Unconditionally-Secure Robust Secret Sharing with Compact Shares

Serge Fehr

CWI Amsterdam

www.cwi.nl/~fehr

Alfonso Cevallos

Leiden University

Rafail Ostrovsky

UCLA

Yuval Rabani

Hebrew University of Jerusalem

(t-out-of-n) Secret Sharing

secret: s

$$\Downarrow$$

shares: $s_1 \quad s_2 \quad \cdots \quad s_n$

- **Privacy**: any t shares give no information on s

$$s_1 \quad s_2 \quad \cdots \quad s_t \quad \longrightarrow \quad ?$$

- **Reconstructability**: any $t+1$ shares uniquely determine s

$$s_1 \quad s_2 \quad \cdots \quad s_{t+1} \quad \longrightarrow \quad s$$

Shamir's Secret Sharing Scheme [Sha79]

secret: $s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

shares: $s_1 = f(x_1) \quad \cdots \quad s_n = f(x_n)$

- **Privacy** and **reconstructability** follow from Lagrange interpolation

Shamir's Secret Sharing Scheme [Sha79]

secret: $s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

shares: $s_1 = f(x_1) \quad \cdots \quad s_n = f(x_n)$

- **Privacy** and **reconstructability** follow from Lagrange interpolation

- Here and in general:
  reconstructability requires correct shares

Robust Secret Sharing

secret: s

shares: $s_1 \quad s_2 \quad \cdots \quad s_n$

- **Privacy**: any t shares give no information on s

$$s_1 \quad \cdots \quad s_t \quad \longrightarrow \quad ?$$

- **Reconstructability**: any $t+1$ shares uniquely determine s

$$s_1 \quad \cdots \quad s_{t+1} \quad \longrightarrow \quad s$$

Robust Secret Sharing

secret: s

shares: $s_1 \quad s_2 \quad \cdots \quad s_n$

Note:
assume dealer to be honest

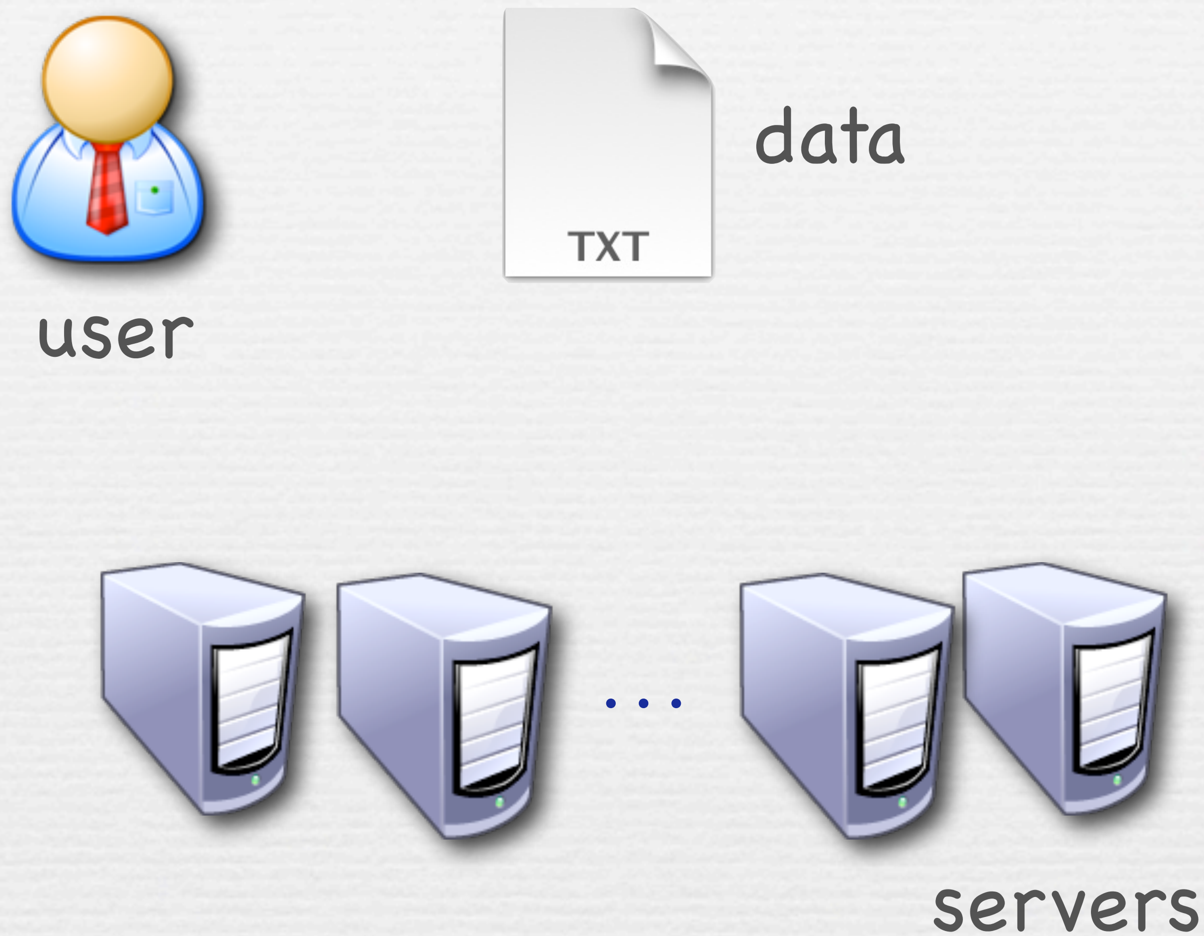
- **Privacy**: any t shares give no information on s

$$s_1 \quad \cdots \quad s_t \quad \longrightarrow \quad ?$$

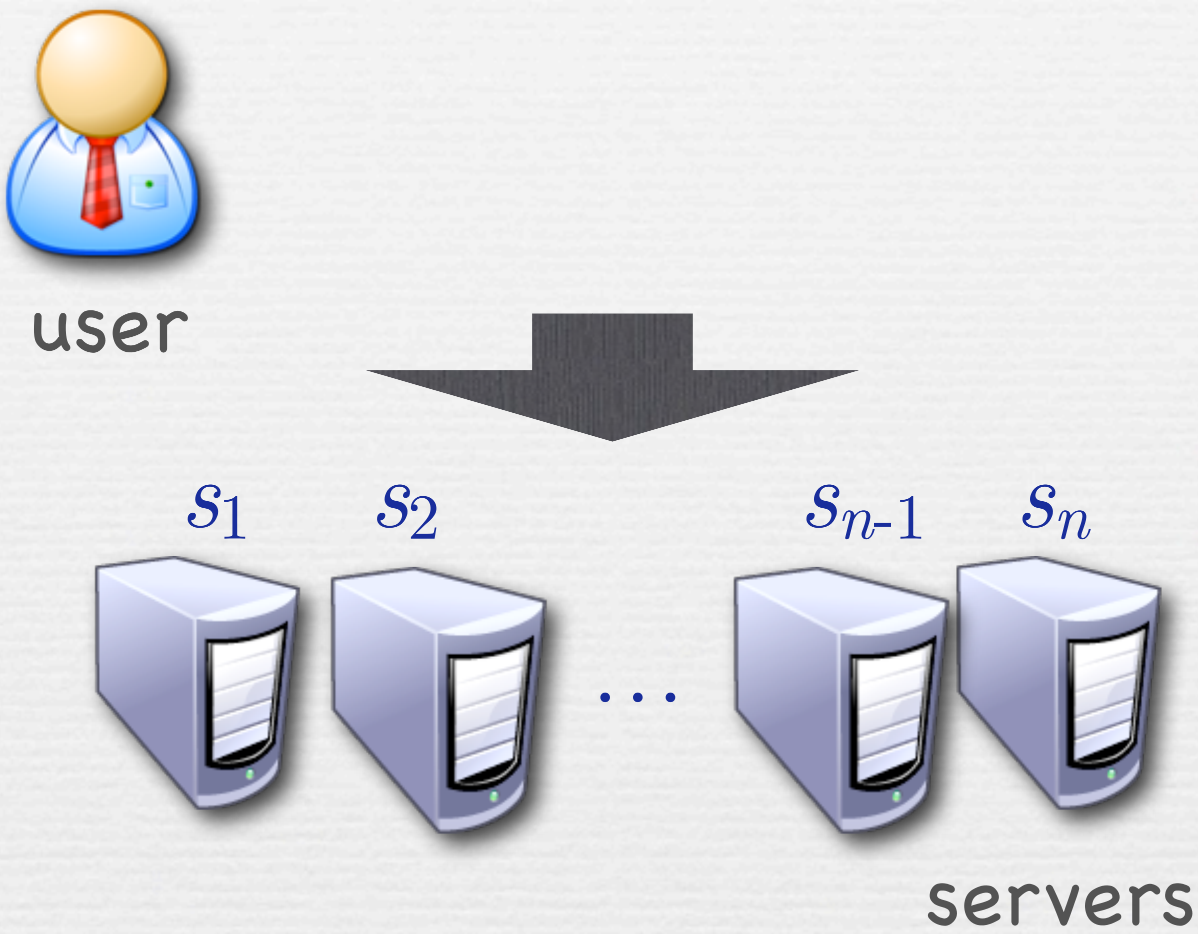
- **Robust reconstructability**:
  the set of all n shares determines s, even if t of them are faulty

$$\hat{s}_1 \quad \cdots \quad \hat{s}_t \quad s_{t+1} \quad \cdots \quad s_n \quad \longrightarrow \quad s$$

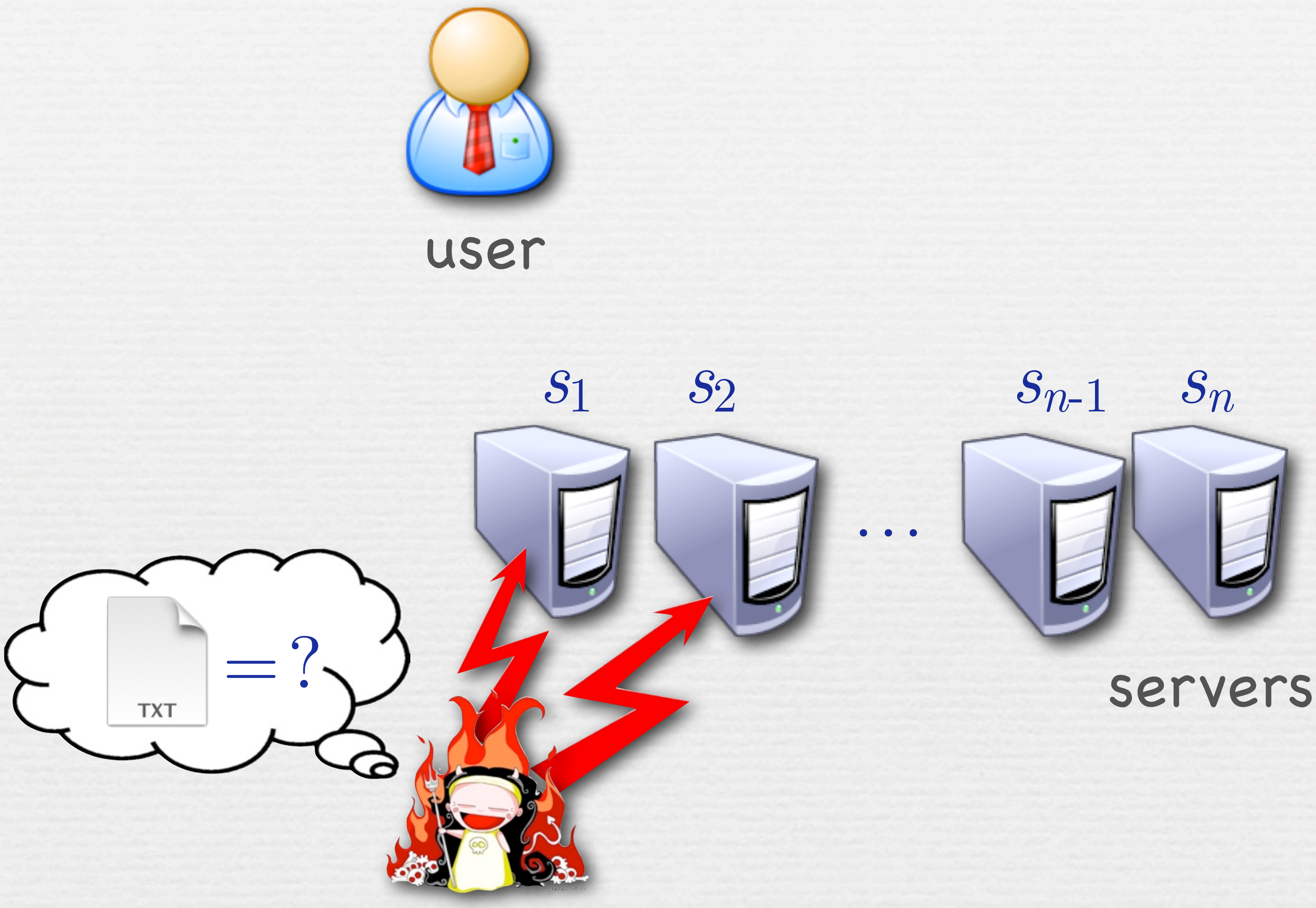
Application: Secure Data Storage



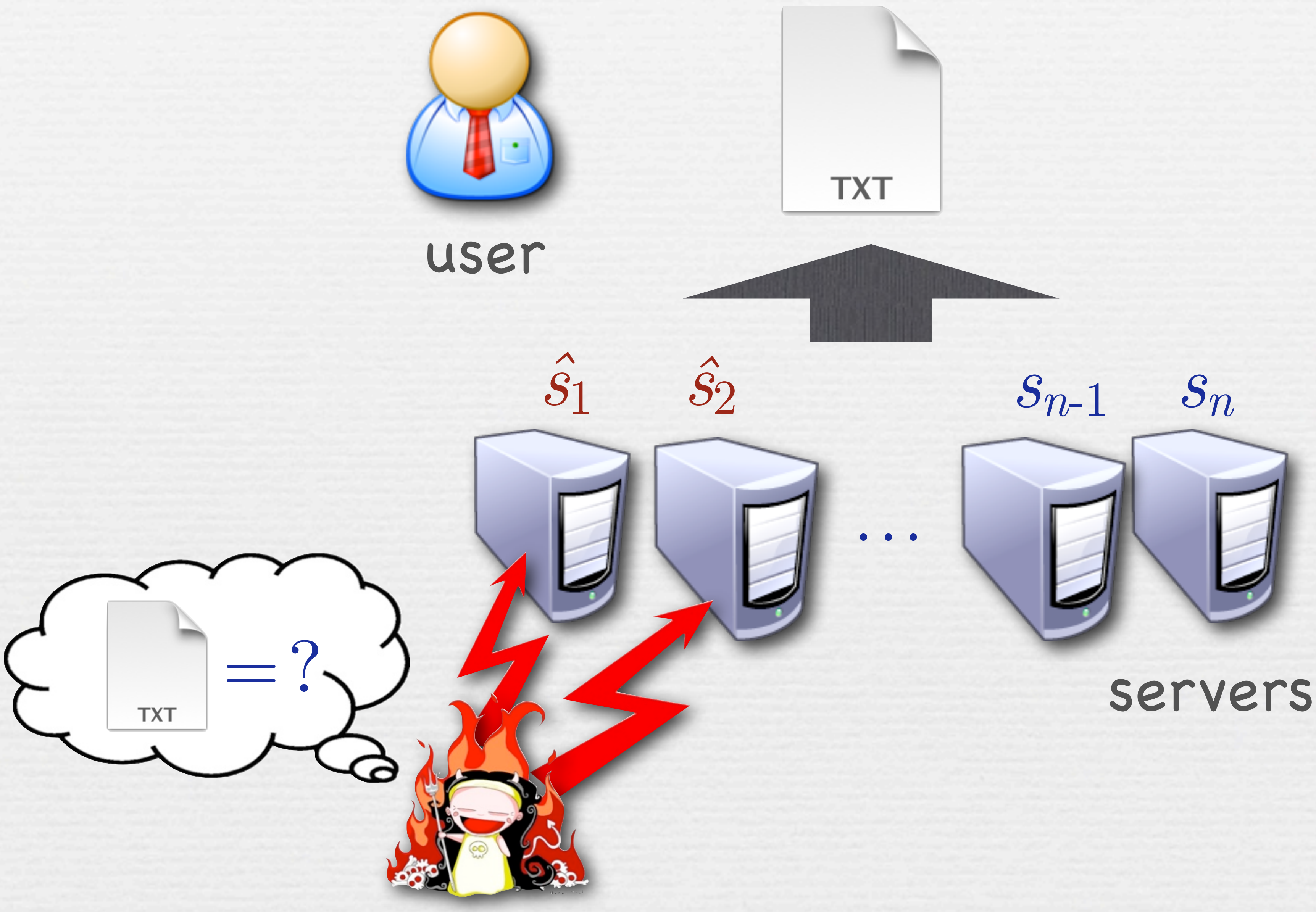
Application: Secure Data Storage



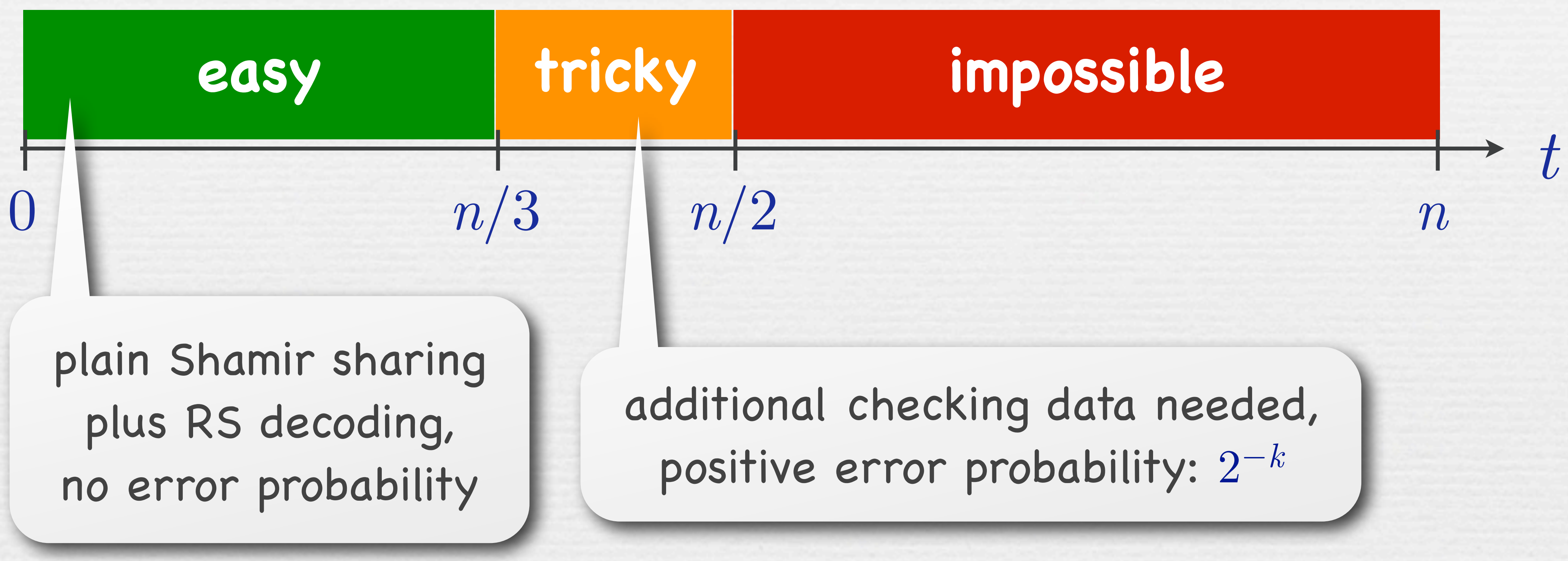
Application: Secure Data Storage



Application: Secure Data Storage



(Im)possibility



(Im)possibility

This work: $n = 2t+1$, with unconditional security

plain Shamir sharing plus RS decoding, no error probability

additional checking data needed, positive error probability: 2^{-k}

Known Results vs Our Result

- Rabin & Ben-Or (1989):
  - Overhead in share size: $O(k \cdot n \cdot \log n)$ ☹
  - Computational complexity: $poly(k, n)$ ☺

Known Results vs Our Result

- Rabin & Ben-Or (1989):
  - Overhead in share size: $O(k{\cdot}n{\cdot}\log n)$ ☹
  - Computational complexity: $poly(k,n)$ ☺
- Cramer & F (2001), based on Cabello, Padró & Sáez (1999), generalized by Kurosawa & Suzuki (2009):
  - Overhead in share size: $O(k+n)$ ☺ (lower bound: $\Omega(k)$)
  - Computational complexity: $exp(n)$ ☹

Known Results vs Our Result

- Rabin & Ben-Or (1989):
  - Overhead in share size: $O(k \cdot n \cdot \log n)$ ☹
  - Computational complexity: $poly(k,n)$ ☺
- Cramer & F (2001), based on Cabello, Padró & Sáez (1999), generalized by Kurosawa & Suzuki (2009):
  - Overhead in share size: $O(k+n)$ ☺ (lower bound: $\Omega(k)$)
  - Computational complexity: $exp(n)$ ☹
- Our new scheme:
  - Overhead in share size: $O(k+n \cdot \log n)$ ☺
  - Computational complexity: $poly(k,n)$ ☺

Further Outline

- Introduction
- The (simple) case $t < n/3$
- The Rabin & Ben-Or scheme
- Our scheme
- Difficulties of the proof
- Conclusion

The (Simple) Case $n = 3t+1$

$$s \in \mathbb{F}$$

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$

$$s_1 = f(x_1) \quad \cdots \quad s_{t+1} \qquad s_{t+2} \quad \cdots \quad s_{2t+1} \qquad s_{n-t+1} \quad \cdots \quad s_n$$

The (Simple) Case $n = 3t+1$

$$s \in \mathbb{F}$$

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$

$s_1 = f(x_1)$ $\cdots$ s_{t+1} s_{t+2} $\cdots$ s_{2t+1} $\hat{s}_{n-t+1}$ $\cdots$ $\hat{s}_n$

The (Simple) Case $n = 3t+1$

$$s \in \mathbb{F}$$

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$

$s_1 = f(x_1)$ $\cdots$ s_{t+1} s_{t+2} $\cdots$ s_{2t+1} $\hat{s}_{n-t+1}$ $\cdots$ $\hat{s}_n$

$t+1$ correct shares
-> determines f

The (Simple) Case $n = 3t+1$

$s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

$s_1 = f(x_1) \quad \cdots \quad s_{t+1}$

$s_{t+2} \quad \cdots \quad s_{2t+1}$

$\hat{s}_{n-t+1} \quad \cdots \quad \hat{s}_n$

$t+1$ correct shares -> determines f

$r=t$ redundant correct shares

The (Simple) Case $n = 3t+1$

$s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

$s_1 = f(x_1) \quad \cdots \quad s_{t+1}$ | $s_{t+2} \quad \cdots \quad s_{2t+1}$ | $\hat{s}_{n-t+1} \quad \cdots \quad \hat{s}_n$

$t+1$ correct shares -> determines f

$r=t$ redundant correct shares

$e=t$ faulty shares

The (Simple) Case $n = 3t+1$

$s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

$s_1 = f(x_1) \quad \cdots \quad s_{t+1}$ | $s_{t+2} \quad \cdots \quad s_{2t+1}$ | $\hat{s}_{n-t+1} \quad \cdots \quad \hat{s}_n$

$t+1$ correct shares -> determines f

$r=t$ redundant correct shares

$e=t$ faulty shares

Reed-Solomon decoding: If $e \leq r$ (satisfied here) then

- f is uniquely determined from $s_1, \ldots, \hat{s}_n$
- f can be efficiently computed (Berlekamp-Welch)

The Rabin & Ben-Or Scheme ($n = 2t+1$)

$s \in \mathbb{F}$

Sharing phase:

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

$s_1 = f(x_1)$ $\quad \ldots \quad$ s_i $\quad \ldots \quad$ s_j $\quad \ldots \quad$ s_n

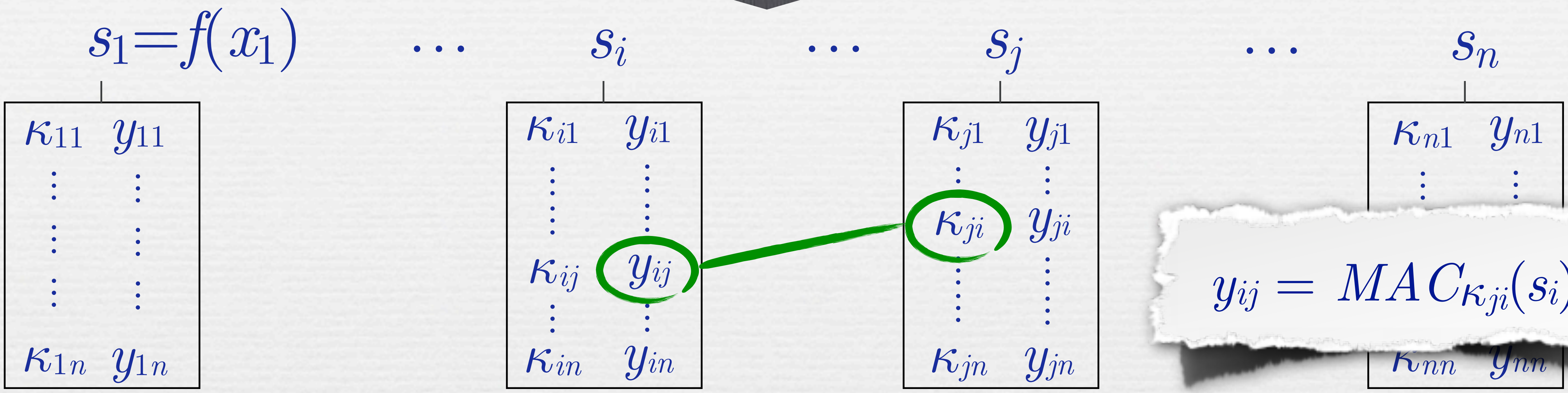
$s_1=f(x_1)$		s_i		s_j		s_n	
κ_{11}	y_{11}	κ_{i1}	y_{i1}	κ_{j1}	y_{j1}	κ_{n1}	y_{n1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	κ_{ji}	y_{ji}	$\vdots$	$\vdots$
$\vdots$	$\vdots$	κ_{ij}	y_{ij}	$\vdots$	$\vdots$	$\vdots$	$\vdots$
		$\vdots$	$\vdots$				
κ_{1n}	y_{1n}	κ_{in}	y_{in}	κ_{jn}	y_{jn}	κ_{nn}	y_{nn}

The Rabin & Ben-Or Scheme ($n = 2t+1$)

$$s \in \mathbb{F}$$

Sharing phase:

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$



$$y_{ij} = MAC_{\kappa_{ji}}(s_i)$$

The Rabin & Ben-Or Scheme ($n = 2t+1$)

$s \in \mathbb{F}$

Sharing phase:

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

$s_1 = f(x_1) \quad \ldots \quad s_i \quad \ldots \quad s_j \quad \ldots \quad s_n$

s_1		s_i		s_j		s_n	
κ_{11}	y_{11}	κ_{i1}	y_{i1}	κ_{j1}	y_{j1}	κ_{n1}	y_{n1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
		κ_{ij}	y_{ij}	κ_{ji}	y_{ji}		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		
κ_{1n}	y_{1n}	κ_{in}	y_{in}	κ_{jn}	y_{jn}	κ_{nn}	y_{nn}

$y_{ij} = MAC_{\kappa_{ji}}(s_i)$

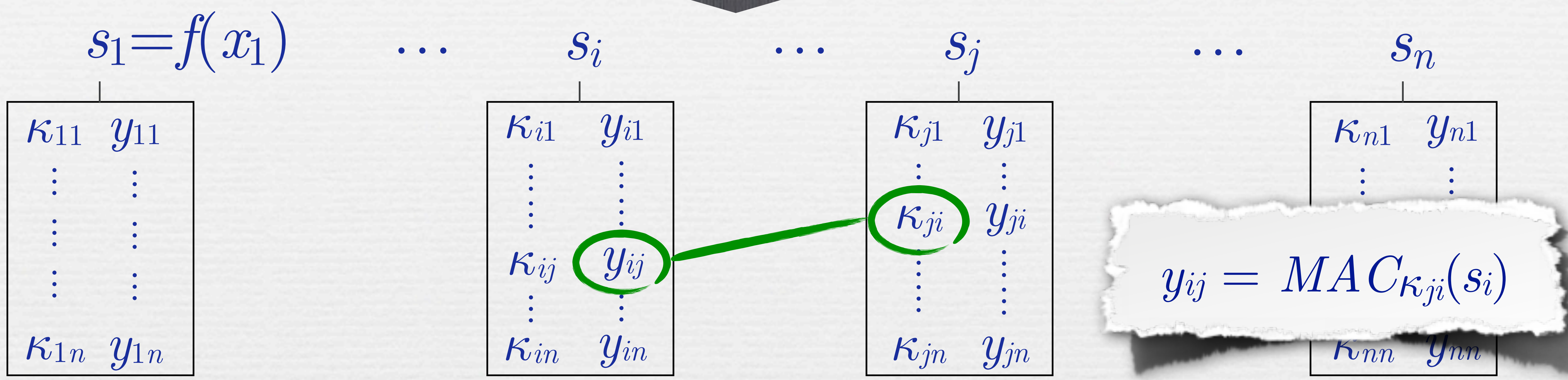
- MAC security: for any $\hat{s}_i \neq s_i$ and $\hat{y}_{ij}$: $P[\hat{y}_{ij} = MAC_{\kappa_{ji}}(\hat{s}_i)] \leq \varepsilon$.
- Example: $\kappa_{ij} = (\alpha_{ij}, \beta_{ij}) \in \mathbb{F}^2$ and $y_{ij} = MAC_{\kappa_{ji}}(s_i) = \alpha_{ij} \cdot s_i + \beta_{ij}$.
- For error probability $\varepsilon \leq 2^{-k}$:
  - bit size $|\kappa_{ij}|, |y_{ij}| \geq k$
  - overhead per share (above Shamir share): $\Omega(k \cdot n)$

The Rabin & Ben-Or Scheme ($n = 2t+1$)

$s \in \mathbb{F}$

Sharing phase:

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$



Reconstruction phase:

1. For every share s_i: accept s_i iff it is approved by $\geq t+1$ players,

   (meaning $\#\{j \mid y_{ij} = MAC_{\kappa_{ji}}(s_i)\} \geq t+1$)

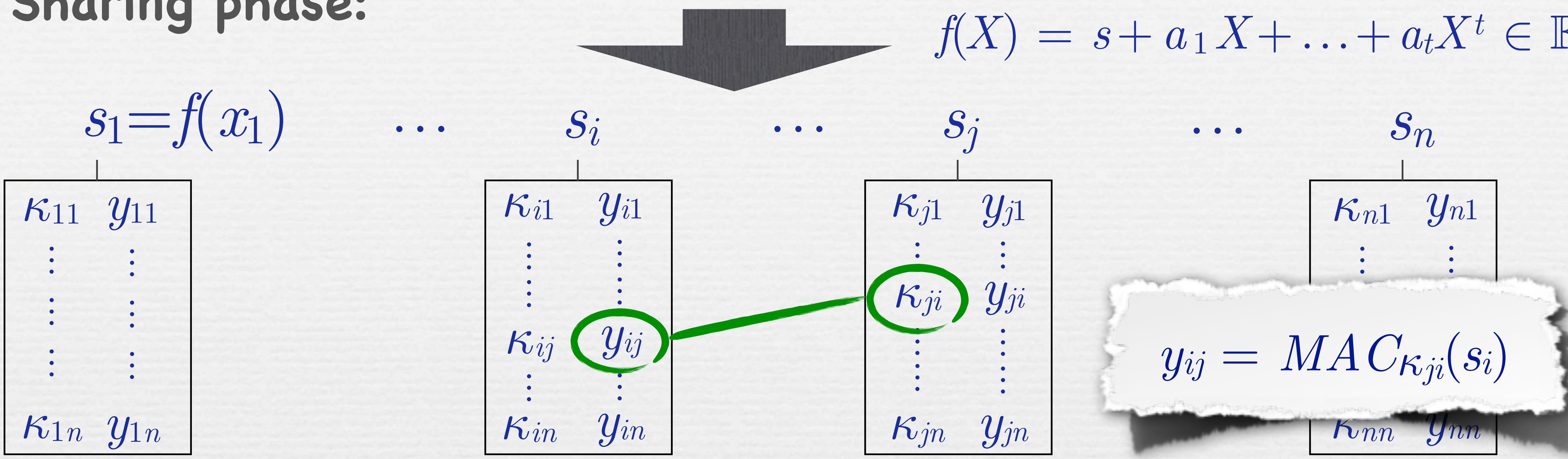
2. Reconstruct s using the accepted shares s_i .

The Rabin & Ben-Or Scheme ($n = 2t+1$)

$s \in \mathbb{F}$

Sharing phase:

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$

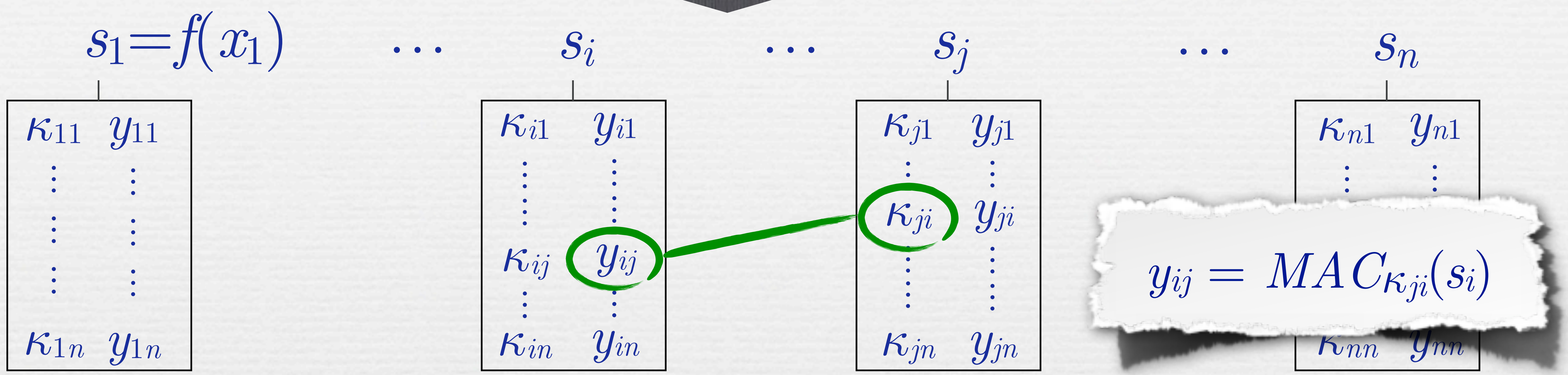


Our New Scheme

Sharing phase:

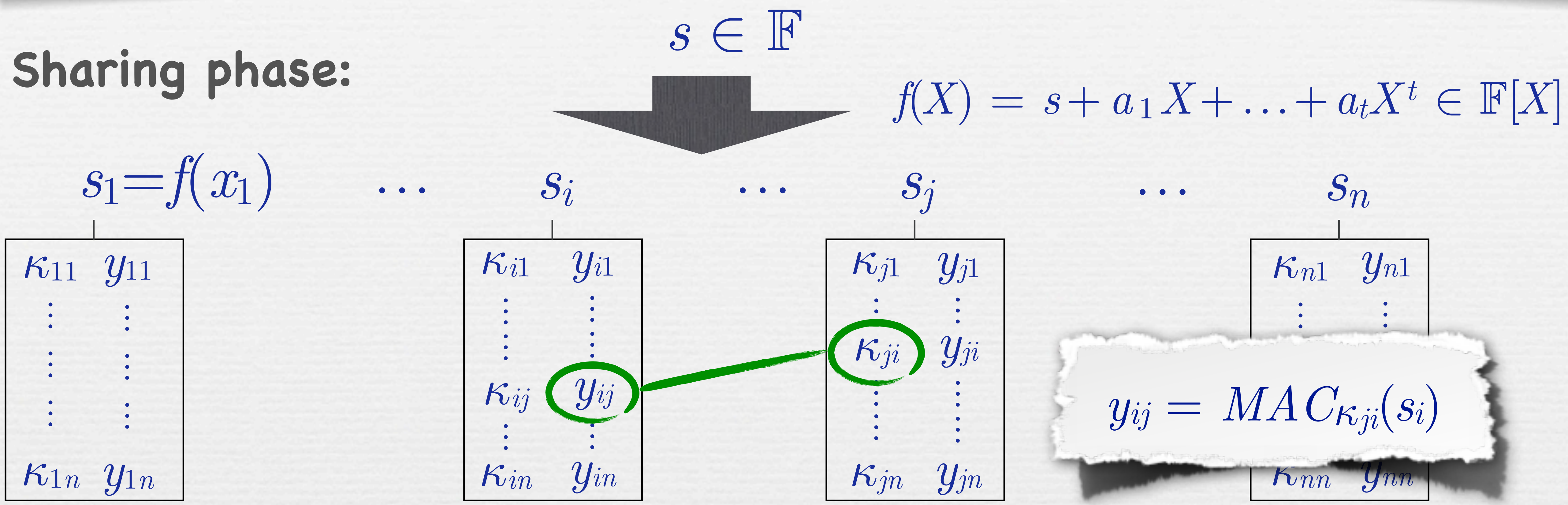
$$s \in \mathbb{F}$$

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$



Our New Scheme

Sharing phase:



- Use **small** tags and keys $|\kappa_{ij}|, |y_{ij}| = \tilde{O}(k/n + 1)$ (instead of $O(k)$)
- Gives: overhead per share: $n \cdot \tilde{O}(k/n + 1) = \tilde{O}(k + n)$

Our New Scheme

Sharing phase:

$$s \in \mathbb{F}$$

$$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$$

$s_1 = f(x_1)$ … s_i … s_j … s_n

- s_1: κ_{11} y_{11}, …, κ_{1n} y_{1n}
- s_i: κ_{i1} y_{i1}, …, κ_{ij} y_{ij}, …, κ_{in} y_{in}
- s_j: κ_{j1} y_{j1}, …, κ_{ji} y_{ji}, …, κ_{jn} y_{jn}
- s_n: κ_{n1} y_{n1}, …, κ_{nn} y_{nn}

$$y_{ij} = MAC_{\kappa_{ji}}(s_i)$$

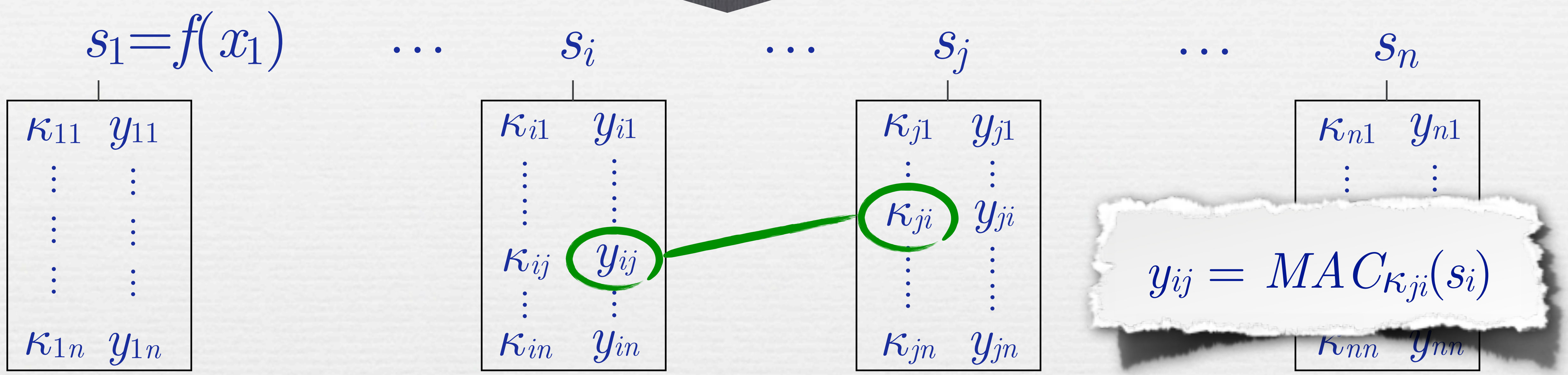
- Use **small** tags and keys $|\kappa_{ij}|, |y_{ij}| = \tilde{O}(k/n+1)$ (instead of $O(k)$)
- Gives: overhead per share: $n \cdot \tilde{O}(k/n+1) = \tilde{O}(k+n)$
- Problem:
  - MAC has weak security
  - incorrect shares may be approved by some honest players
  - Rabin & Ben-Or reconstruction fails

Our New Scheme

Sharing phase:

$s \in \mathbb{F}$

$f(X) = s + a_1 X + \ldots + a_t X^t \in \mathbb{F}[X]$



- Use **small** tags and keys $|\kappa_{ij}|, |y_{ij}| = \tilde{O}(k/n + 1)$ (instead of $O(k)$)
- Gives: overhead per share: $n \cdot \tilde{O}(k/n + 1) = \tilde{O}(k + n)$
- Problem
  - MAC
  - incorrect shares may be approved by some honest players
  - Rabin & Ben-Or reconstruction fails

Need: better reconstruction procedure

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$

How to Reconstruct

- Example: Say that
  - $\{j|\, y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1

How to Reconstruct

- Example: Say that
  - $\{j |\, y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j |\, y_{2j} = MAC_{\kappa_{j2}}(s_2)\} = \{1,\ldots,t+1\}$

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j \mid y_{2j} = MAC_{\kappa_{j2}}(s_2)\} = \{1,\ldots,t+1\}$ -> accept s_2

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j \mid y_{2j} = MAC_{\kappa_{j2}}(s_2)\} = \{1,\ldots,t+1\}$ -> accept s_2
  - $\{j \mid y_{3j} = MAC_{\kappa_{j3}}(s_3)\} = \{2,\ldots,t+1\}$

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j \mid y_{2j} = MAC_{\kappa_{j2}}(s_2)\} = \{1,\ldots,t+1\}$ -> accept s_2
  - $\{j \mid y_{3j} = MAC_{\kappa_{j3}}(s_3)\} = \{2,\ldots,t+1\}$ -> reject s_3
  - ...

How to Reconstruct

- Example: Say that
  - $\{j|\, y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j|\, y_{2j} = MAC_{\kappa_{j2}}(s_2)\} = \{1,\ldots,t{+}1\}$ -> accept s_2
  - $\{j|\, y_{3j} = MAC_{\kappa_{j3}}(s_3)\} = \{2,\ldots,t{+}1\}$ -> reject s_3
  - ...

- s_2 is approved by $\leq t$ **honest** players (as player 3 is dishonest)
  => s_2 stems from dishonest player

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{K_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j \mid y_{2j} = MAC_{K_{j2}}(s_2)\} = \{1,\ldots,t+1\}$ -> accept s_2
  - $\{j \mid y_{3j} = MAC_{K_{j3}}(s_3)\} = \{2,\ldots,t+1\}$ -> reject s_3
  - ...
- s_2 is approved by $\leq t$ **honest** players (as player 3 is dishonest)

  => s_2 stems from dishonest player
- Rabin & Ben-Or reconstruction: accepts s_2

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{K_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j \mid y_{2j} = MAC_{K_{j2}}(s_2)\} = \{1,\ldots,t{+}1\}$ -> accept s_2
  - $\{j \mid y_{3j} = MAC_{K_{j3}}(s_3)\} = \{2,\ldots,t{+}1\}$ -> reject s_3
  - ...
- s_2 is approved by $\leq t$ **honest** players (as player 3 is dishonest) => s_2 stems from dishonest player
- Rabin & Ben-Or reconstruction: accepts s_2
- Our new reconstruction: will rejects s_2

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j$
  - $\{j$
  - .
- s_2 is s dishonest)

  =>
- Rabi
- Our new reconstruction: will rejects s_2

Rabin & Ben-Or reconstruction:

Accept every share s_i that is approved by $t+1$ players.

Our new reconstruction:

Accept every share s_i that is approved by $t+1$ players **with accepted shares**.

How to Reconstruct

- Example: Say that
  - $\{j \mid y_{1j} = MAC_{\kappa_{j1}}(s_1)\} = \{1,\ldots,n\}$ -> accept s_1
  - $\{j$
  - $\{j$
  - ...
- s_2 is ... s dishonest)
  => ...
- Rabi...
- Our n...

Rabin & Ben-Or reconstruction:

Accept every share s_i that is approved by $t+1$ players.

Our new reconstruction:

Accept every share s_i that is approved by $t+1$ players **with accepted shares**.

Plus: Reed-Solomon decoding.

Our New Reconstruction Procedure

(Init) Set $Good := \{1,\ldots,n\}$

(Loop) For every $i \in Good$:

if $\#\{j \in Good \mid y_{ij} = MAC_{K_{ji}}(s_i)\} \leq t$ then

- set $Good := Good \setminus \{i\}$
- redo (Loop)

(Dec) Set $s := \text{Reed-Solomon}(\{s_i\}_{i \in Good})$

Our New Reconstruction Procedure

```
(Init) Set  Good := {1,...,n}

(Loop) For every i ∈ Good:

         if #{j ∈ Good | y_ij = MAC_K_ji(s_i)} ≤ t then
           - set  Good := Good \ {i}
           - redo (Loop)

(Dec)  Set  s := Reed-Solomon({s_i}_{i ∈ Good})
```

(Init) Set $Good := \{1,\ldots,n\}$

(Loop) For every $i \in Good$:

if $\#\{j \in Good \mid y_{ij} = MAC_{K_{ji}}(s_i)\} \leq t$ then

- set $Good := Good \setminus \{i\}$
- redo (Loop)

(Dec) Set $s := \text{Reed-Solomon}(\{s_i\}_{i \in Good})$

Our New Reconstruction Procedure

```
(Init)  Set $Good := \{1,\dots,n\}$
```

(Init) Set $Good := \{1,\ldots,n\}$

(Loop) For every $i \in Good$:

if $\#\{j \in Good \mid y_{ij} = MAC_{K_{ji}}(s_i)\} \leq t$ then

- set $Good := Good \setminus \{i\}$
- redo (Loop)

(Dec) Set $s := \text{Reed-Solomon}(\{s_i\}_{i \in Good})$

Our New Reconstruction Procedure

```
(Init)  Set  Good := {1,...,n}

(Loop)  For every i ∈ Good:

            if #{j ∈ Good | y_ij = MAC_K_ji(s_i)} ≤ t then
              - set Good := Good \ {i}
              - redo (Loop)

(Dec)   Set  s := Reed-Solomon({s_i}_{i ∈ Good})
```

(Init) Set $Good := \{1,\ldots,n\}$

(Loop) For every $i \in Good$:

if $\#\{j \in Good \mid y_{ij} = MAC_{K_{ji}}(s_i)\} \leq t$ then

- set $Good := Good \setminus \{i\}$
- redo (Loop)

(Dec) Set $s := \text{Reed-Solomon}(\{s_i\}_{i \in Good})$

Our New Reconstruction Procedure

(Init) Set $Good := \{1,\ldots,n\}$

(Loop) For every $i \in Good$:

if $\#\{j \in Good \mid y_{ij} = MAC_{K_{ji}}(s_i)\} \leq t$ then

- set $Good := Good \setminus \{i\}$
- redo (Loop)

(Dec) Set $s :=$ Reed-Solomon$(\{s_i\}_{i \in Good})$

Our New Reconstruction Procedure

```
(Init) Set Good := {1,...,n}

(Loop) For every i ∈ Good:

          if #{j ∈ Good | y_ij = MAC_{κ_ji}(s_i)} ≤ t then
             - set Good := Good \ {i}
             - redo (Loop)

(Dec)  Set s := Reed-Solomon({s_i}_{i∈Good})
```

Main Theorem. If MAC is ε-secure then our scheme is δ-robust with

$$\delta \le e\cdot((t+1)\cdot\varepsilon)^{(t+1)/2}$$

(where $e = \exp(1)$).

Our New Reconstruction Procedure

```
(Init)  Set Good := {1,...,n}

(Loop)  For every i ∈ Good :

           if #{j ∈ Good | y_ij = MAC_κji(s_i)} ≤ t then
             - set Good := Good \ {i}
             - redo (Loop)

(Dec)   Set s := Reed-Solomon({s_i}_{i∈Good})
```

Main Theorem. If MAC is ε-secure then our scheme is δ-robust with

$$\delta \leq e \cdot ((t+1) \cdot \varepsilon)^{(t+1)/2}$$ (where $e = \exp(1)$).

Corollary. Using MAC with $|\kappa_{ij}|, |y_{ij}| = O(k/n + \log n)$ gives

$$\delta \leq 2^{-\Omega(k)} .$$

What Makes the Proof Tricky

What Makes the Proof Tricky

1. **Optimal strategy for dishonest players is unclear**

- In Rabin & Ben-Or: an incorrect share for every dishonest player
- Here: some dishonest players may hand in correct shares

What Makes the Proof Tricky

1. **Optimal strategy for dishonest players is unclear**

- In Rabin & Ben-Or: an incorrect share for every dishonest player
- Here: some dishonest players may hand in correct shares
- Such a passive dishonest player:
  - stays in Good
  - can support (i.e. vote for) bad shares
- **The more** such passive dishonest players:
  - **the easier it gets** for bad shares to survive
  - **the more** bad shares have to survive to fool RS decoding (# bad shares $\geq$ # correct shares of dishonest players)
- Optimal trade-off: unclear

What Makes the Proof Tricky

2. Circular dependencies

- Whether $\hat{s}_i$ gets accepted **depends** on whether $\hat{s}_j$ gets accepted …

What Makes the Proof Tricky

2. Circular dependencies

- Whether $\hat{s}_i$ gets accepted depends on whether $\hat{s}_j$ gets accepted …
- … and vice versa
- Cannot analyze individual bad shares
- If we try, we run into a circularity

The Proof

Notation:

- $\mathcal{A}$/$\mathcal{P}$/$\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

The Proof

Notation:

- $\mathcal{A}$/$\mathcal{P}$/$\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell = |\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell = |\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \binom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \ \exists \mathcal{H}' \in \binom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}' \colon y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)\ ]$$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P}$,$\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell=|\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$P[\ldots] \leq \varepsilon$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \binom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \exists \mathcal{H}' \in \binom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}'\ y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)]$$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell = |\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$P[\ldots] \leq \varepsilon$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \binom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \ \exists \mathcal{H}' \in \binom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}' : y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)]$$

$$\leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} P[\forall i \in \mathcal{A}'\ \exists \ldots \forall \ldots\,]$$

The Proof

Notation:

- $\mathcal{A}$/$\mathcal{P}$/$\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell=|\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$P[\ldots] \leq \varepsilon$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \binom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \exists \mathcal{H}' \in \binom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}'\ y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)]$$

$$\leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} P[\forall i \in \mathcal{A}'\ \exists \ldots \forall \ldots] \leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} \prod_{i \in \mathcal{A}'} P[\exists \ldots \forall \ldots] \leq \ldots$$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t{+}1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell=|\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$P[\dots] \leq \varepsilon$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \tbinom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \exists \mathcal{H}' \in \tbinom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}'\ y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)]$$

$$\leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} P[\forall i \in \mathcal{A}'\ \exists \dots \forall \dots] \leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} \prod_{i \in \mathcal{A}'} P[\exists \dots \forall \dots] \leq \dots$$

$$\leq \sum_{\ell} \binom{a}{\ell} \cdot \left(\binom{t+1}{a-\ell+1} \cdot \varepsilon^{a-\ell+1} \right)^{\ell}$$

The Proof

Notation:

- $\mathcal{A}/\mathcal{P}/\mathcal{H}$ = active/passive cheaters, and honest players where (wlog) $|\mathcal{A}|+|\mathcal{P}| = t$ and $|\mathcal{H}| = t+1$
- $\boldsymbol{\mathcal{S}}$ = players that survive checking phase ($\mathcal{P},\mathcal{H} \subseteq \boldsymbol{\mathcal{S}}$)

Observations:

- Error probability given by $\delta = P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,]$
- $\delta = 0$ if $|\mathcal{A}| \leq |\mathcal{P}|$. Thus: may assume $a := |\mathcal{A}| > t/2$

Actual proof:

$$P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| > |\mathcal{P}|\,] = \sum_{\ell=|\mathcal{P}|+1}^{a} P[\,|\mathcal{A} \cap \boldsymbol{\mathcal{S}}| = \ell\,]$$

$P[\ldots] \leq \varepsilon$

$$\leq \sum_{\ell} P[\exists \mathcal{A}' \in \binom{\mathcal{A}}{\ell}\ \forall i \in \mathcal{A}'\ \exists \mathcal{H}' \in \binom{\mathcal{H}}{a-\ell+1}\ \forall j \in \mathcal{H}'\ y_{ij} = MAC_{\boldsymbol{\kappa}_{ji}}(\hat{s}_i)]$$

$$\leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} P[\forall i \in \mathcal{A}'\ \exists \ldots \forall \ldots] \leq \sum_{\ell} \sum_{\mathcal{A}' \in \binom{\mathcal{A}}{\ell}} \prod_{i \in \mathcal{A}'} P[\exists \ldots \forall \ldots] \leq \ldots$$

$$\leq \sum_{\ell} \binom{a}{\ell} \cdot \left(\binom{t+1}{a-\ell+1} \cdot \varepsilon^{a-\ell+1} \right)^{\ell} \leq \ldots \leq e \cdot ((t+1) \cdot \varepsilon)^{(t+1)/2}$$

$\blacksquare$

Summary

- First robust secret sharing scheme for $n = 2t+1$, with
  - small overhead $O(k+n{\cdot}\log n)$ in share size
  - efficient sharing and reconstruction procedures
- Scheme is simple and natural adaptation of Rabin & Ben-Or
- Proof is non-standard and non-trivial

Summary

- First robust secret sharing scheme for $n = 2t{+}1$, with
  - small overhead $O(k{+}n{\cdot}\log n)$ in share size
  - efficient sharing and reconstruction procedures
- Scheme is simple and natural adaptation of Rabin & Ben-Or
- Proof is non-standard and non-trivial
- Open problem:
  - Scheme with overhead $O(k)$ (= proven lower bound)
- Note:
  - All known schemes have a $\Omega(n)$ gap (for different reasons)
  - Not known if this is inherent or not.

Summary

- First robust secret sharing scheme for $n = 2t+1$, with
  - small overhead $O(k+n{\cdot}\log n)$ in share size
  - efficient sharing and reconstruction procedures
- Scheme is simple and natural adaptation of Rabin & Ben-Or
- Proof is non-standard and non-trivial
- Open problem:
  - Scheme with overhead $O(k)$ (= proven lower bound)
- Note:
  - All kr ... t reasons)
  - Not k

THANK YOU