Threshold Schemes

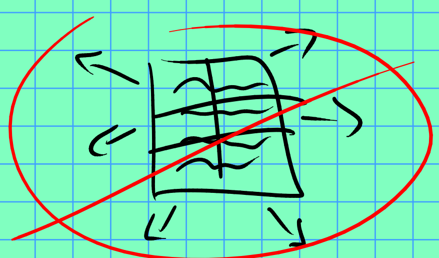
+ Secret Sharing
+ Zero Knowledge Proofs
+ Signatures

(secret key of public/secret pair)

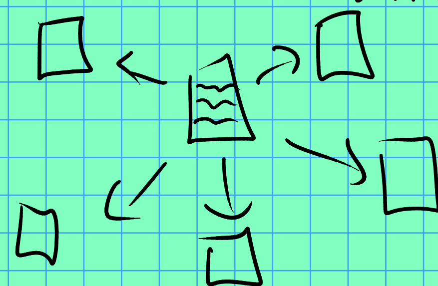
<u>Idea</u> (for encryption): Say key is "shared" among n players. Goal: allow any subset of $> t$ players to decrypt messages encrypted w/ the public key, <u>without ever holding the secret key in any computer's memory</u>.

(decryption is distributed...) $1 < t < n$

What does "shared" mean for the key?



We can do better!



Goal: any collection of $\leq t$ shares contains <u>no information</u> about secret! Yet secret can be reconstructed from any subset of $\geq t+1$ shares.

Main tool: polynomial interpolation. (E.g. Lagrange)

Representations of $f(X) = \sum_{i=0}^{t} c_i X^i$:

- $\{c_i\}_{i=0}^{t}$

– roots of f : (inputs r_i s.t. $f(r_i) = 0$)
$\in \mathbb{C}$

Then $c(x-r_1)(x-r_2) \cdots (x-r_t) = f(x)$

* input/output behavior of f at **any** distinct $t+1$ points: i.e.,
$\{(\alpha_i, \beta_i)\}_{i=0}^{t}$ s.t. $f(\alpha_i) = \beta_i$

looking ahead: to share a secret $s \in \mathbb{Z}_p$ w/ n players w/ threshold t, we will choose random coefficients $c_1, \ldots c_t \in \mathbb{Z}_p$ and set $f(x) = s + c_1 x + c_2 x^2 + \cdots c_t x^t$.

Shares of s? Send player i $f(i)$
(say players are numbered $1, 2, \ldots n$).

Reconstruct? use $t+1$ shares (i/o behavior of f) to find <u>coefficients</u> of f.
Secret $= f(0)$ (constant term)

How to interpolate? Find $f(x)$ from i/o behavior $\{(\alpha_i, \beta_i)\}_{i=0}^{t}$. Want $f(x)$ to have degree $\leq t$, and
$f(\alpha_i) = \beta_i \quad \forall\ i = 0, \ldots, t$.

Warm up/building block: Say you have polynomials (of degree $\leq t$) $\ell_i(x)$ s.t.
$$\ell_i(\alpha_j) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{else } (i = j) \end{cases}$$

How then to find $f(X)$?

Set $f(x) = \beta_0 l_0(x) + \beta_1 l_1(x) + \cdots + \beta_t l_t(x)$

$\deg(f) \le \max\{\deg(l_i)\} \le t$

$$f(\alpha_1) = \cancel{\beta_0 l_0(\alpha_1)} + \underline{\beta_1 l_1(\alpha_1)} + \cancel{\cdots} + \cancel{\beta_t l_t(\alpha_1)}$$

$$= \beta_1$$

More generally, $f(\alpha_i) = \beta_i$

So... how to find $l_i(x)$? Know the roots!!

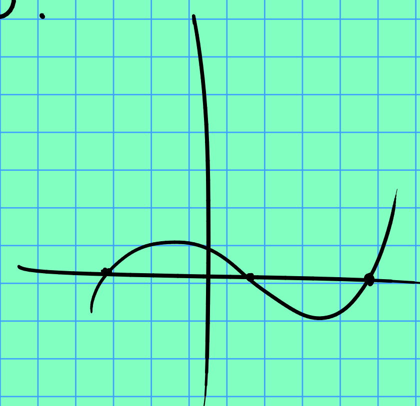
$l_i(x)$ should have roots $\{\alpha_j\}_{j \ne i}$.

So $l_i(x) = c\,(x-\alpha_0)\cdots(x-\alpha_{i-1})(x-\alpha_{i+1})\cdots(x-\alpha_t)$

How to find c? Want $l_i(\alpha_i) = 1$.

So just divide: $c = \dfrac{1}{\tilde{l}_i(\alpha_i)}$ ← why not 0?

where $\tilde{l}_i(x) = \prod_{j \ne i} (x - \alpha_j)$.



Alternate method for interpolation:

<u>Vandermonde Matrix</u>.

Idea: matrix of powers of elements can "linearize" polynomial evaluation.

$$f(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + \cdots + c_t x^t$$

$$V = \begin{pmatrix} 1 & \alpha_0 & \alpha_0^2 & \alpha_0^3 & \cdots & \alpha_0^t \\ 1 & \alpha_1 & \alpha_1^2 & \alpha_1^3 & \cdots & \alpha_1^t \\ \vdots & & \ddots & & & \vdots \\ 1 & \alpha_t & \alpha_t^2 & \alpha_t^3 & \cdots & \alpha_t^t \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_t \end{pmatrix} = \begin{pmatrix} f(\alpha_0) \\ f(\alpha_1) \\ \vdots \\ f(\alpha_t) \end{pmatrix}$$

For interpolation, we have α_i + $f(\alpha_i)$ $(=\beta_i)$ and want to recover c_i's.

Good news: V is invertible! (and it is easy to compute!)

Say $W = V^{-1}$. Then

$$W \begin{pmatrix} \beta_0 = f(\alpha_0) \\ \vdots \\ \beta_t = f(\alpha_t) \end{pmatrix} = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_t \end{pmatrix}$$

" the secret!

$\otimes$ the secret $(s = c_0)$ is a <u>linear function</u> of the shares:

$$s = c_0 = \sum_{i=0}^{t} w_i \beta_i = w_0 \beta_0 + \cdots + w_t \beta_t .$$

coefficients w_i known to participants (could be derived from knowledge of identities of other players)

<u>Threshold Decryption</u> (From El Gamal)

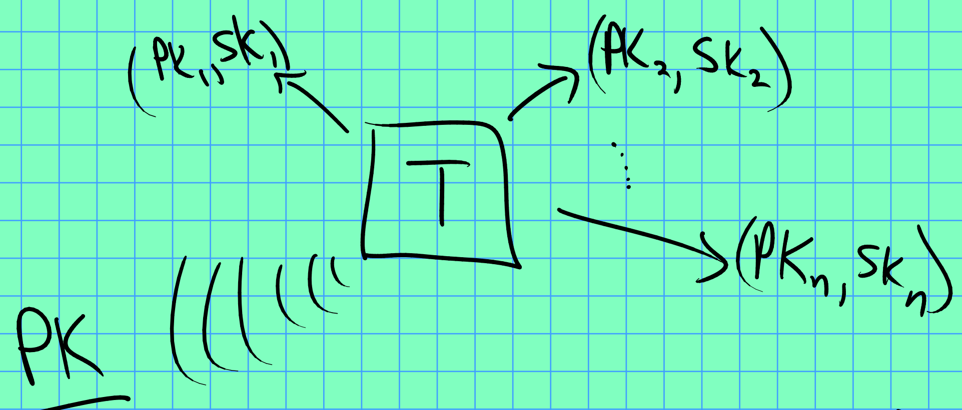
$\langle \gamma \rangle = G < \mathbb{Z}_p^\times$.

$|\langle \gamma \rangle| = q$

$SK = s$

$PK = \gamma^s$

(p, q prime)



How to decrypt w/o reconstructing s??

<u>Reminder</u>: $E_{PK}(m) = (\gamma^b, (PK)^b \cdot m)$ $\quad (m \in \langle \gamma \rangle)$

$b \in_R \mathbb{Z}_q$

(recall: $\langle g \rangle = \{ g^0, g^1, g^2, \dots g^{q-1} \}$)

$$D_{SK}((h,k)) = k \cdot \underline{(h^{-1})^s} \qquad \begin{pmatrix} s = SK; \\ PK = g^s \end{pmatrix}$$

$= m.$

we want to do this from the SK_i w/o reconstructing s !!

Notation: set $s_i \triangleq SK_i$. To reconstruct s,

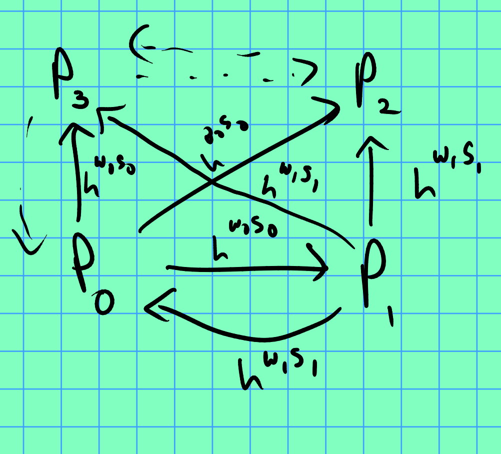
compute $s = \sum_{i=0}^{t} w_i s_i$ where w_i are from V^{-1}.

Say ciphertext is (h,k). $\begin{pmatrix} h = g^b, \\ k = (g^s)^b \cdot m \end{pmatrix}$

$(g^s)^b = h^s$.

Observe: $h^s = h^{\sum w_i s_i} = \prod \underbrace{h^{w_i s_i}}$

- can be computed by player i
- hides s_i in an exponent! ($h^{w_i s_i}$ doesn't reveal s_i to other players)



After sharing, all players have

$$\{ h^{w_0 s_0}, h^{w_1 s_1}, h^{w_2 s_2}, h^{w_3 s_3} \}$$

$\times$

$= h^s$

How do you know $h^{w_i s_i}$ from player i is correct?

You don't. But we can fix it! (ZKP.)