

CS 578 – Cryptography

Prof. Michael Backes

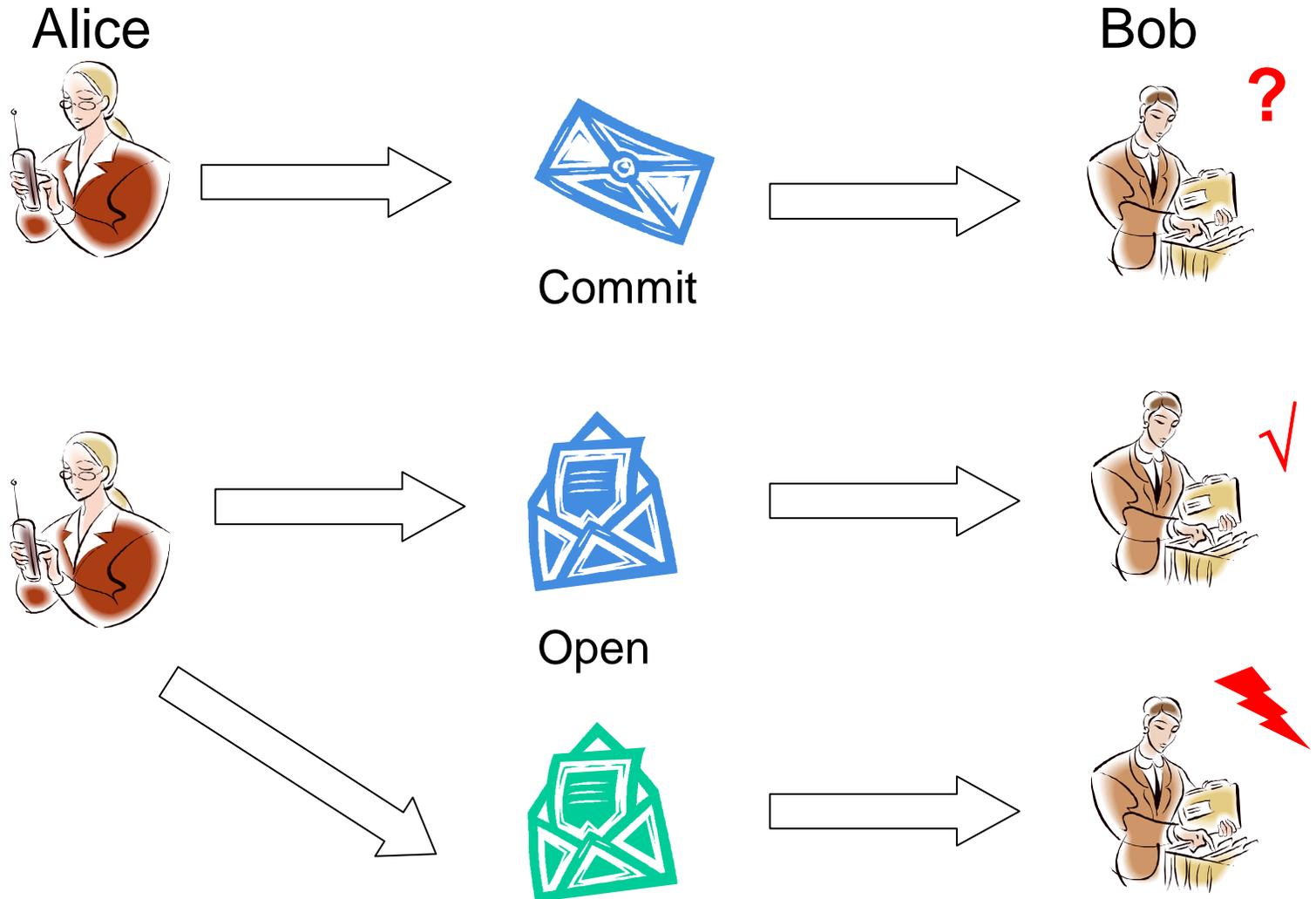
**Commitment Schemes,
Secret Sharing**

July 7, 2006

Administrative Announcements

- Handouts
 - “Secret Sharing” on Tuesday

Recall: Commitment Schemes



Recall: Commitment Schemes

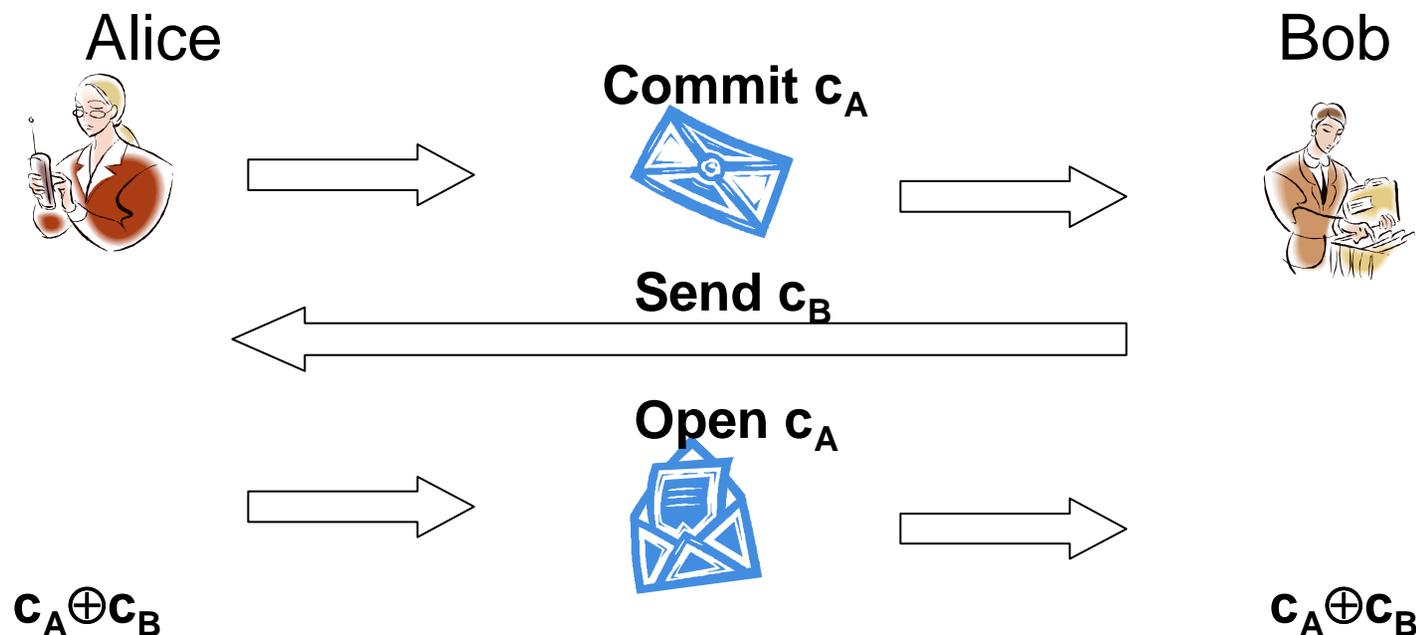
- Definition (**Commitment Scheme**): A (two-party) commitment scheme consists of two interactive algorithms (commit, open):
 - **commit**: The committer receives a message m ; the recipient obtains no input. Neither committer nor recipient make any output except for a value $\text{acc} \in \{\text{ok}, \text{error}\}$.
 - **open**: The same committer and recipient (maintaining state from the commit phase) take part. No inputs are needed. Recipient either outputs “(accept, m)” for some m or “reject”.

Recall: Commitment Schemes

- Properties of a commitment scheme
 - **Correctness**: If both committer and recipient are honest and run *commit* (on m) and then *open*, then the recipient outputs (accept, m).
 - **Binding**: Even if the committer is dishonest, he cannot open the commitment in two different ways, i.e., after a successful *commit*, causing the recipient to output (accept, m) or (accept, m') in the *open* protocol.
 - **Hiding**: The *commit* protocol should not give the recipient any information on m .

Application: Coin Tossing

- Idea: Alice and Bob throw a coin c_A , c_B and compute $c_A \oplus c_B$.
- But: c_A , c_B may be correlated!



Recall: An Impossibility Result

- **Theorem:** No commitment scheme can be information-theoretically binding and information-theoretically hiding.
- (But information-theoretically binding and computationally hiding is possible, and vice versa)

Recall: The Quadratic Residuosity Scheme

- Quadratic residuosity scheme (for single bits)
- *commit* (on m):
 - Committer randomly chooses n -bit primes p, q with $p \equiv 3 \pmod{4}$ and $q \equiv 3 \pmod{4}$, and sets $N := pq$.
Committer then chooses $y \leftarrow_{\mathcal{R}} Z_N^*$, computes
$$x := (-1)^m \cdot y^2 \pmod{N}$$
and outputs $\text{com} = (N, x)$.
- *Open*:
 - Committer sends (m, p, q, y) .
 - Recipient checks if p, q primes with $p \equiv 3 \pmod{4}$ and $q \equiv 3 \pmod{4}$, and if $N = pq$. Then he checks if $x = (-1)^m \cdot y^2 \pmod{N}$. If so, he outputs (accept, m) , else reject.

Recall: The Quadratic Residuosity Scheme

- **Theorem:** The quadratic residuosity commitment scheme is information-theoretically binding, and it is computationally hiding under the quadratic residuosity assumption.

[proof on the board]

- Extension to multiple bits easily doable

Information-theoretically Hiding Schemes

- Discrete Logarithm scheme (for messages from G_q)
- *commit* (on m):
 - Recipient picks random n -bit prime q , random $n_p(n)$ -bit prime p such that $q \mid p-1$ and two random elements $g, h \in Z_p^*$ of order q . Recipient sends (p, q, g, h) to the committer.
 - Committer verifies that p and q are prime and that g and h are of order q .
 - Committer then chooses $k \leftarrow_R \{1, \dots, q\}$ and outputs
$$\text{com} := g^m h^k \text{ mod } p$$
- *Open*:
 - Committer sends (m, k) .
 - Recipient verifies that $\text{com} = g^m h^k \text{ mod } p$. If so, he outputs (accept, m) , else reject

The Discrete-Logarithm Scheme

- **Theorem:** The discrete logarithm commitment scheme is information-theoretically hiding, and it is computationally binding under the Discrete Logarithm assumption for subgroups G_q of Z_p^* .

[proof on the board]

- Very efficient scheme:
 - Commit only requires two exponentiations with exponent of size $\log q$
 - Commitment only one element of size $\log p$

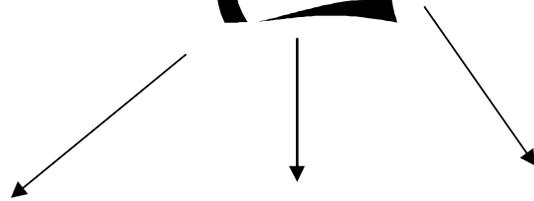
The Discrete-Logarithm Scheme

- Special properties of the scheme
 - The discrete logarithm commitment scheme is homomorphic if several values are committed using the same value $pk_{REC} = (p, q, g, h)$:
If a committer has made two commitments com_1 and com_2 with respect to the same value $pk_{REC} = (p, q, g, h)$, he can open the product $com_1 \cdot com_2$ as a commitment to $m_1 + m_2 \pmod q$.
Opening the product commitment does not reveal additional information about the individual contents.

[proof on the board]

Secret Sharing

Secret:



Availability:



=



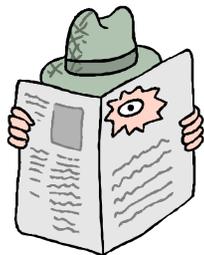
+



+



Secrecy:



+



+



=

?

Coca-Cola trade secrets 'stolen'

US prosecutors have charged three people with stealing secrets from soft drinks company Coca-Cola and trying to sell them to its main rival PepsiCo.



[Coca Cola's] chief executive, Neville Isdell, said that "information is the lifeblood of the company".

Mr Isdell said that Coca-Cola would be reviewing its security procedures.

www.bbc.co.uk, 7/6/06

Police strike at heart of mafia

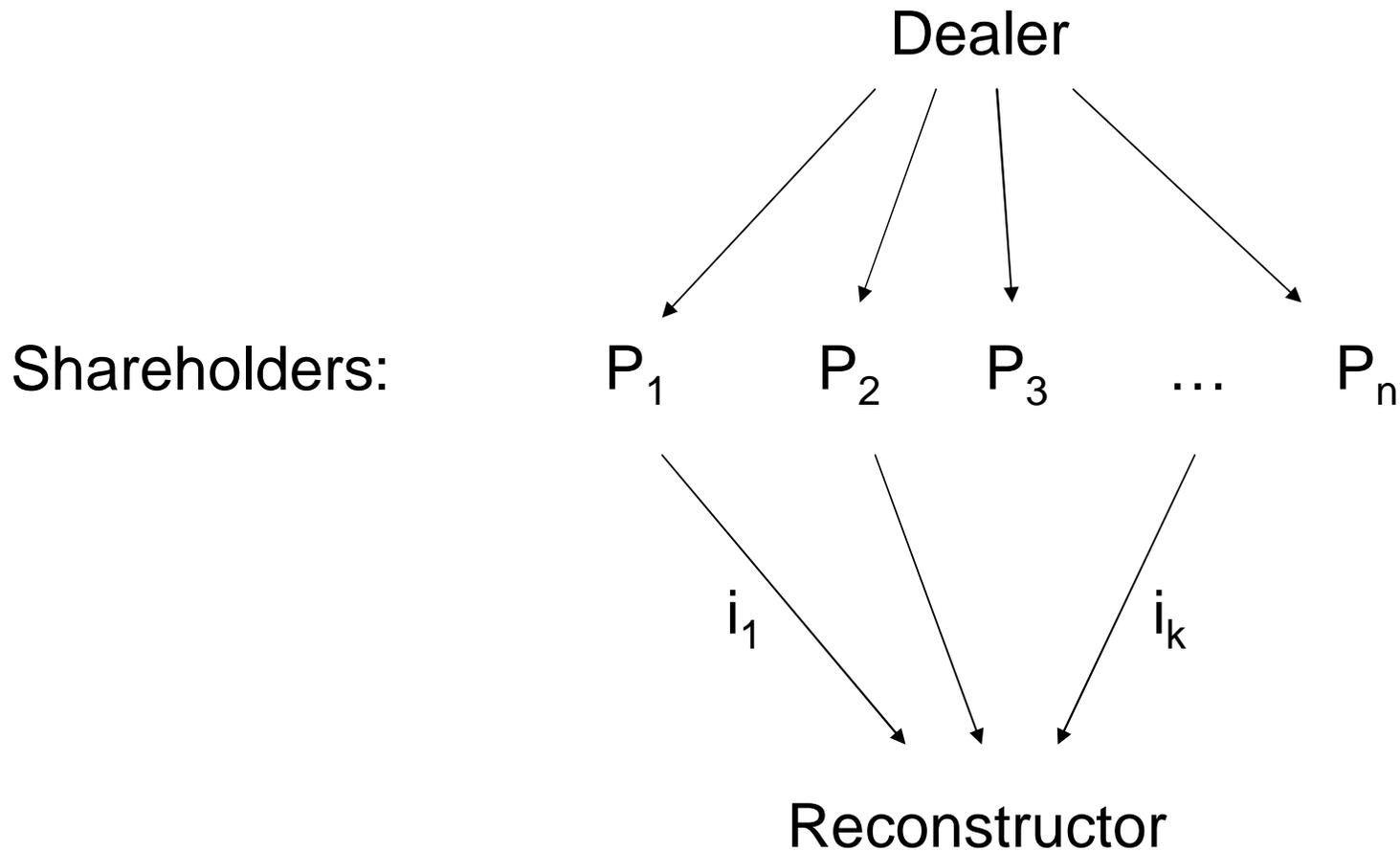


A bloody mafia power struggle appears to have been narrowly averted after dawn raids across the Sicilian capital of Palermo on Tuesday netted 24 alleged Cosa Nostra "godfathers", including a mobster said by investigators to be trying to take over as the next capo di tutti i capi, or "boss of all bosses".

“It came two months after the seizure of the crime syndicate's fugitive commander, **Bernardo Provenzano**, and was made **possible, in part, by evidence found at his hideout.**”

www.guardian.co.uk, 6/20/2006

Secret Sharing



Definition k-of-n Secret Sharing

- Participants:
 - Dealer, Reconstructor, (n) Shareholders P_1, \dots, P_n
- Algorithms
 - a set of secrets S
 - $\text{Share}(s) = s_1, \dots, s_n \in S_i$
 - $\text{Reconstruct}(s_{i_1}, \dots, s_{i_k}) = s^*$
- Requirements
 - **Availability:** $s^* = s$ if dealer, reconstructor, and the k shareholders are honest.
 - **Secrecy:** An adversary learning at most k values s_{ij} gains *no* information (information theoretical) about the secret s

Simple Examples

- Bad example: pas-----
 ---SWO--
 -----rd

- Good Example (n-of-n)

- To share bit b:

$$r_1, r_2 \leftarrow_{\mathbb{R}} \{0, 1\}$$

$$r_3 := b \oplus r_1 \oplus r_2$$

- Shares: r_1, r_2, r_3

XOR-scheme

- Availability:

$$r_1 \oplus r_2 \oplus r_3 = r_1 \oplus r_2 \oplus (b \oplus r_1 \oplus r_2) = b$$

- Secrecy: Given any two shares s_1, s_2 , this can be opened to *any* secret b :

$$s_1 \oplus s_2 \oplus (s_1 \oplus s_2 \oplus 0) = 0$$

$$s_1 \oplus s_2 \oplus (s_1 \oplus s_2 \oplus 1) = 1$$

Algebraic Fact

- **Fact:** Let p be a polynomial of degree $k-1$ over a field F . If p has k zero points, then $p \equiv 0$.
- **Corollary:** Let pol and pol' be polynomials of degree $k-1$ that coincide at k points. Then these polynomials are equal.

Proof: Let x_1, \dots, x_k such that

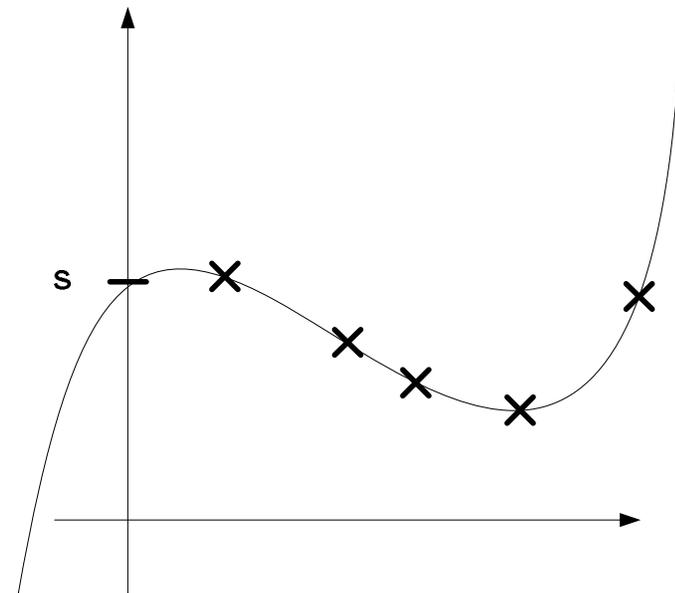
$$\text{pol}(x_i) = \text{pol}'(x_i) \quad \Rightarrow (\text{pol} - \text{pol}')(x_i) = 0$$

$$\Rightarrow (\text{pol} - \text{pol}') \equiv 0$$

$$\Rightarrow \text{pol} \equiv \text{pol}'$$

Shamir's scheme (k-of-n)

- Parameters:
 - A finite field F where $|F| > n$, and a set $S \subseteq F$ of secrets, for simplicity $S = F = \mathbb{Z}_p$
- For each participant i , fix $0 \neq x_i \in \mathbb{Z}_p$ such that $x_i \neq x_j$ for $i \neq j$ (publicly known), e.g., $x_i = i$



Shamir: Sharing

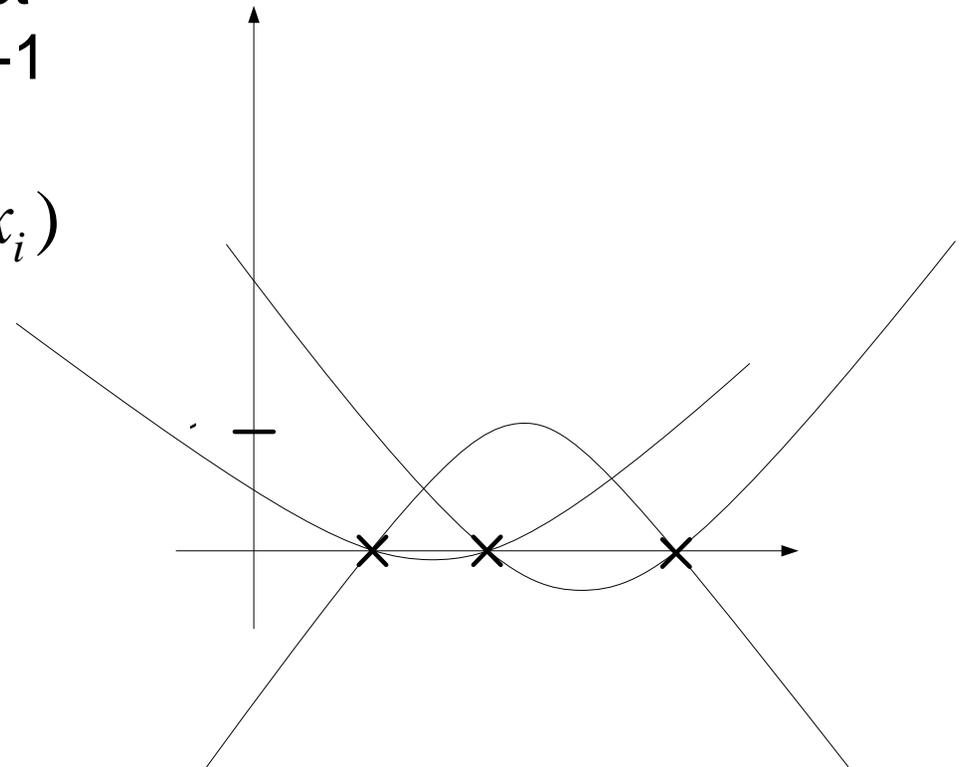
- Sharing:
 - choose random polynomial of degree $k-1$ (over Z_p), i.e.,
$$a_i \leftarrow_{\mathbb{R}} Z_p \quad (i=1, \dots, k-1),$$
$$\text{pol}(x) = a_{k-1}x^{k-1} + \dots + a_1x^1 + s$$
 - Compute
$$y_i := \text{pol}(x_i)$$
 - Give participant i the value y_i

Shamir: Reconstruction

- Let $Avail = \{i_1, \dots, i_k\}$ the set of indices of available shares.
- For all $i^* \in Avail$, construct polynomials of degree $k-1$

$$cpol_{i^*}(x) := \prod_{i \in Avail \setminus \{i^*\}} (x - x_i)$$

$$bpol_{i^*}(x) := \frac{cpol_{i^*}(x)}{cpol_{i^*}(x_{i^*})}$$



Shamir: Availability

- Let

$$\text{pol}^*(x) := y_{i_1} \text{bpol}_{i_1}(x) + \dots + y_{i_k} \text{bpol}_{i_k}(x)$$

- For all $i \in \{i_1, \dots, i_k\}$ we have

$$\text{pol}^*(x_i) = y_i$$

[proof on the board]

- A pol and pol^* are of degree $k-1$ and coincide in k point, thus

$$\text{pol}(x) = \text{pol}^*(x),$$

in particular

$$s = \text{pol}(0) = \text{pol}^*(0) = s^*$$

Shamir: Secrecy

- (Information-theoretical) secrecy:
For all $k-1$ indices i_1, \dots, i_{k-1} ,
for all shares $s_{i_1}, \dots, s_{i_{k-1}}$, and
for all secrets s ,
there exists a unique polynomial pol^* which is
consistent with these values

[proof on the board]

Shamir: Improvements

- Reconstruction: No need to reconstruct the whole polynomial:

$$s = \sum_{i \in Avail} y_i \cdot bpol(0)$$

where

$$bpol_i(0) = \prod_{i \in Avail \setminus \{i^*\}} \frac{-x_i}{x_{i^*} - x_i}$$

Shamir: Improvements (cont'd)

- Splitting long secrets:
 - complexity is quadratic in the size of the secret
 - Splitting the secret into parts of constant size
⇒ linear complexity
- Sharing several secrets:
 - The polynomials bp_{ol_i} can be re-used

Shamir: Optimality

- Shares are as big as the secret, can one do better?

No!

- One can proof that this is optimal, i.e.,

$$|S| \leq |S_i|$$

[Proof sketch]

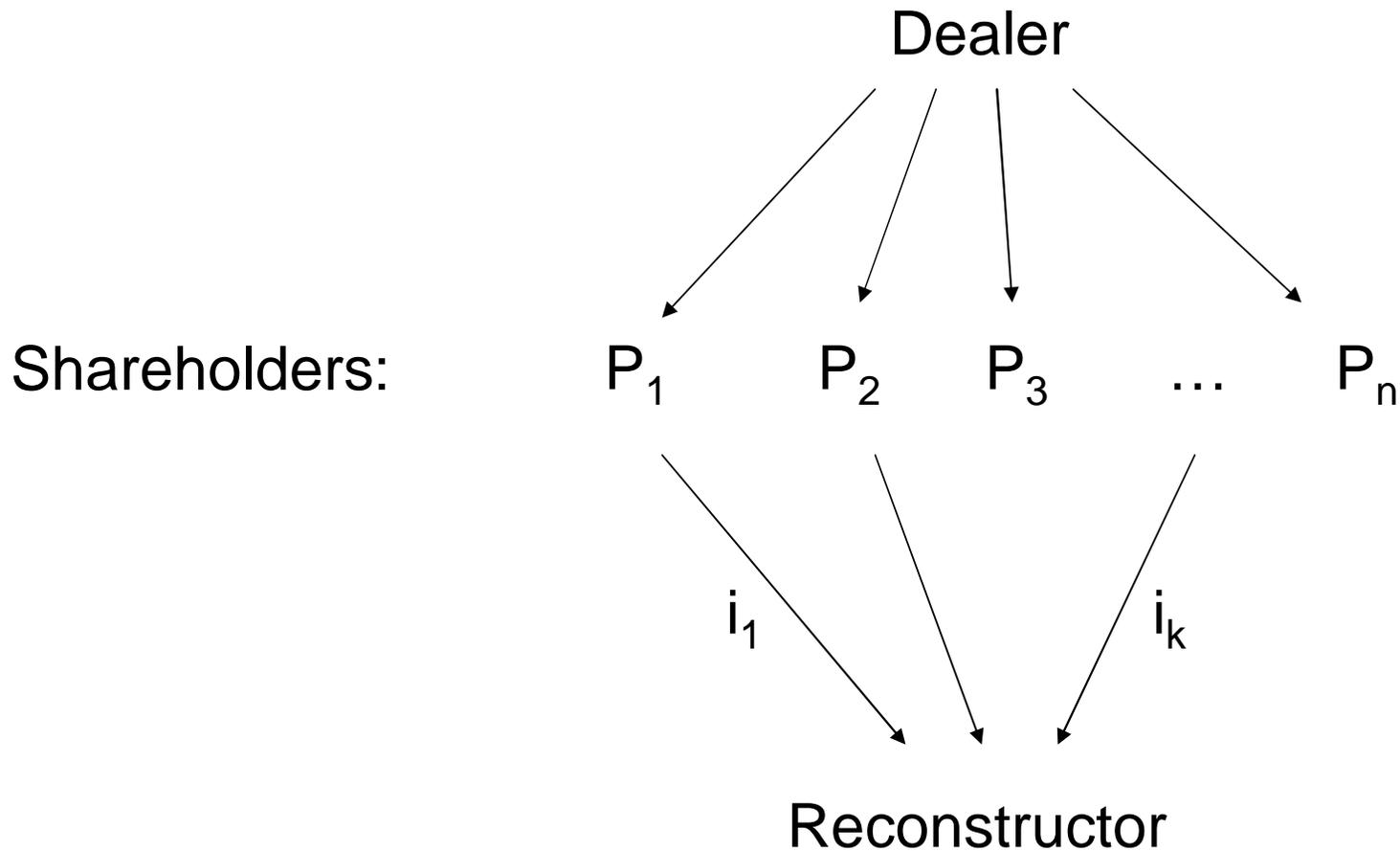
Shamir: Computing with Secrets

- Let two shares be given
 $s = \text{pol}(0)$ and $s' = \text{pol}'(0)$,
 $y_i = \text{pol}(x_i)$ and $y'_i = \text{pol}'(x_i)$
- Then the $y_i + y'_i$ are shares for $s + s'$
[proof on the board]
- For $c \in \mathbb{Z}_p$: The $c \cdot y_i$ are shares for $c \cdot s$
[proof on the board]

Proactive Secret Sharing

- If one share gets compromised?
- Not practical to change the secret, so the uncompromised shares need to be renewed.
- Dealer generates a new random polynomial with constant term zero
- Computes a new $y'_i := \text{pol}^*(x_i)$, shareholder adds y'_i and his y_i
- Attacker can only recover the secret if he can find enough other non-updated shares to reach the threshold.

Verifiable Secret Sharing



Verifiable Secret Sharing (VSS)

- Malicious shareholders: If one uses a wrong share, the secret cannot be reconstructed
- Malicious dealer: Can output inconsistent shares

⇒ Pedersen's scheme

Pedersen: Sharing

- Create a Shamir share ($a_0=s$):
$$\text{pol}(x) = a_{k-1}x^{k-1} + \dots + a_1x^1 + a_0,$$
$$y_i := \text{pol}(x_i)$$
- The dealer commits to the a_i using the DLog scheme, i.e.,
$$b_i \leftarrow_{\mathcal{R}} \mathbb{Z}_p, \text{com}_i = g^{a_i} h^{b_i} \quad (i=0, \dots, n),$$
these commitments are broadcasted to everyone.
- Let
$$\text{pol}' := b_{k-1}x^{k-1} + \dots + b_1x^1 + b_0$$
$$z_i := \text{pol}'(x_i)$$
- Shareholder i gets y_i, z_i

Pedersen: Sharing (cont'd)

- Shareholder verifies that

$$g^{y_i} h^{z_i} = com_{k-1}^{x_i^{k-1}} \cdot \dots \cdot com_1^{x_i^1} \cdot com_0 =: comshare_i$$

[proof on the board]

Pedersen: Reconstruction

- The reconstructor tests if

$$g^{y_i} h^{z_i} = comshare_i$$

- If this is correct, then reconstruct the secret from the y_i 's as before