IERG4150
Intro. to Cryptography

Sherman Chow
Chinese University of Hong Kong
Fall 2022
Lecture 3: Secret Sharing

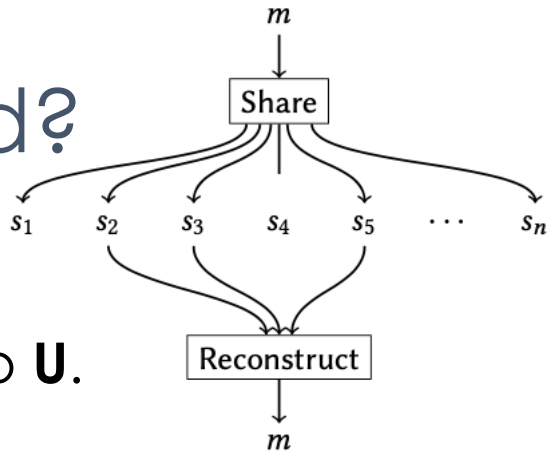
Motivating Stories

- Two-man rule: a minimum of two authorized individuals is needed to do a certain task, *e.g.*, pressing "nuke"
  - Secret inaccessible to a single person
- Only two employees are privy to a certain secret recipe, and they are not permitted to travel together
  - Make sure the secret would not lost forever
- Quorum: at least *t* members of a group are needed to conduct certain business
  - any of *t* can do; any *t*-1 of them cannot
- It's beyond single-recipient encryption we studied (so far)
  - whoever got the ciphertext and the key can decrypt, no other help

(Threshold) Secret-Sharing Scheme

- A *t*-out-of-*n* threshold secret-sharing scheme Σ is defined by
  - a suite of algorithms {Share(·), Reconstruct(·)}
  - parameterized by threshold *t*, total number *n*, & msg. space **M**.
- Share($m \in \mathbf{M}$): a randomized algorithm that
  - takes a ("secret") message $m \in \mathbf{M}$ as input, and
  - outputs a sequences of (secret) shares = $(s_1, \ldots, s_n)$
- We name the users $\{1, \ldots, n\}$ with user i receives share s_i.
- Reconstruct($\{s_i\}_{i \subseteq \{1, \ldots, n\}}$): a deterministic algorithm that
  - takes a collection of *t* or more shares as input, and
  - outputs a message *m'*

Are the users (un)authorized?



for $t \in [1, 2, 3]$

- Let $\mathbf{U} \subseteq \{1, \ldots, n\}$ be a subset of users.
- $\mathbf{s} = \{s_i \mid i \in \mathbf{U}\}$: the set of shares belonging to $\mathbf{U}$.
- $\mathbf{U}$ is said to be authorized if $|\mathbf{U}| \geq t$
  - otherwise, it is unauthorized.

- Correctness: for all $\mathbf{U} \subseteq \{1,\ldots, n\}$ s.t. $|\mathbf{U}| \geq t$ and for all $m \in \mathbf{M}$, if $\mathbf{s} \leftarrow \text{Share}(m)$, we have $\text{Reconstruct}(\{\boldsymbol{s}_i \mid i \in \mathbf{U}\}) = \boldsymbol{m}$.
- Security: if you know only an unauthorized set of shares, you learn *no* information about the choice of message.

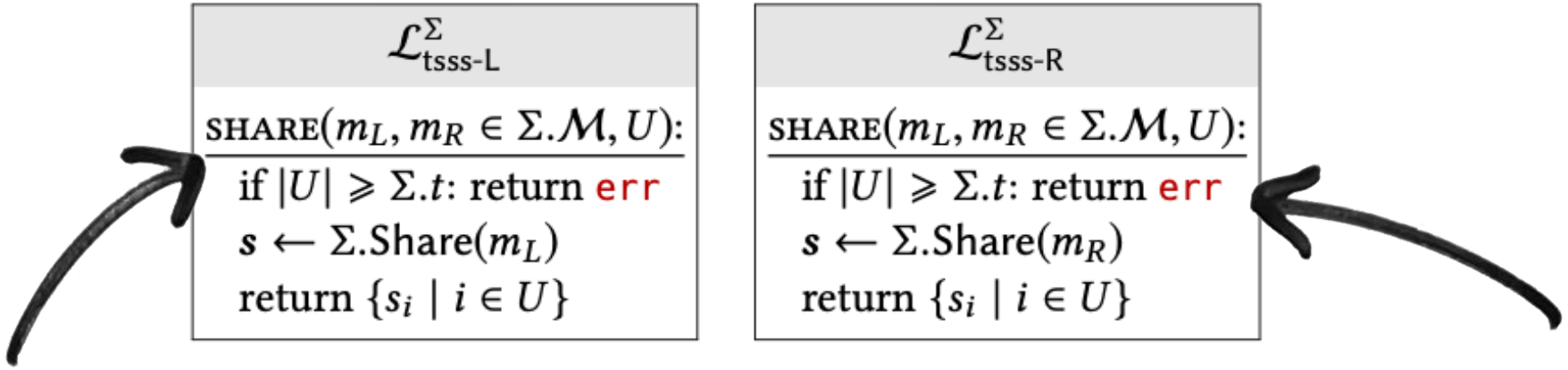
Formalizing Security via Two Libraries

- Both $L_{tss\text{-}L} \equiv L_{tss\text{-}R}$ allow $\mathcal{A}$ to learn a set of shares
  - for an unauthorized set (of course)
- They differ only in which secret (left or right) is shared.

- If they are interchangeable, *i.e.*, Σ is secure if $L_{tss\text{-}L} \equiv L_{tss\text{-}R}$
- we conclude that: seeing an unauthorized set of shares
- leaks no information about the choice of secret message.

- (We will omit $\mathbf{U} \subseteq \{1, \ldots, n\}$ in the definition for brevity.)

TSS's Formal Security Definition



condition for avoiding "trivial win"

Comparison

$\mathcal{L}^{\Sigma}_{\text{tsss-L}}$

SHARE$(m_L, m_R \in \Sigma.\mathcal{M}, U)$:
- if $|U| \geqslant \Sigma.t$: return err
- $s \leftarrow \Sigma.\text{Share}(m_L)$
- return $\{s_i \mid i \in U\}$

$\mathcal{L}^{\Sigma}_{\text{tsss-R}}$

SHARE$(m_L, m_R \in \Sigma.\mathcal{M}, U)$:
- if $|U| \geqslant \Sigma.t$: return err
- $s \leftarrow \Sigma.\text{Share}(m_R)$
- return $\{s_i \mid i \in U\}$

- We studied encryption's
- Let the calling program choose the two secret messages
  - modeling an adversary that has partial knowledge or influence on the secret *m* being shared (§2: p.18)
- The libraries make it impossible for the calling program to obtain the shares of an *authorized* set
  - ✗ a user is never allowed to distribute an authorized set of shares
  - ✓ the definition only guarantees security when $|\mathbf{U}| < t$
  - ✓ the definition said nothing about security when $|\mathbf{U}| \geq t$
- Ex.: $\mathcal{A}$ receives shares of [1, 5] in 1 call and [6, 10] in another?
  - We talked about the "answer" of this exercise in §2: p.19 already

What is not considered/protected?

- Only hides the internal differences between the libraries
- No protection of "anything that is the same btw. 2 libraries"
- Does not require shares to hide which user they belong to
- Ex.: Given a secure TSS scheme, prove that TSS', which just appends each user's identity to his/her share, is secure too.

- Correctness and security definitions do not cover operational characteristics/requirements
  - *e.g.*, who should run Share()? (stay tuned… till the end of lecture)

Example of an Insecure TSS Scheme

- Why? To illustrate the "beauty" of a secure scheme
- Let's consider $(t, n) = (5, 5)$, $\mathbf{M} = \{0, 1\}^{500}$

Share(m):

split m into $m = s_1 \| \cdots \| s_5$,
where each $|s_i| = 100$
return $(s_1, \ldots, s_5)$

Reconstruct($s_1, \ldots, s_5$):

return $s_1 \| \cdots \| s_5$

- Attack:

"It is easy to (design a scheme that) reveal(s) information about the secret gradually as more shares are obtained."

$\mathcal{A}$

$s_1 := \text{SHARE}(0^{500}, 1^{500}, \{1\})$

return $s_1 \stackrel{?}{=} 0^{100}$

A simple 2-out-of-2 TSS scheme

- Consider **M** = $\{0, 1\}^\ell$

$\underline{\text{Share}(m):}$
$s_1 \leftarrow \{0,1\}^\ell$
$s_2 := s_1 \oplus m$
return (s_1, s_2)

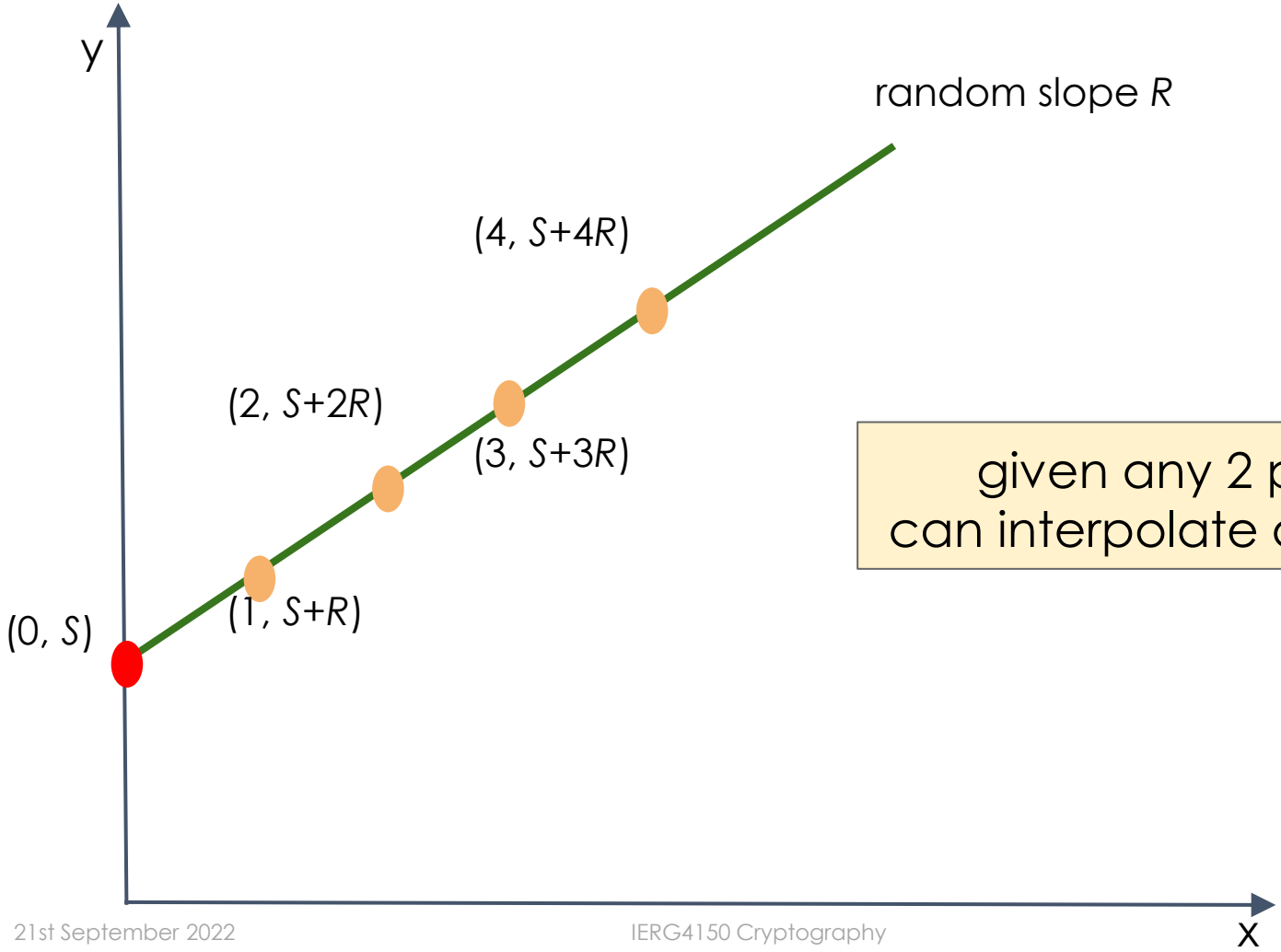
$\underline{\text{Reconstruct}(s_1, s_2):}$
return $s_1 \oplus s_2$

- As a (2, 2) scheme, the only authorized set of users is {1, 2}.
- Reconstruct is written to expect both shares s_1 and s_2.

- Formal security proof: 3-xor-sss.pdf
  - the proof shows that any one-time secure encryption can do

From (2, 2) to (*t*, *n*)

- In (2, 2) secret sharing, each share must leak absolutely no information about the secret.
- For general (*t*, *n*), the shares must leak absolutely no information about the secret, until the number of shares reaches the threshold value *t*.
- How do we do that? We will (re-)study some math.
  - Polynomial interpolation
  - Modular arithmetic



given any 2 points,
can interpolate and find *S*

Polynomials

- 2 points define a line (in Euclidean geometry)
- 3 points define a parabola, *i.e.*, a degree-2 polynomial
  - $ax^2 + bx + c$ for some constants a, b, c, say, in **R** (real numbers)
- (d + 1) points define a unique degree-d polynomial

- f is a degree-d polynomial means $f(x) = \sum_{i \in [0, d]} f_i x^i$
  - for convenience, we mandate $f_d \neq 0$ ("non-trivial" degree-d poly.)

Polynomials Interpolation over the Reals

- Theorem: Let $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1}) \in \mathbf{R}^2$ be a set of points whose x_i values are all distinct. Then there is a unique degree-d polynomial f with real coefficients that satisfies $y_i = f(x_i)$ for all i.

- We need to fine the coefficients $\{f_i\}$ in $f(x) = \Sigma f_i x^i$.
- Start with $x = 0$, $f(0) = f_0$. Easy~
- Let's try $x = 1$? (suppose it is the x-coordinate of a given point)
- $f(1) = \Sigma f_i$, we only know f_0 so far, we have many to solve.
- Do we have a "cleverer" or more elegant/systematic way?

Constructive Proof (Illustration)

- Let's consider $\ell_1(\boldsymbol{x}) = \dfrac{(\boldsymbol{x}-x_2)(\boldsymbol{x}-x_3)\cdots(\boldsymbol{x}-x_{d+1})}{(x_1-x_2)(x_1-x_3)\cdots(x_1-x_{d+1})}$
- $\boldsymbol{x}$ denotes a formal variable while x_i denotes a number
- The numerator is a degree-d polynomial
- $x_1, \ldots, x_{d+1}$ are those "scalars" given in the theorem (p.14).
- The denominator is just a scalar
  - Note that $(x_1 - x_1)$ is not there, hence it is not zero.
- Why we consider this polynomial $\ell_1()$?
- $\ell_1(x_1) = 1$, but $\ell_1(x_i) = 0$ for all $i \neq 1$ $\because$ $(x_1 - x_i)$ is there

Lagrange polynomials

- Generally, consider $\ell_j(\boldsymbol{x}) = \dfrac{(\boldsymbol{x} - x_1) \cdots (\boldsymbol{x} - x_{j-1})(\boldsymbol{x} - x_{j+1}) \cdots (\boldsymbol{x} - x_{d+1})}{(x_j - x_1) \cdots (x_j - x_{j-1})(x_j - x_{j+1}) \cdots (x_j - x_{d+1})}$
- Pattern:
  - the numerator is "missing" the term $(\boldsymbol{x} - x_j)$
  - the denominator is missing the term $(x_j - x_j)$
- $\ell_j(x_j) = 1$, but $\ell_j(x_i) = 0$ for all $i \neq j$
- Lagrange polynomials = those satisfy the above

$f(\mathbf{x})$ passing thro' $\{(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})\}$

- Now consider $f(\boldsymbol{x}) = y_1\ell_1(\boldsymbol{x}) + y_2\ell_2(\boldsymbol{x}) + \cdots + y_{d+1}\ell_{d+1}(\boldsymbol{x})$
- f is of degree d since it's a sum of degree-d polynomials
- y_i's are just scalar (given in the theorem on p.14)

- What if $f()$ is evaluated at those x_i's in the theorem?
- Recall $\ell_j(x_j) = 1$, but $\ell_j(x_i) = 0$ for all $i \neq j$
- $f(x_i) = y_1 \cdot 0 + y_2 \cdot 0 + \cdots + y_i \cdot 1 + \cdots + y_{d+1} \cdot 0$
- $= y_i$ (as desired!)

Uniqueness of f

- We just show such f "works" but what if there's another?
- Suppose that there are two degree-d polynomials f and f'
- such that $f(x_i) = f'(x_i) = y_i$ for $i \in \{1, \ldots, d + 1\}$.
- $g(\mathbf{x}) = f(\mathbf{x}) - f'(\mathbf{x})$ is also degree-d
- By our starting condition, $g(x_i) = 0$ for all i.
- Each x_i is a root of g, so g has at least $(d + 1)$ roots.
- Degree-d $g(\mathbf{x})$ has $(d + 1)$ roots?
- The only degree-d polynomial with $(d + 1)$ roots is the identically-zero polynomial $g(\mathbf{x}) = 0$, which implies $f = f'$

Lagrange Interpolation in Action

i	1	2	3	4
x_i	3	4	5	2
y_i	1	1	9	6

$$\ell_1(x) = \frac{(x-x_2)(x-x_3)(x-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)} = \frac{(x-4)(x-5)(x-2)}{(3-4)(3-5)(3-2)} = \frac{x^3-11x^2+38x-40}{2}$$

$$\ell_2(x) = \frac{(x-x_1)(x-x_3)(x-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)} = \frac{(x-3)(x-5)(x-2)}{(4-3)(4-5)(4-2)} = \frac{x^3-10x^2+31x-30}{-2}$$

$$\ell_3(x) = \frac{(x-3)(x-4)(x-2)}{(5-3)(5-4)(5-2)} = \frac{x^3-9x^2+26x-24}{6}$$

$$\ell_4(x) = \frac{(x-3)(x-4)(x-5)}{(2-3)(2-4)(2-5)} = \frac{x^3-12x^2+47x-60}{-6}$$

$$\begin{aligned} f(x) &= y_1 \cdot \ell_1(x) + y_2 \cdot \ell_2(x) + y_3 \cdot \ell_3(x) + y_4 \cdot \ell_4(x) \\ &= 1 \cdot \ell_1(x) + 1 \cdot \ell_2(x) + 9 \cdot \ell_3(x) + 6 \cdot \ell_4(x) \\ &= \frac{1}{6}\begin{pmatrix} 1\cdot(& 3x^3 & -33x^2 & +114x & -120) \\ +1\cdot(& -3x^3 & +30x^2 & -93x & +90) \\ +9\cdot(& x^3 & -9x^2 & +26x & -24) \\ +6\cdot(& -x^3 & +12x^2 & -47x & +60) \end{pmatrix} \\ &= \frac{1}{6}\left(3x^3-12x^2-27x+114\right) \\ &= \frac{x^3}{2} - 2x^2 - \frac{9x}{2} + 19 \end{aligned}$$

You can do many sanity checks here.

Modular Arithmetic

- We can't have a uniform distribution over the real numbers $\mathbf{R}$.
- Define $\mathbf{Z}_p = \{0, 1, \ldots, p-1\}$ and $\mathbf{Z}_p^* = \mathbf{Z}_p \setminus \{0\}$ for a prime p
- Intuition: Consider OTP over $(\mathbf{Z}_p^*, \cdot)$ instead of $(\{0,1\}^\ell, \oplus)$
  - You may consider $(\mathbf{Z}_p, +)$ too
- Enc(k, m) outputs $c := k \cdot m \bmod p$
- Given any $c \in \mathbf{Z}_p^*$, can you "explain" it is encrypting m?
- Yes, if there always exists one and only one $k \in \mathbf{Z}_p$
- Does it? Yes, if p is prime
- k is also known as the multiplicative inverse of m mod p
  - (It can be found using extended Euclidean algorithm, stay tuned…)

Polynomials mod p

- "Old" Theorem: Let $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1}) \in \mathbf{R}^2$ be a set of points whose x_i values are all distinct. Then there is a unique degree-d polynomial f with coefficients from $\mathbf{R}$ that satisfies $y_i = f(x_i)$ for all i.

- Let p be a prime. Let $(x_1, y_1), \ldots, (x_{d+1}, y_{d+1}) \in (\mathbf{Z}_p)^2$ be a set of points whose x_i values are all distinct. Then there is a unique degree-d polynomial f with coefficients from $\mathbf{Z}_p$ that satisfies $y_i \equiv_p f(x_i)$ for all i.

- The same proof works if you can always "divide" mod p (except by 0) when p is a prime (+, -, · are trivial)

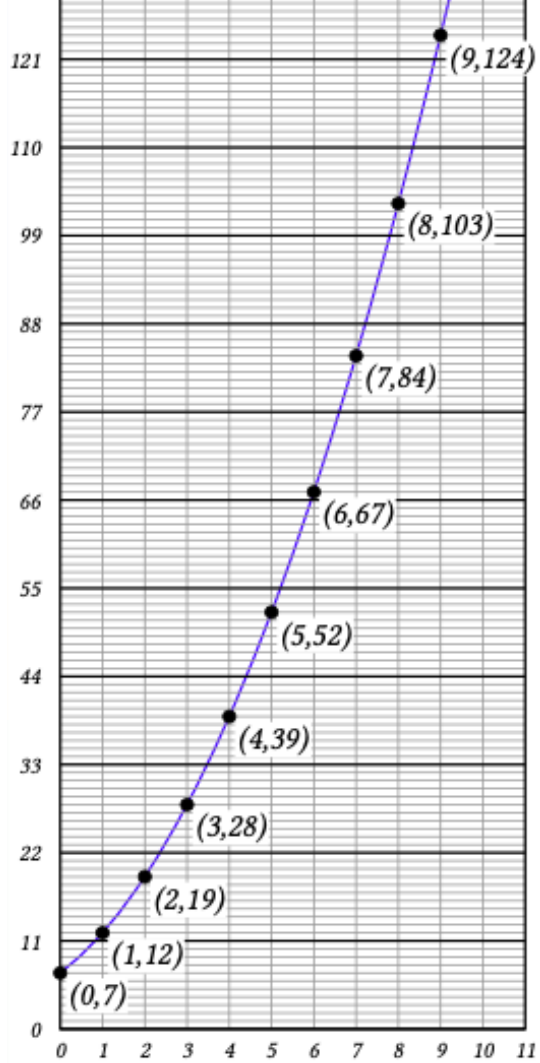
- What if fewer points are given? (Why consider this question?)

Generalizing for # of points $k \leq d + 1$

- Let p be a prime. Let $\{(x_1, y_1), \ldots, (x_{d+1}, y_{d+1})\} \in (\mathbf{Z}_p)^2$ be a set of points whose x_i values are all distinct. Then there is a unique degree-d polynomial f with coefficients from $\mathbf{Z}_p$ that satisfies $y_i \equiv_p f(x_i)$ for all i.
- Let p be a prime. Let $\{(x_1, y_1), \ldots, (x_k, y_k)\} \in (\mathbf{Z}_p)^2$ be a set of points whose x_i values are all distinct. Let $k \leq d + 1$, $p > d$. The number of degree-d polynomials f with coefficients from $\mathbf{Z}_p$ that satisfies $y_i \equiv_p f(x_i)$ for all i is exactly p^{d+1-k}.
  - There are exactly p^{d+1-k} polynomials of degree-d that hit any set of k points, mod p. // k = # points in next few slides

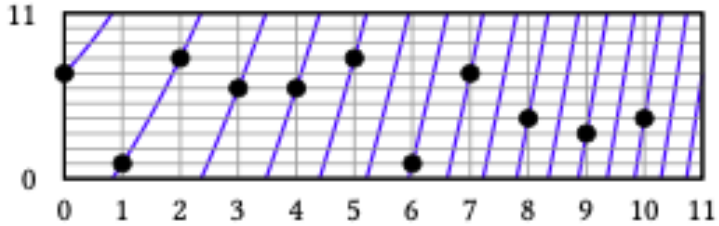
Proof by Mathematical Induction

- Base case (lacking 0 points): $d+1-k = 0$, $p^{d+1-k} = p^0 = 1$.
- With a set **P** of $k \leq d$ points (lacking $d - k$ points), let $x^* \in \mathbf{Z}_p$ be a value that doesn't appear as a x-coordinate of points in **P**.
- Every polynomial must evaluate to *some* value y^* at x^*. -- (*)
- Let **P'** = **P** ∪ $\{(x^*, y^*)\}$ // Step case: lacking $d - (k + 1)$ points
- [# of deg.-d poly. passing thro' points in **P**] // Inductive step
- = $\sum_{y^* \in \mathbf{Z}_p}$ [# of deg.-d poly. passing thro' points in **P'**] // due to (*)
- = $\sum_{y^* \in \mathbf{Z}_p} p^{d+1-(k+1)}$ // by the step case for $d+1-k-1 = d - k$
- = p^{d+1-k} // will be useful for proving security

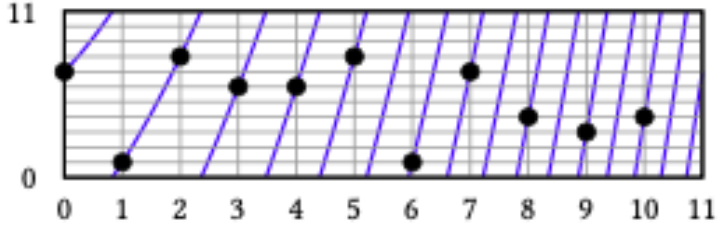


What does a "polynomial mod *p*" look like?

- Let's consider degree-2 polynomial: $f(\mathbf{x}) = \mathbf{x}^2 + 4\mathbf{x} + 7$
- Left-hand side: over $\mathbf{R}$.
- Below: over $\mathbf{Z}_p$ (can't "go beyond" 11)
- Only 11 points, not the blue curve
- A "mod-11" parabola



Shamir Secret Sharing



$f(\mathbf{x}) = \mathbf{x}^2 + 4\mathbf{x} + 7 \bmod 11$

i	1	2	4	5
x_i	1	2	4	5
y_i	1	8	6	8

- Adi Shamir (1979), "How to share a secret." Communications of the ACM, 22 (11): 612–613
- $\mathbf{M} = \mathbf{Z}_p$, $n < p$, $t \leq n$.
- Idea: set $m := f(0)$, pick a random degree-t poly. mod p

Share(m):

$f_1, \ldots, f_{t-1} \leftarrow \mathbb{Z}_p$
$f(\boldsymbol{x}) := m + \sum_{j=1}^{t-1} f_j \boldsymbol{x}^j$
for $i = 1$ to n:
  $s_i := (i, f(i) \,\%\, p)$
return $\boldsymbol{s} = (s_1, \ldots, s_n)$

Reconstruct($\{s_i \mid i \in U\}$):

$f(\boldsymbol{x}) :=$ unique degree-$(t-1)$ polynomial mod p passing through points $\{s_i \mid i \in U\}$
return $f(0)$

Share() *is a randomized algorithm*

Security of Shamir Secret Sharing

$\mathcal{L}_{\text{shamir-real}}$

$\text{POLY}(m, t, U \subseteq \{1, \ldots, p\})$:

if $|U| \geqslant t$: return err

$f_1, \ldots, f_{t-1} \leftarrow \mathbb{Z}_p$

$f(x) := m + \sum_{j=1}^{t-1} f_j x^j$

for $i \in U$:

$s_i := (i, f(i) \% p)$

return $\{s_i \mid i \in U\}$

Share(m):

$f_1, \ldots, f_{t-1} \leftarrow \mathbb{Z}_p$

$f(x) := m + \sum_{j=1}^{t-1} f_j x^j$

for $i = 1$ to n:

$s_i := (i, f(i) \% p)$

return $s = (s_1, \ldots, s_n)$

$\mathcal{L}_{\text{shamir-rand}}$

$\text{POLY}(m, t, U \subseteq \{1, \ldots, p\})$:

if $|U| \geqslant t$: return err

for $i \in U$:

$y_i \leftarrow \mathbb{Z}_p$

$s_i := (i, y_i)$

return $\{s_i \mid i \in U\}$

- $L_{\text{shamir-rand}}$ gives uniformly chosen points, unrelated to any polynomial.

Proof of $L_{\text{shamir-real}} \equiv L_{\text{shamir-rand}}$

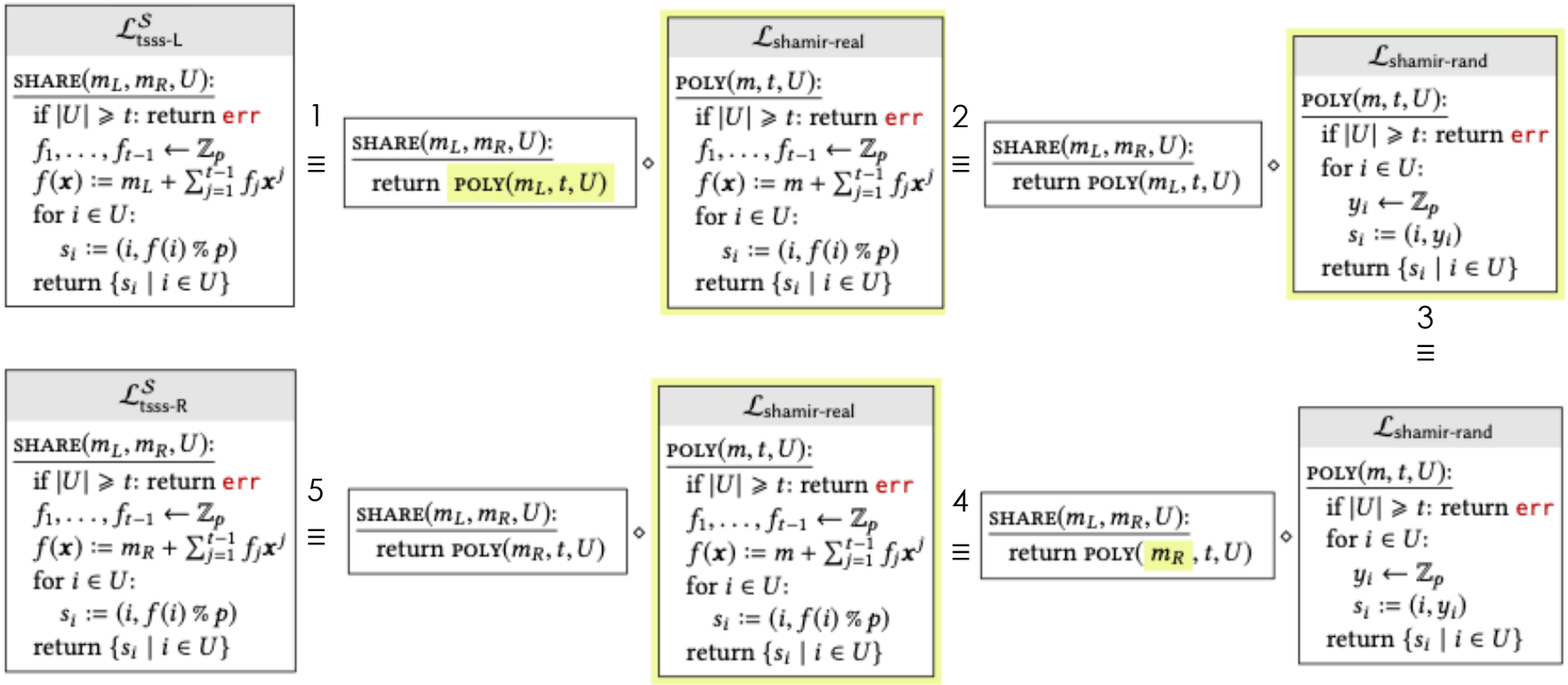
$\textsc{poly}(m, t, U \subseteq \{1, \ldots, p\})$:
- if $|U| \geqslant t$: return err
- $f_1, \ldots, f_{t-1} \leftarrow \mathbb{Z}_p$
- $f(x) := m + \sum_{j=1}^{t-1} f_j x^j$
- for $i \in U$:
  - $s_i := (i, f(i) \% p)$
- return $\{s_i \mid i \in U\}$

- Fix $m \in \mathbf{Z}_p$, set $\mathbf{U}$ of users with $|\mathbf{U}| < t$, for each $i \in \mathbf{U}$, fix $y_i \in \mathbf{Z}_p$
- We consider the probability that a call to poly(m, t, $\mathbf{U}$) outputs $\{(i, y_i) \mid i \in \mathbf{U}\}$, in each of the two libraries
  - *cf*. fix m and c, compute Pr[`Eavesdrop`(m) = c] back then
- $L_{\text{shamir-real}}$ chooses a random degree-$(t-1)$ polynomial f s.t. $f(0) \equiv_p m$ (only k=1 pt.) # of such $f = p^{d+1-k} = p^{t-1+1-1}$
- For POLY to (have chosen a f such that it) outputs points consistent with $\mathbf{P} = \{(i, y_i) \mid i \in \mathbf{U}\}$, # of such $f = p^{t-1+1-(|\mathbf{U}|+1)}$
- Prob. = #desired/#possible = $p^{t-|\mathbf{U}|-1} / p^{t-1} = p^{-|\mathbf{U}|}$
- $L_{\text{shamir-rand}}$ chooses its $|\mathbf{U}|$ outputs from $\mathbf{Z}_p$. Prob hitting $\mathbf{P}$ = $p^{-|\mathbf{U}|}$ too

$\mathcal{L}_{\text{shamir-rand}}$

$\textsc{poly}(m, t, U \subseteq \{1, \ldots, p\})$:
- if $|U| \geqslant t$: return err
- for $i \in U$:
  - $y_i \leftarrow \mathbb{Z}_p$
  - $s_i := (i, y_i)$
- return $\{s_i \mid i \in U\}$

Hybrid Argument for Proving $L_{tss\text{-}L} \equiv L_{tss\text{-}R}$



Who should run Share()?

- "A trusted dealer"
- Let's twist our question:
- What if we just want the users to establish a random secret?

- Select t of them to be a "trusted dealer group" **U**.
- All agree on a prime p.
- Each user i in **U** just pick $\{(i, y_i)\}$ where $y_i \in_R \mathbf{Z}_p$
  - $\in_R$: randomly choose from
- $\mathbf{P} = \{(i, y_i) \mid i \in \mathbf{U}\}$ defined a shared degree-$(t-1)$ poly. $f(x)$
- All in **U** interpolate to compute $\{(j, y_j = f(j))\}$ for each other user j
- Not really "secure" but easy to describe at the moment

More Applications of Secret Sharing [**]

- Once the secret has been reconstructed, it's reconstructed
- How about we want it to do something, say decryption?

- Ex.: Consider (n, n) share of a key k in $\{0, 1\}^{\ell}$
- How do the users perform (n, n)-encryption and (n, n)-decryption for OTP without explicitly recovering the key k?

A "Real World" Application of SS [**]

- We will teach you what digital signature is
- It takes a "private (signing) key"
- We use SS to share the private key into shares
- "Threshold signing": Sign with the share separately
  - reconstruct the signature instead of the key
  - but one needs a "compatible" (and secure) scheme
  - *i.e.*, not every scheme is extensible to the threshold case
- store a key in plaintext on the PC – **bad idea** (*e.g.*, malware)
- better: **split the key** between several devices

Visual Secret Sharing a.k.a. Visual Crypto. [**]

- https://www.youtube.com/watch?v=iKDUltn-ngA