How to share a secret

Lilika Markatou

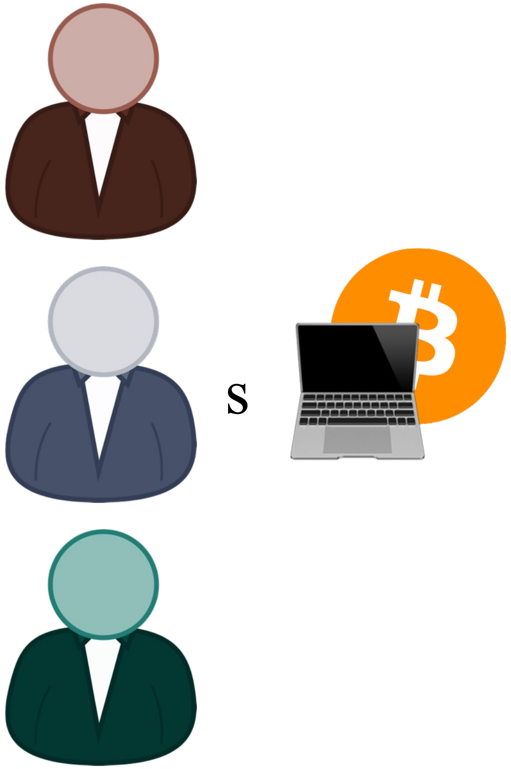
Key to Bitcoin Wallet

n friends who mined bitcoin in 2010.

How to share key s?

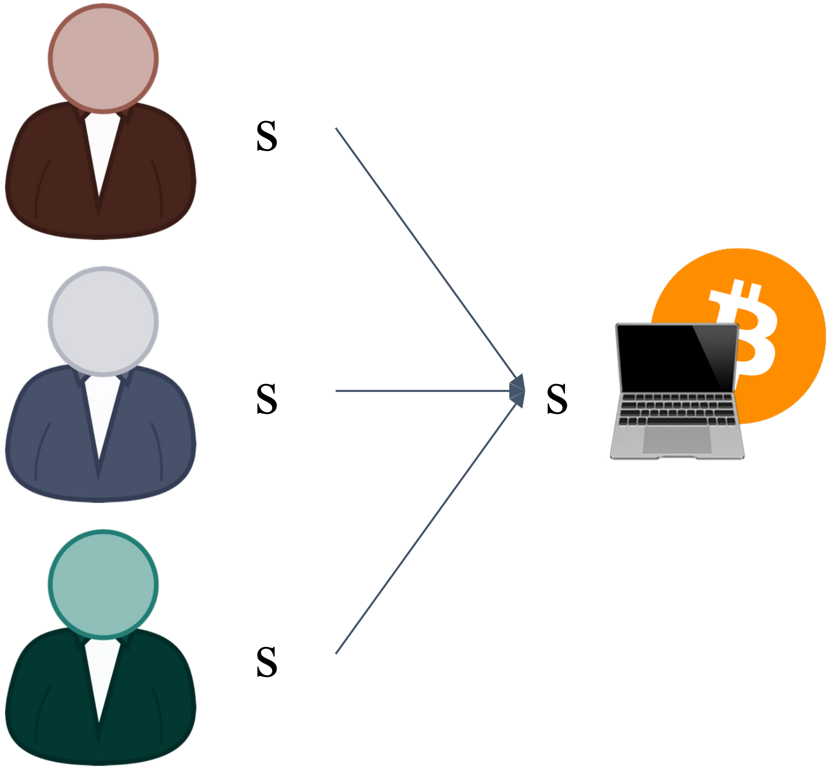
(or n bank managers who need to access the vault)

(or n generals with access to nuclear codes)



1-out-of n Secret Sharing

- Everyone knows s!

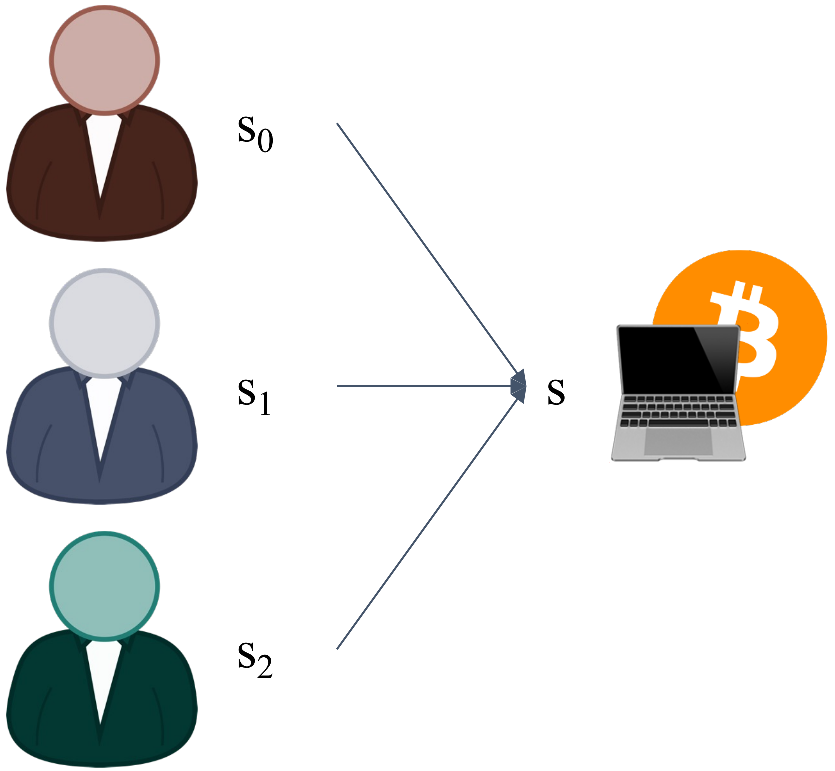


Drawbacks:

- What if one of them gets kidnapped?
- What if not everyone is trusted?

n-out-of n Secret Sharing

- Split s into 3 shares.
- Everyone is needed to reconstruct s from the shares



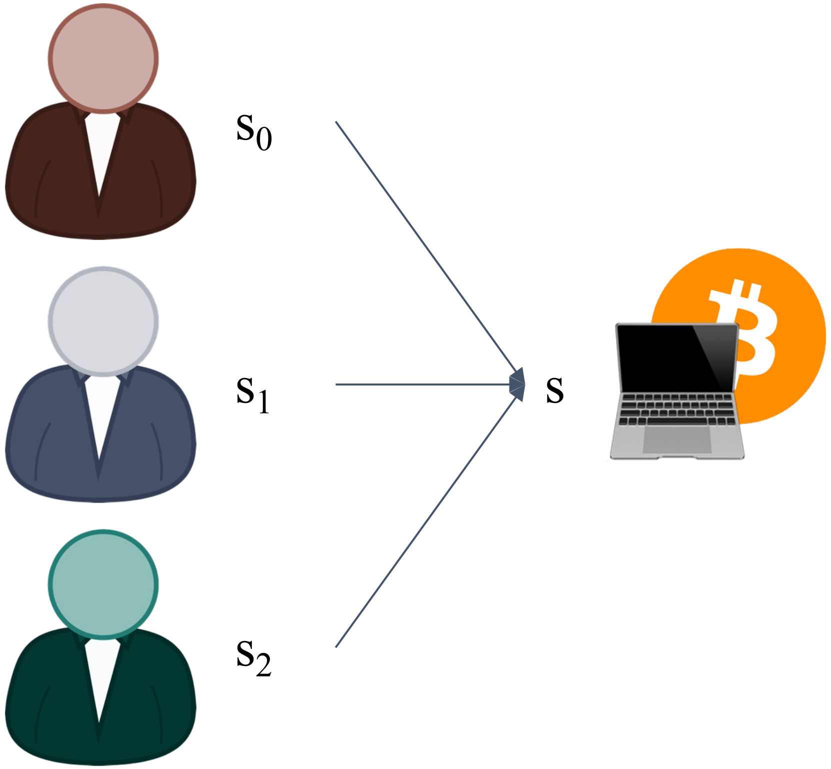
n-out-of n Secret Sharing

Trusted Dealer:

1. Picks s_0 and s_1 randomly
2. $s_2 = s \oplus s_0 \oplus s_1$

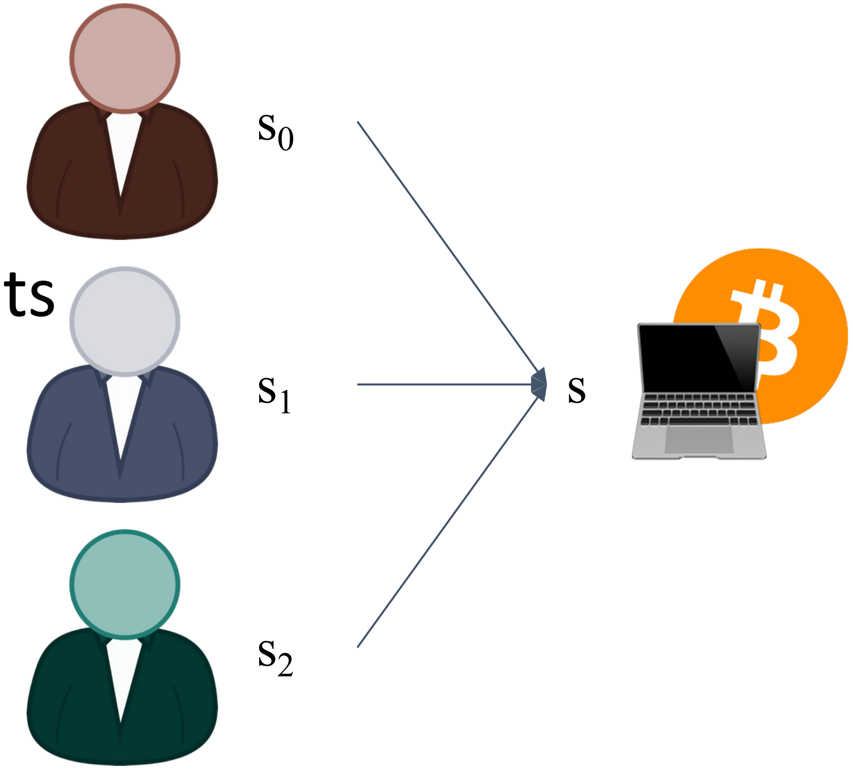
Secret Recovery:

$$\begin{aligned} s &= s_0 \oplus s_1 \oplus s_2 \\ &= s_0 \oplus s_1 \oplus \mathbf{s \oplus s_0 \oplus s_1} \\ &= \mathbf{(s_0 \oplus s_0)} \oplus s \oplus \mathbf{(s_1 \oplus s_1)} \\ &= (00..00) \oplus s \oplus (00..00) \\ &= s \end{aligned}$$



n-out-of n Secret Sharing

- Information Theoretic Security:
  - No information about s can be recovered from fewer than all shares.
- With fewer than 3 shares all possible secrets are equally likely.



For any s_{fake} and shares s_0 , s_1:

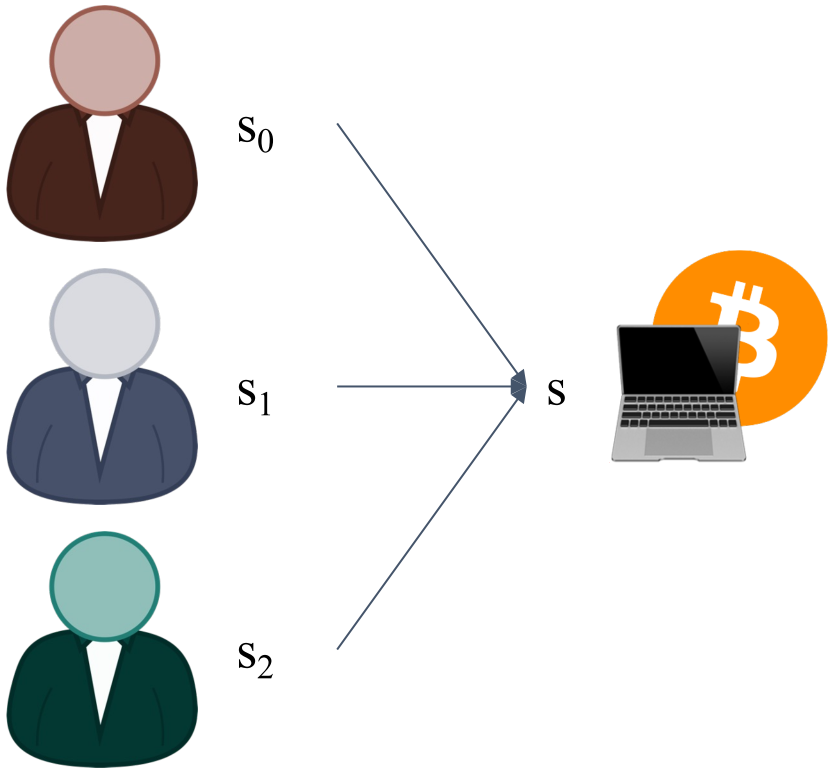
$$s_2 = s_{fake} \oplus s_0 \oplus s_1$$

n-out-of n Secret Sharing

Drawbacks:

- What if someone loses their share?
- What if no consensus is reached?

Can only two shares suffice to recover s?

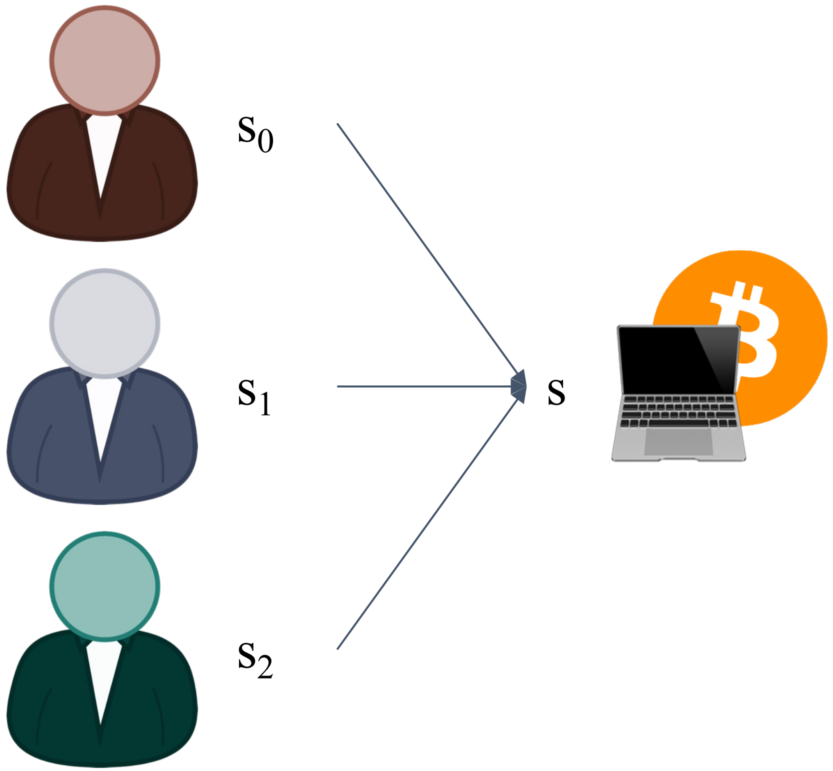


"How to share a secret"

Adi Shamir (1979)

TUDelft

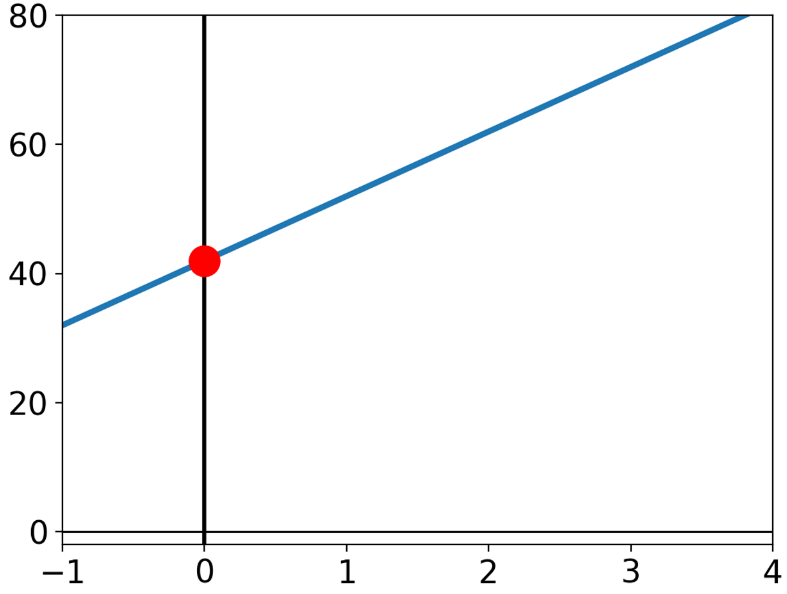
2-Out-Of-n Secret Sharing



Any two shares are sufficient to recover s.

Lines

- Let's say the secret is $s=42$.
- Consider $f(x) = 10x + 42$, where 10 was picked randomly.
- The secret is recovered by computing $f(0)$.



- Each user is given a share, which is a point of f. Any **two** users can recover f.

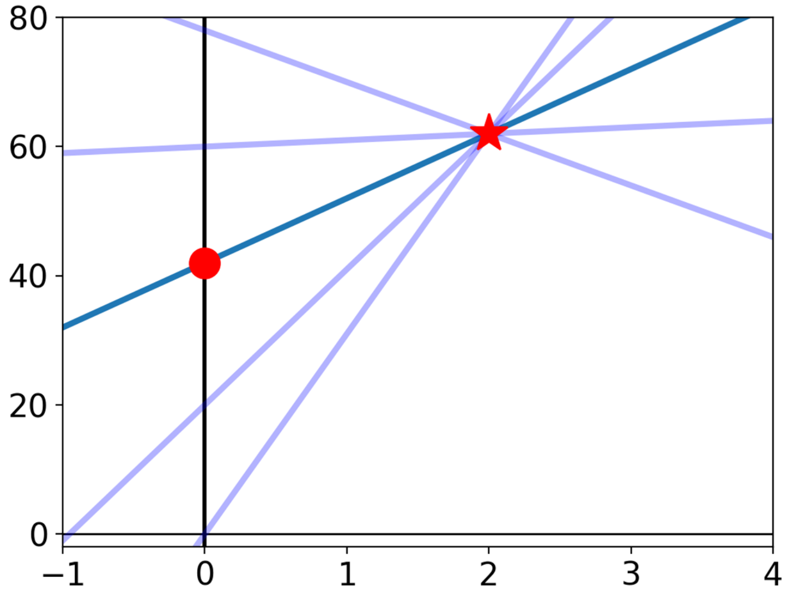
$$s_1 = (2,62),\ s_2 = (1,52) \Rightarrow f(x) = (62\text{-}52)/(2\text{-}1)\ x + (52\text{-}10) \Rightarrow f(x) = 10x + 42 \Rightarrow \mathbf{s = f(0) = 42}$$

TU Delft

What can you recover with only one share?

Lines

- Let’s say the secret is $s=42$.
- Consider $f(x) = 10x + 42$, where 10 was picked randomly.
- The secret is recovered by computing $f(0)$.

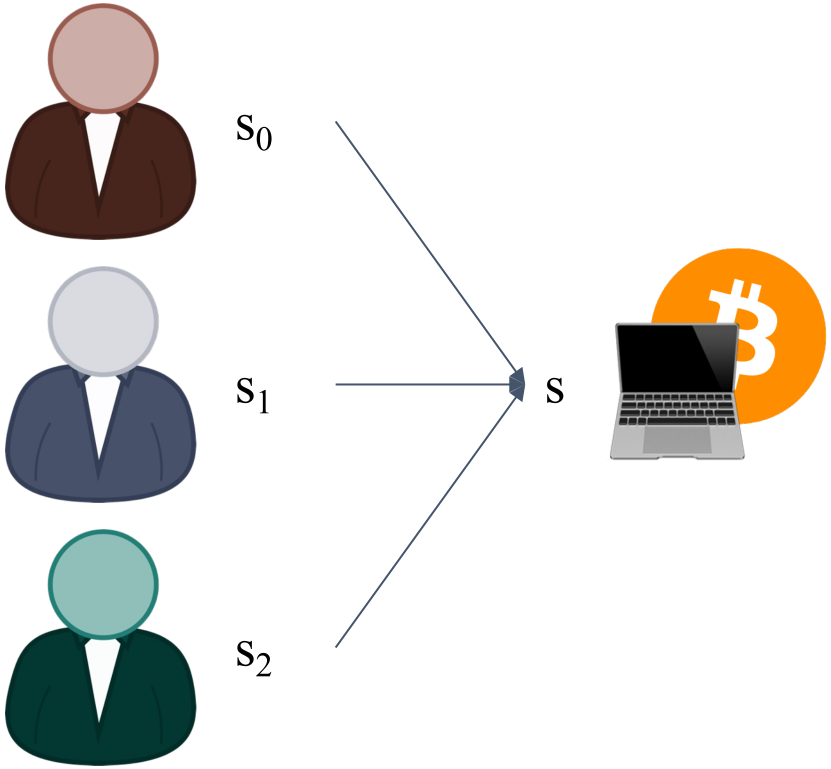


- Each user is given a share, which is a point of f. Any **two** users can recover f.

$$s_1 = (2,62),\ s_2 = (1,52) \Rightarrow f(x) = (62\text{-}52)/(2\text{-}1)\ x + (52\text{-}10) \Rightarrow f(x) = 10x + 42 \Rightarrow \mathbf{s = f(0) = 42}$$

TU Delft

t-Out-Of-n Secret Sharing



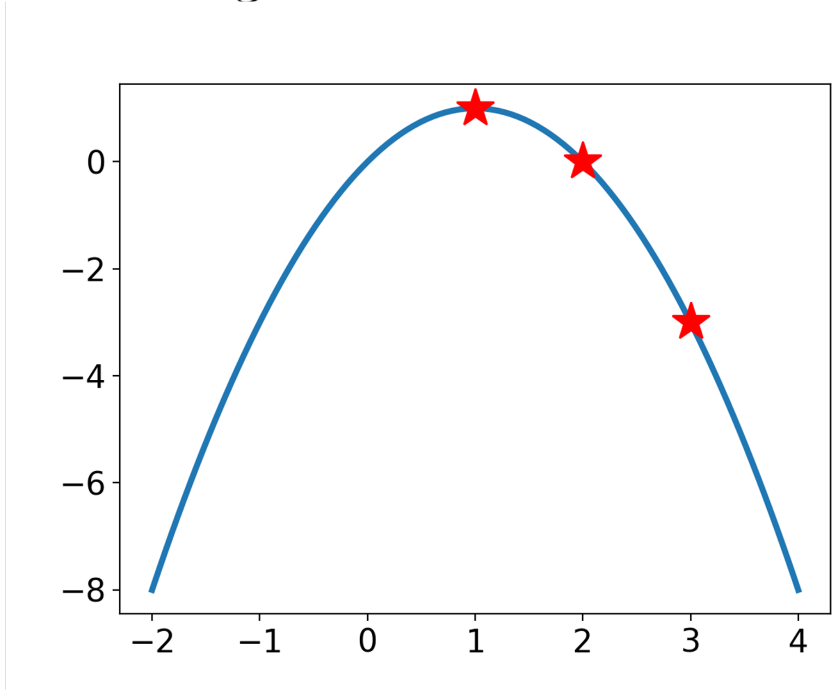
Any t shares are sufficient to recover s.

Polynomials

t district points define exactly **one polynomial** of degree t-1

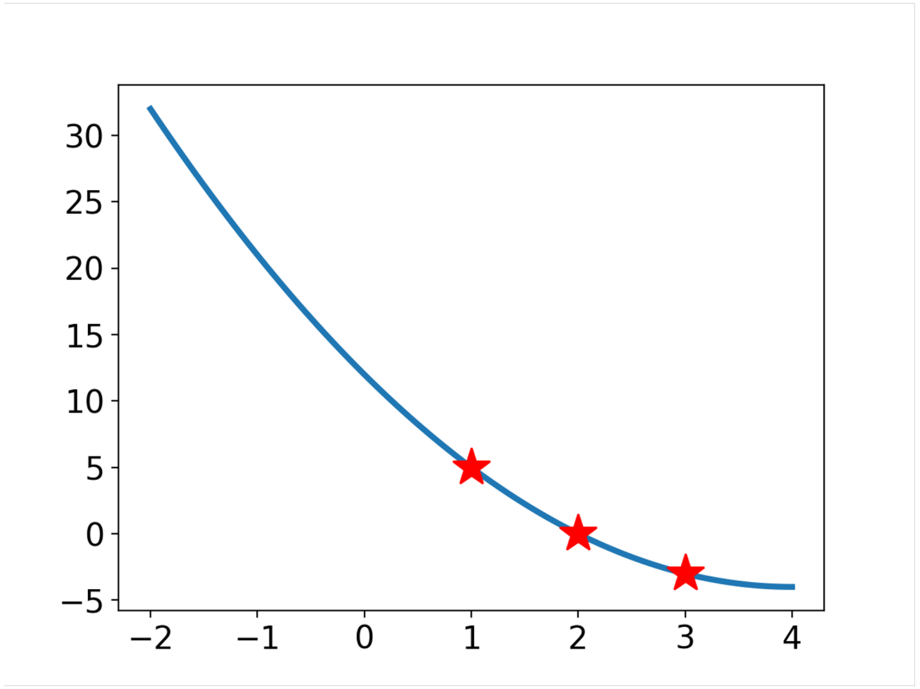
Points: (1,1), (2,0), (3,-3)

$$y = 2x - x^2$$



Points: **(1,5)**, (2,0), (3,-3)

$$y = x^2 - 8x + 12$$



TU Delft

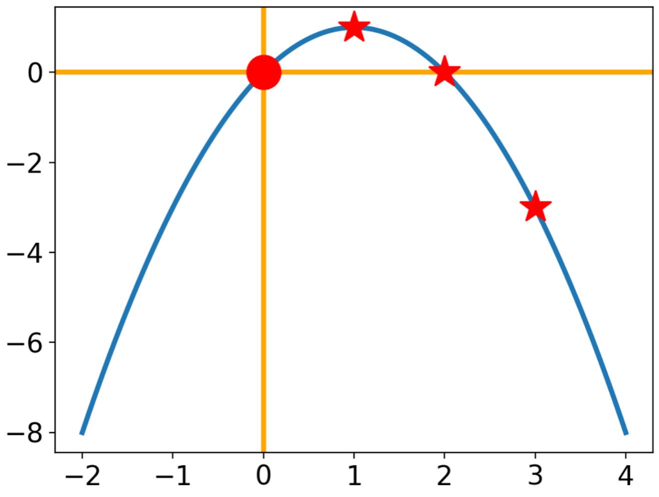
Information Theoretic Security:

No information about s can be recovered from fewer than t shares.

With fewer than t shares all possible secrets are equally likely.

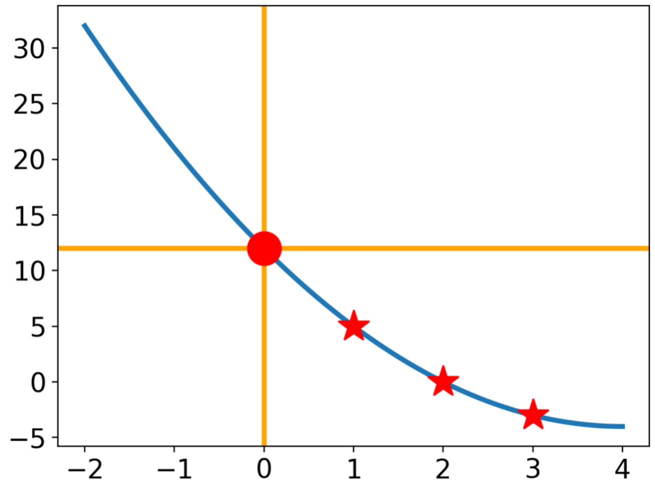
(1,1), (2,0), (3,-3)

s = 0



(1,5), (2,0), (3,-3)

s = 12



Secret Sharing

A t-out-of-n secret sharing scheme:

- **Share(s)** returns $s_1, s_2, .. , s_n$
- **Recover(s_k, … , s_{k+t-1})** returns s

Correctness:
For any subset S_t of Share(s) of size t:

$$\text{Recover}(S_t) = s$$

Security:
Using any subset of Share(s) of size **smaller than** t, absolutely **nothing** can be learnt about s.

Shamir Secret Sharing

A Trusted Dealer must compute Share(s).

Share(s):

1. Pick t-1 random numbers.
   (Sample a_1, a_2, .. , a_{t-1} uniformly at randomly.)
2. Define polynomial:

$$P(x) = s + \sum_{i=1}^{t-1} a_i x^i$$

3. Return n shares, each being **a point** on the polynomial.

$$s_i = (x_i, P(x_i))$$

Shamir Secret Sharing

Recover(s_k, … , s_{k+t-1}):

Find a polynomial q of degree t-1 that s_k, … , s_{k+t-1} are points of q. ($s_i = (x_i, y_i)$)

Lagrange interpolation

$$q(x) = \sum_{i=k}^{k+t-1} y_i \prod_{\substack{j \neq i}}^{k \leq j \leq t+k-1} \frac{x - x_j}{x_i - x_j}$$

Recover secret :

$$s = q(0)$$

Shamir Secret Sharing

Advantages

- Information theoretic security
- “Small” shares
- Security can be adjusted by updating the polynomial (and re-issuing shares)
- A different number of shares can be issued to each user

Drawbacks

- Participants can cheat.
  - Shares could be incorrect
  - No one knows if the secret is correct or not
- Total Trust in the Dealer

Applications: End-to-End Encryption

- Uses
  - Messaging
  - Secure Computing on the Cloud
- Private Key
  - The key should not be on the Server
  - What if the user loses the key?
- Secret Sharing!
  - Give shares to n friends (or colleagues), which can help recover key if it’s lost.

Applications: Cryptocurrency

- Uses
  - Bitcoin
  - Zcash
- Private Key
  - Key protects assets
  - If given to a server, the server now controls the assets.
- Secret Sharing!
  - Give shares to n friends (or colleagues), which can help recover key if it’s lost.

Applications: DNS

- **DNS**: Maps domain names to addresses: www.tudelft.nl ⇒ 54.73.174.150
- **DNSSEC:** Authenticates the mapping to protect against attacks, like poisoning the responses.
- In case of catastrophic failure, the master cryptographic keys need to be recovered.
  - 7 Recovery Key Share Holders hold a share of the key.
  - 5 needed for recovery.

Richard Lamb, program manager for DNSSEC at ICANN (Internet Corporation for Assigned Names and Numbers):

> *If you round up five of these guys, they can decrypt [the root key] should the West Coast fall in the water and the East Coast get hit by a nuclear bomb.*

Shamir Secret Sharing

t-out-of-**n** secret sharing:

- Split a secret **s** into **n** shares such that:
  - Any fewer than **t** shares reveal nothing about the secret
  - **t** shares reveal the secret **s**

Observation: **t** points uniquely define a polynomial of degree **t-1**

- Shares: points on a polynomial **P** of degree **t-1**
- Secret: **P**(0)