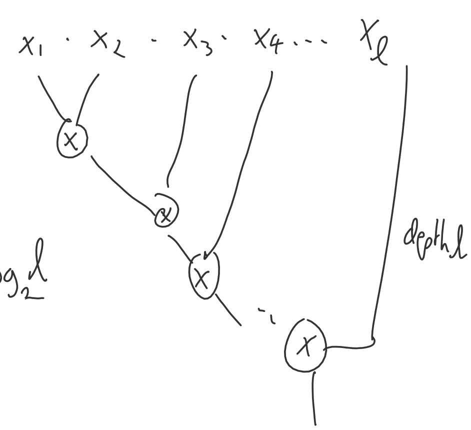
not(A ∧ B)

$[x],[y] \in \{0,1\}$

Emulating boolean $\underline{NAND}(x,y) = 1-xy$

Performance:
- Computation – lagrange interpolation / FFT
- Communication
  - bytes sent (message complexity)
  - rounds of communication
  - preprocessing elements consumed





Suppose we want:

input: $[x]_t$

output: $[x^2]_t$

— w/ beaver $[a]_t, [b]_t, [ab]_t$,

Communication $\underline{\text{open}(D)}$ open(E)   N field elements broadcast per opened element.

— Hint: $[a], [a^2]$

with one opening.

— $D = [x] - [a]$

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

— $$2(x-a)[x] + [a^2] - (x-a)(x-a)$$

$$2x^2 - 2ax + a^2 - x^2 - a^2 + 2ax$$

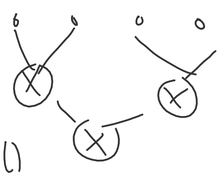
$$[x^2]$$

- Constant Round unbounded fan-in multiplication

input: $[x_1]_t, \dots, [x_\ell]_t$ assume $x_i \neq 0$

output: $[x_1 \cdot x_2 \cdot \dots \cdot x_\ell]_t$

in constant depth



$\log_2 \ell$

Hint: $[r_0], \dots [r_\ell]$ context.get_random()

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

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

$\vdots$

$$[a_\ell]$$

$A_1 = \text{open}([a_1]) \dots \quad A_\ell = \text{open}([a_\ell])$

$$A_i = r_{i-1}^{-1} \cdot x_i \cdot r_i$$

$$A_1 \cdot A_2 \cdot \dots \cdot A_\ell$$

$$([x_1 \dots x_\ell]) = [r_0] \cdot r_0^{-1} \cdot x_1 \cdot x_2 \cdots x_\ell \cdot r_\ell \cdot [r_\ell^{-1}]$$

Double Sharing for degree reduction

Input: $[x][y]_t$

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

Goal: $[xy]_t$

$$[x]_t \cdot [y]_t = [xy]_{2t} \qquad \begin{matrix} 2t \leq N \\ 3t+1 \end{matrix}$$

$$[xy]_{2t} \Longrightarrow [xy]_t$$

Double Sharing: $[r]_t, [r]_{2t}$ $\quad r \xleftarrow{\$} \mathbb{F}_p$

$$(xy - r) = D = \text{open}([xy]_{2t} - [r]_{2t})$$

$$[xy]_t = D + [r]_t$$

$$(xy-r) + [r]_t$$

Application: Dot Product

Input: $[\vec{x}], [\vec{y}]$

Output: $[\vec{x} \cdot \vec{y}]$

$$\sum_i \vec{x}[i] \cdot \vec{y}[i]$$

$$[\vec{x} \cdot \vec{y}]_{2t} = \sum_i [x[i]]_t \cdot [y[i]]_t$$

Randomness extraction

Goal: $[r]_t$ $\quad r \xleftarrow{\$} \mathbb{F}_p$

$N=4$

$x \rightarrow [x]_t$ $\quad$ deg 1

(x_i, y_i) for interpolation

$\Updownarrow$

$(i, [x_i])$ here

Dealer

knows x

$[x]_t^{(4)}$

deg $N-1$

$\phi(i) = x_i \qquad \phi(N+i) = r_i$

Input: $[x_1]_t, \ldots, [x_N]_t \Longrightarrow [r_1]_t, \ldots [r_\ell]$

$\ell = N - t$

each x_i was chosen and secret shared by party i

$t=2$ $\quad$ t $\quad$ $N-t$

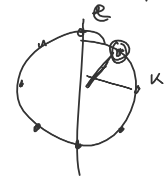
$$\underbrace{x_1 \quad x_2 \quad x_3 \quad x_4}_{N-t \text{ degrees of freedom}} \left(x_5 \quad x_6 \right) \Rightarrow \left(r_1, \ldots, r_{N-t} \right)$$

$\mathbb{F}_p^4$

Fourier transform for efficient polynomial interpolation and evaluation

- Goal: reduce computational cost from N^2 (using Lagrange interpolation) to $N \log N$

$e^{2\pi i}$ $\quad e^{2\pi i/8 \cdot k}$



- Roots of unity

  $\omega \in \mathbb{F}_p$ is a n'th root of unity if $\omega^n = 1$

  $n = |\omega| \qquad \omega \in \mathbb{Z}_p^*$

- in $\mathbb{Z}_p^*$ every element $\omega \in \mathbb{Z}_p^*$ is a $p-1$'th root of unity $\omega^{p-1} = 1$
- if $\omega^n = 1$, then $n \mid p-1$
- primitive / principal n'th root of unity:
  n is the smallest integer so $\omega^n = 1$

- Let n be a power of 2, let $n \mid p-1$, and ω is a primitive n'th root of unity
  Let $f = a_0 + a_1 x + \ldots a_{n-1} x^{n-1}$ be a degree $n-1$ polynomial

Then define

$$\text{DFT}_{\omega, n}(f) = \left(f(1), f(\omega), f(\underline{\omega^2}), f(\omega^3), \ldots f(\omega^{n-1}) \right)$$

DFT: discrete fourier transform

$f(0), \ldots \underbrace{f(1), \ldots f(N-1)}$

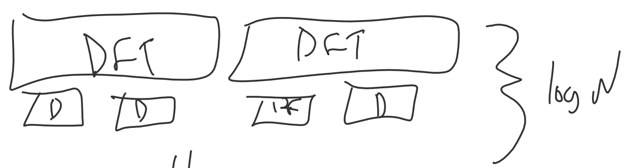
e.g. $\underline{n=8}$,

$$f(\omega^2) = a_0 + a_1(\omega^2) + a_2(\omega^2)^2 + a_3(\omega^2)^3 + \dots a_7\,\omega^{14}$$

$$= (a_0 + a_4) + (a_1 + a_5)\,\omega^2 + \dots (a_3 + a_7)\,\omega^6$$

$a_4(\omega^2)^4 = a_4 \cancel{(\omega^8)}$ $\quad a_5(\omega^2)^5 = a_5\,\omega^2$

$$\underline{DFT}_{\omega,n}(f) = \underline{DFT}_{\omega^2, \underline{n/2}}(f_{evens}) \oplus \underline{DFT}_{\omega^2, \underline{n/2}}(f_{odd})$$

FFT = 

$\Downarrow$

$N \log N$

Inverse DFT gives you interpolation