Short Leakage Resilient and Non-malleable Secret Sharing Schemes

Sruthi Sekar

Berkeley
UNIVERSITY OF CALIFORNIA

Nishanth Chandran

Microsoft Research

Bhavana Kanukurthi

IISc

Sai Lakshmi Bhavana Obbattu

Microsoft Research

Secret Sharing Schemes

Shamir and Blakely (1979)

Secret Sharing Schemes

Shamir and Blakely (1979)

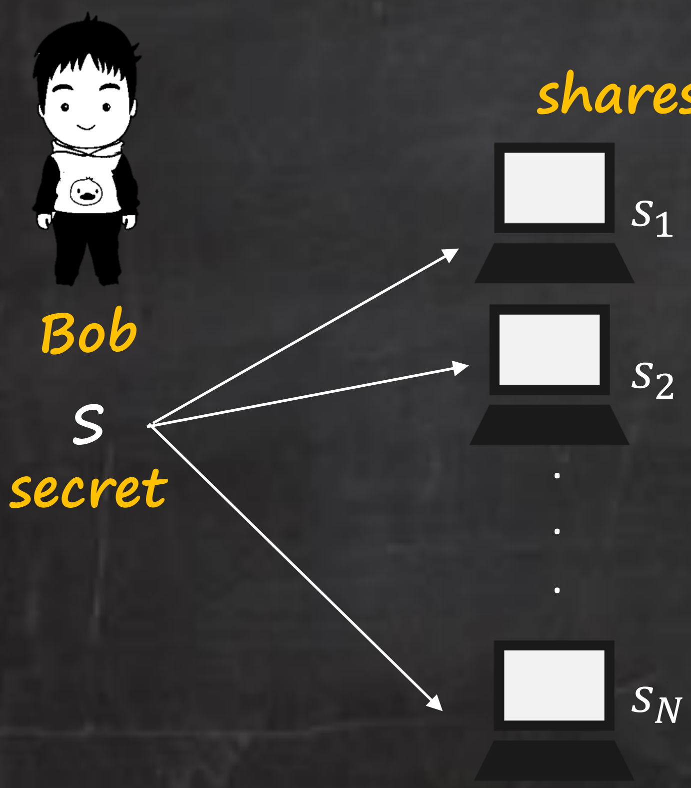
Bob

s

secret

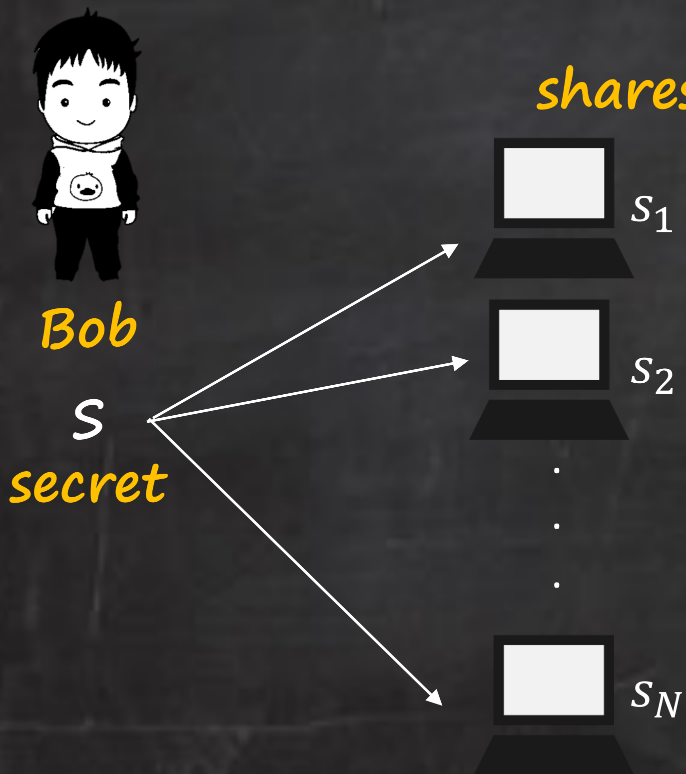
Secret Sharing Schemes

Shamir and Blakely (1979)



Secret Sharing Schemes

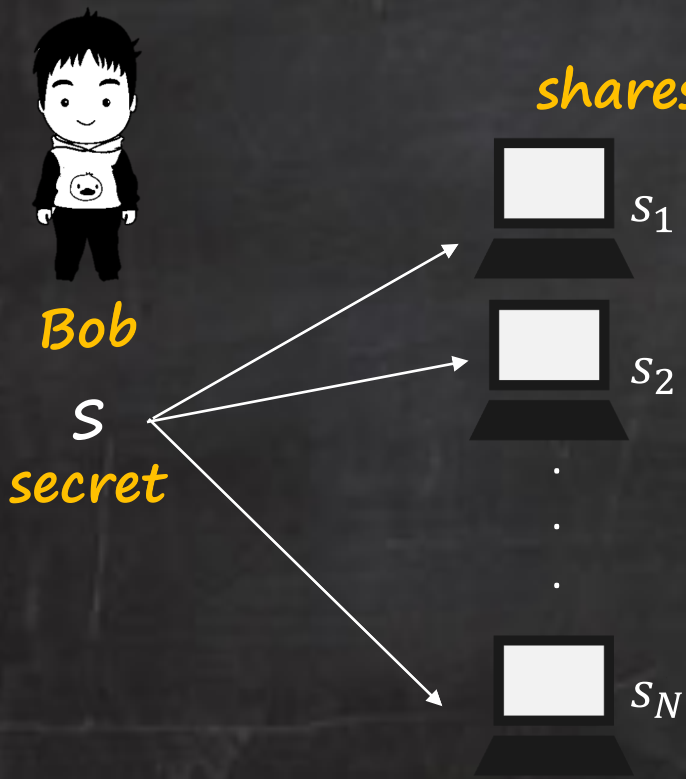
Shamir and Blakely (1979)



<u>Correctness</u>:
$\geq t$ shares give s.

Secret Sharing Schemes

Shamir and Blakely (1979)

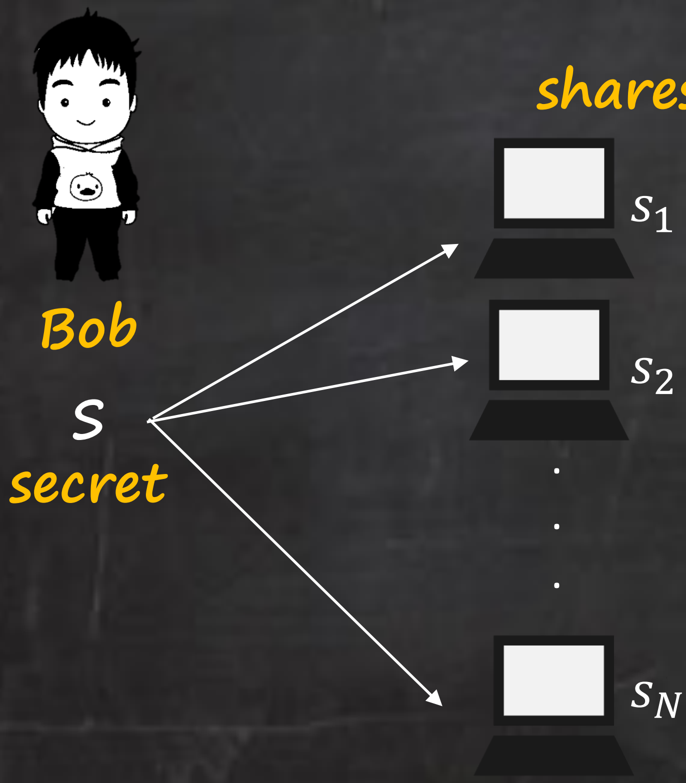


<u>Correctness</u>:
$\geq t$ shares give s.

<u>Privacy</u>:
An adversary with $< t$ shares gets no information about s.

Secret Sharing Schemes

Shamir and Blakely (1979)



__Correctness__:

$\geq t$ shares give s.

__Privacy__:

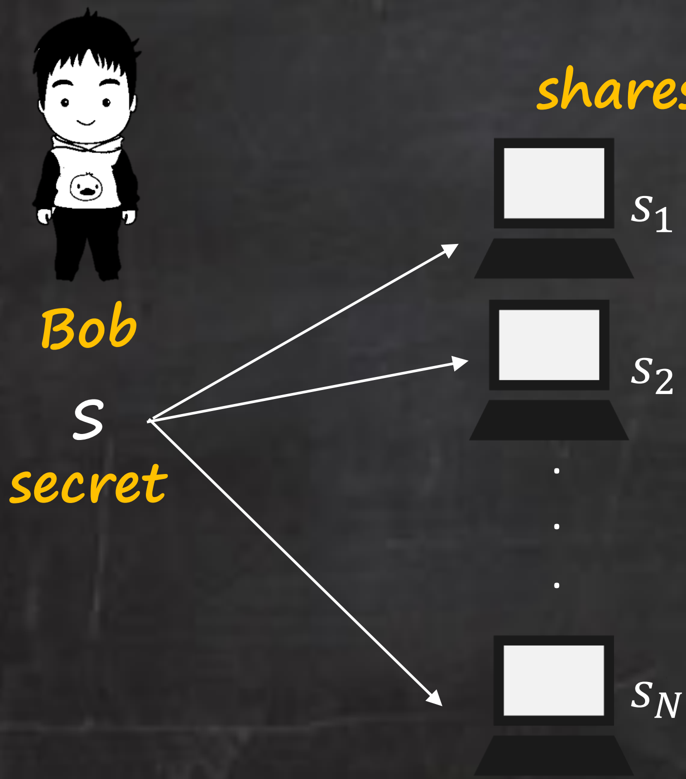
An adversary with $< t$ shares gets no information about s.

$$\forall\ s, s'$$

$$(< t \text{ shares of } s) \approx (< t \text{ shares of } s')$$

Secret Sharing Schemes

Shamir and Blakely (1979)



Correctness:

$\geq t$ shares give s.

Privacy:

An adversary with $< t$ shares gets no information about s.

$\forall$ s,s′

$(< t$ shares of s$) \approx (< t$ shares of s′$)$

Statistical distance

Secret Sharing Schemes

Shamir and Blakely (1979)

B

secr

s.

$\forall$ s,s′

(< t shares of s) ≈ (< t shares of s′)

Secret Sharing Schemes

Shamir and Blakely (1979)

LEAKAGE ATTACKS [Kocher(1996)]

secr

s.

$\forall$ s,s'

(< t shares of s) ≈ (< t shares of s')

Secret Sharing Schemes

Shamir and Blakely (1979)

B

secr

LEAKAGE ATTACKS [Kocher(1996)]

What if in addition to $t-1$ shares, adversary gets (arbitrary) bounded # of bits from other shares too?

s.

$\forall$ s,s′

(< t shares of s) $\approx$ (< t shares of s′)

Secret Sharing Schemes

Shamir and Blakely (1979)

<u>LEAKAGE ATTACKS [Kocher(1996)]</u>

What if in addition to $t-1$ shares, adversary gets (arbitrary) bounded # of bits from other shares too?

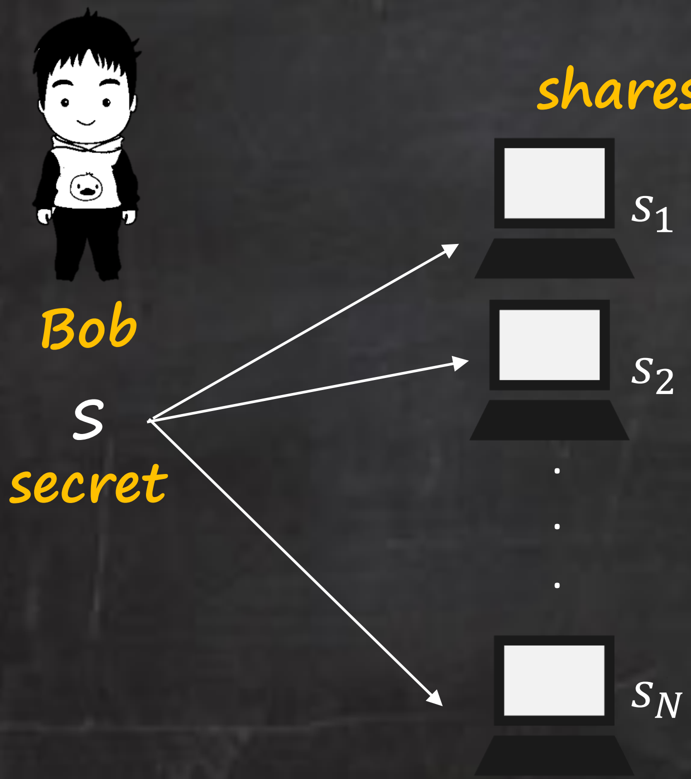
[Guruswami, Wooters (2016)]: Shamir SS breaks, given 1-bit leakage on remaining shares.

$\forall$ s,s′

$(< t$ shares of s$) \approx (< t$ shares of s′$)$

Leakage Resilient Secret Sharing

Dziembowski and Pietrzak (2007)

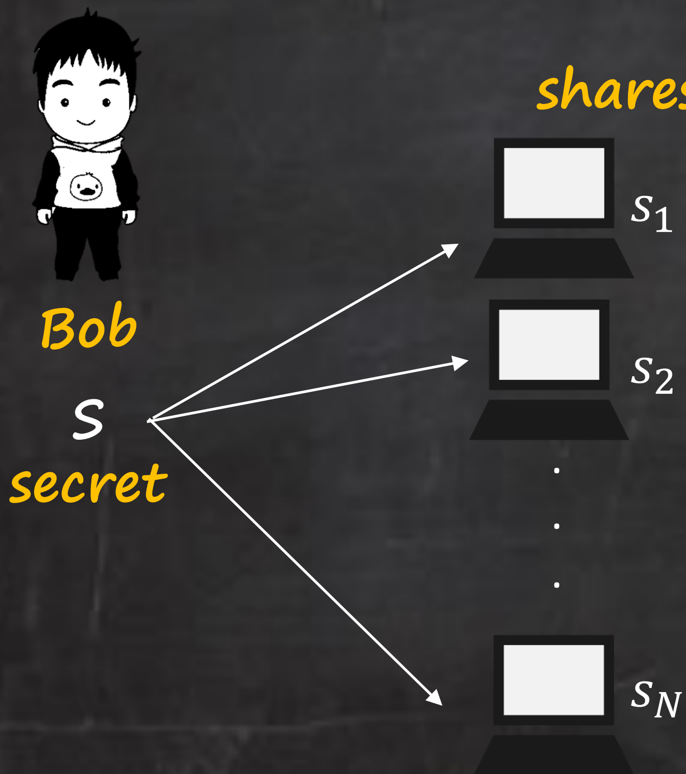


Correctness:

$\geq t$ shares give s.

Leakage Resilient Secret Sharing

Dziembowski and Pietrzak (2007)



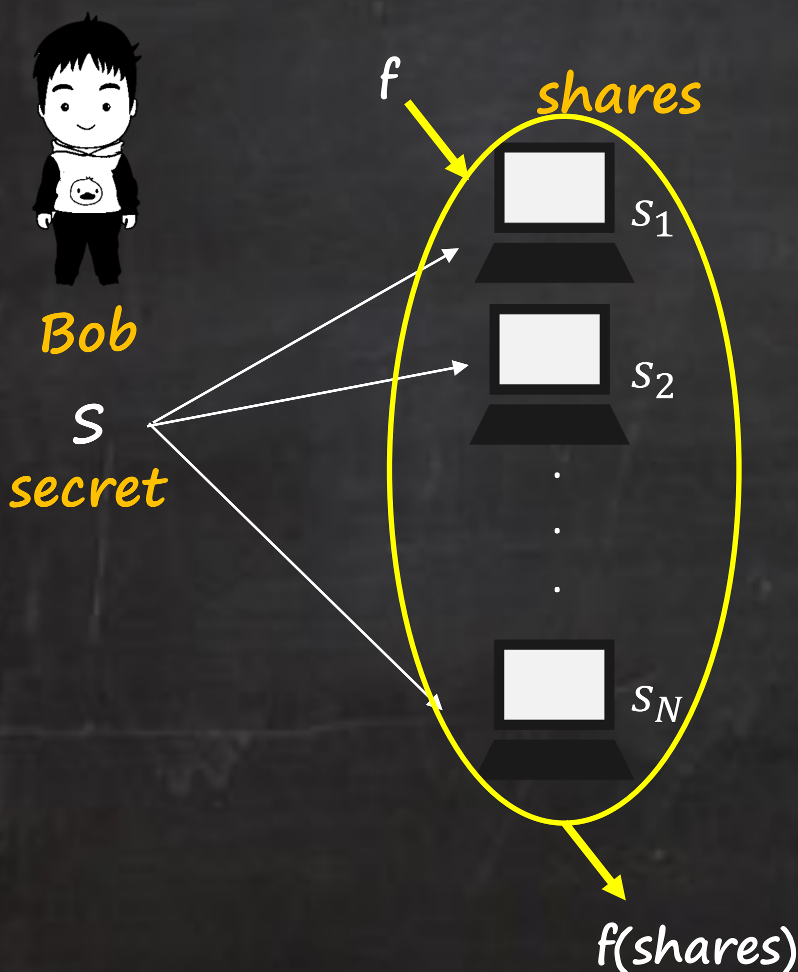
<u>Correctness</u>:

$\geq t$ shares give s.

<u>Leakage Resilience</u>:

Leakage Resilient Secret Sharing

Dziembowski and Pietrzak (2007)



Correctness:

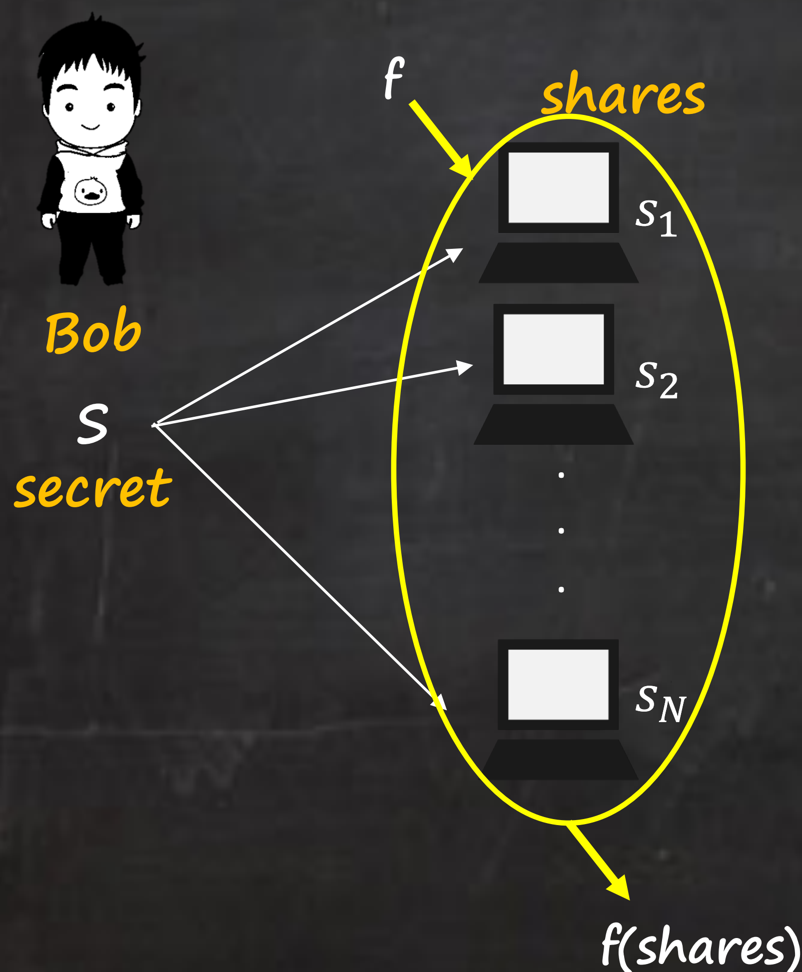
$\geq t$ shares give s.

Leakage Resilience:

$\mathcal{F}$ be some function family.
For function f $\in \mathcal{F}$, f(shares) gives no information about s.

Leakage Resilient Secret Sharing

Dziembowski and Pietrzak (2007)



Correctness:

$\geq t$ shares give s.

Leakage Resilience:

$\mathcal{F}$ be some function family.
For function $f \in \mathcal{F}$, f(shares) gives no information about s.

$$\forall\ s, s'$$

$$f(\text{shares of } s) \approx f(\text{shares of } s')$$

Leakage Resilient Secret Sharing

Parameters

Leakage Resilient Secret Sharing

Parameters

- Share Size: size of the largest share amongst the N shares. Best share size one can hope for an LRSS: $message\ length + \mu$, where μ is the #leakage-bits per share.

Leakage Resilient Secret Sharing

Parameters

- Share Size: size of the largest share amongst the N shares.
  Best share size one can hope for an LRSS: $message\ length + \mu$, where μ is the #leakage-bits per share.

- Leakage Family:

Leakage Resilient Secret Sharing

Parameters

- Share Size: size of the largest share amongst the N shares.
  Best share size one can hope for an LRSS: $message\ length + \mu$, where μ is the #leakage-bits per share.

- Leakage Family:

$\mathcal{F}$: Local Leakage Family

Leakage Resilient Secret Sharing

Parameters

- Share Size: size of the largest share amongst the N shares. Best share size one can hope for an LRSS: *message length* $+ \mu$, where μ is the #leakage-bits per share.
- Leakage Family:

$\mathcal{F}$: Local Leakage Family

Adversary

$\mathsf{f} = (f_1, f_2, \ldots, f_N)$

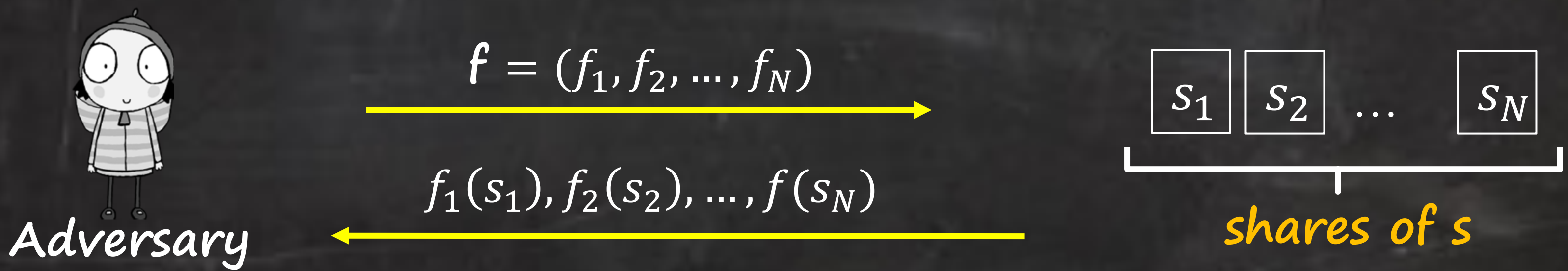
Leakage Resilient Secret Sharing

Parameters

- Share Size: size of the largest share amongst the N shares.
  Best share size one can hope for an LRSS: $message\ length + \mu$,
  where μ is the #leakage-bits per share.

- Leakage Family:

$\mathcal{F}$: Local Leakage Family



Leakage Resilient Secret Sharing

Parameters

- <u>Share Size</u>: size of the largest share amongst the N shares.
  Best share size one can hope for an LRSS: $message\ length + \mu$,
  where μ is the #leakage-bits per share.

- <u>Leakage Family</u>:

$\mathcal{F}$: **Local Leakage Family**

$\mathsf{f} = (f_1, f_2, \dots, f_N)$ →

← $f_1(s_1), f_2(s_2), \dots, f(s_N)$

$t - 1$ of these are full shares,
rest arbitrary functions outputting μ bits each.

s_1	s_2	…	s_N

shares of s

Leakage Resilient Secret Sharing

Prior Works

Leakage Resilient Secret Sharing

Prior Works

- Long line of research: [DDV10, LL12, GK18, BDIR18,GK18, BS19,SV19, ADN+19,KMS19,FV19,BFV19, LCG+19,CGG+20, BFO+20, CKOS21,MPSW21,MNP+21]

Leakage Resilient Secret Sharing

Prior Works

- Long line of research: [DDV10, LL12, GK18, BDIR18,GK18, BS19,SV19, ADN+19,KMS19,FV19,BFV19, LCG+19,CGG+20, BFO+20, CKOS21,MPSW21,MNP+21]

Most of these works focus on stronger leakage models (adaptive, joint)

However, the share size of these schemes is $\omega(message\ length)$!

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

- If $t \geq (N - o(log N))$, Shamir allows leaking $1/4^{th}$ bits of each share.

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

- If $t \geq (N - o(log N))$, Shamir allows leaking $1/4^{th}$ bits of each share.

- If $t \leq \alpha N$, Shamir allows leaking only constant bits of each share.

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

- If $t \geq (N - o(\log N))$, Shamir allows leaking $1/4^{th}$ bits of each share.

- If $t \leq \alpha N$, Shamir allows leaking only constant bits of each share.

This is the best one can hope from Shamir SS—[NS20]

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

- If $t \geq (N - o(\log N))$, Shamir allows leaking $1/4^{th}$ bits of each share.

- If $t \leq \alpha N$, Shamir allows leaking only constant bits of each share.

2. Generic Compiler [ADN+19, SV19]

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR18]
For a large characteristic field, large N, and only constant number of full share corruptions:

- If $t \geq (N - o(log N))$, Shamir allows leaking $1/4^{th}$ bits of each share.
- If $t \leq \alpha N$, Shamir allows leaking only constant bits of each share.

2. Generic Compiler [ADN+19, SV19]

- Best known for arbitrary N and t [SV19]:
  Share size is $(3 . message\ length + \mu)$, with μ-bits of leakage per share ($\mu \leq (1 - o(1))\ . message\ length$).

Leakage Resilient Secret Sharing

Prior Works: Local Leakage Model

1. Leakage resilience of Shamir SS [BDIR...]
For a large characteristic f... umber of
full share co...

- If $t \geq$ (N... e.

- If ...

Can we get an LRSS scheme
with optimal share size and leakage?

2. Ge...

-Best known ...
Share size is $(3\,.$ ... bits of leakage per share
$(\mu \leq (1 - o(1))\,.\,message\ length)$.

Leakage Resilient Secret Sharing

Our Results

Leakage Resilient Secret Sharing

Our Results

- We build the first information-theoretic LRSS scheme for the **threshold access structures** against the **local leakage model** (allowing μ bits of leakage per share), with a share size of $message\ length + \mu$!

Leakage Resilient Secret Sharing

Our Results

- We build the first information-theoretic LRSS scheme for the **threshold access structures** against the **local leakage model** (allowing μ bits of leakage per share), with a share size of $message\ length + \mu$!
- Our compiler works for general access structures too.

Leakage Resilient Secret Sharing

Our Results

- We build the first information-theoretic LRSS scheme for the **threshold access structures** against the **local leakage model** (allowing μ bits of leakage per share), with a share size of $message\ length + \mu$!

- Our compiler works for general access structures too.

- Application:
  -We can build a non-malleable secret sharing scheme against **independent tampering** with an improved **share size** of ($4.\ message\ length$),

Leakage Resilient Secret Sharing

Our Results

- We build the first information-theoretic LRSS scheme for the **threshold access structures** against the **local leakage model** (allowing μ bits of leakage per share), with a share size of $message\ length + \mu$!

- Our compiler works for general access structures too.

- Application:
  -We can build a non-malleable secret sharing scheme against **independent tampering** with an improved **share size** of (4. $message\ length$),

  -Introduce and build non-malleable randomness sharing against independent tampering with a share size of (2. $message\ length$).

Leakage Resilient Secret Sharing

Our Results

- We build the first information-theoretic LRSS scheme for the **threshold access structures** against the **local leakage model** (allowing μ bits of leakage per share), with a share size of $message\ length + \mu$!

- Our compiler works for general access structures too.

- Application:
  -We can build a non-malleable secret sharing scheme against **independent tampering** with an improved **share size** of ($4. message\ length$),

  -Introduce and build non-malleable randomness sharing against independent tampering with a share size of ($2. message\ length$).

OUR CONSTRUCTION

Building Blocks

Linear Extractors

Building Blocks

Linear Extractors

RANDOMNESS EXTRACTORS

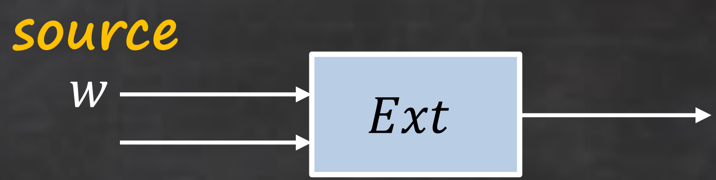
[Nisan and Zuckerman, 1996]

Building Blocks

Linear Extractors

$w \leftarrow W$

$H_\infty(W|Z) \geq k$

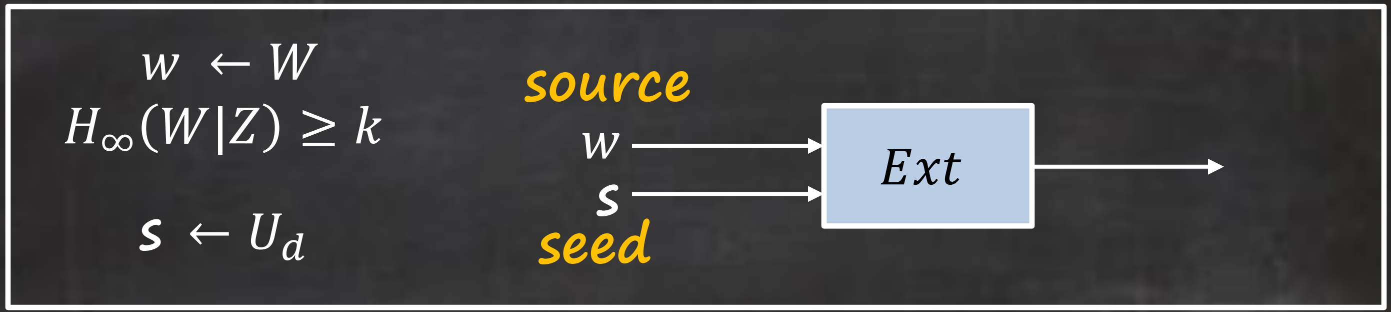


RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

Building Blocks

Linear Extractors



RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

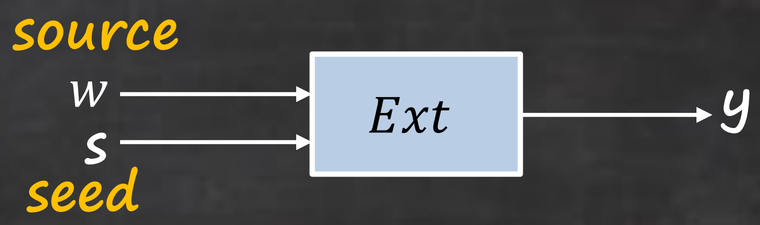
Building Blocks

Linear Extractors

$w \leftarrow W$

$H_\infty(W|Z) \geq k$

$s \leftarrow U_d$



RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

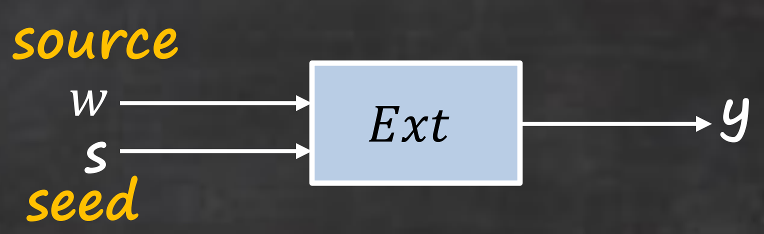
Building Blocks

Linear Extractors

$w \leftarrow W$

$H_\infty(W|Z) \geq k$

$s \leftarrow U_d$



RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

- **Uniformity**: $Ext(W; U_d), Z, U_d \approx U_l, Z, U_d$

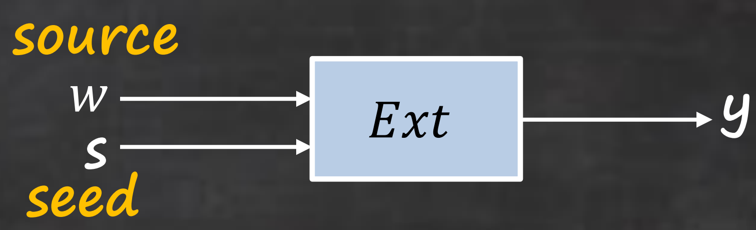
Building Blocks

Linear Extractors

$w \leftarrow W$

$H_\infty(W|Z) \geq k$

$s \leftarrow U_d$



RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

- **Uniformity**: $Ext(W; U_d), Z, U_d \approx U_l, Z, U_d$
- **Linearity**: For each $s \in \{0,1\}^d$, $Ext(\,.,\, s)$ is a linear function.

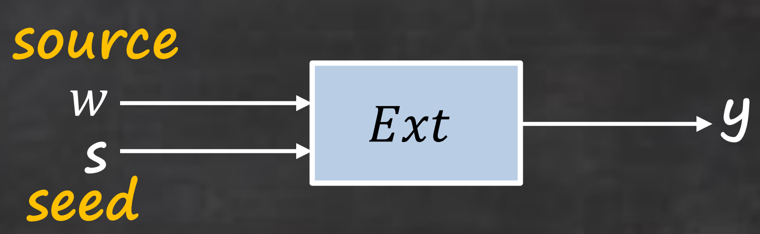
Building Blocks

Linear Extractors

$w \leftarrow W$

$H_\infty(W|Z) \geq k$

$s \leftarrow U_d$



RANDOMNESS EXTRACTORS

[Nisan and Zuckerman, 1996]

- **Uniformity**: $Ext(W; U_d), Z, U_d \approx U_l, Z, U_d$
- **Linearity**: For each $s \in \{0,1\}^d$, $Ext(\,.\,, s)$ is a linear function.

We use invertibility of such linear extractors!

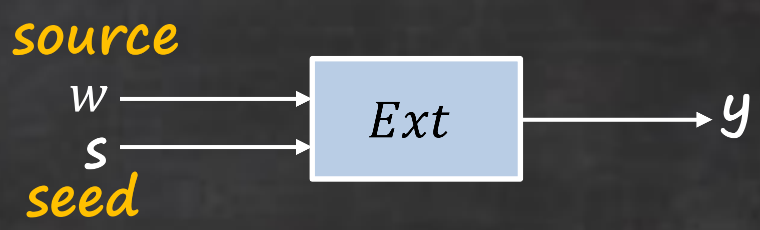
Building Blocks

Linear Extractors: Invertibility

$w \leftarrow U_\eta$

$H_\infty(U_\eta | Z) \geq k$

$s \leftarrow U_d$



LINEAR RANDOMNESS EXTRACTORS

For the above Ext, there exists efficient $InvExt$ such that:

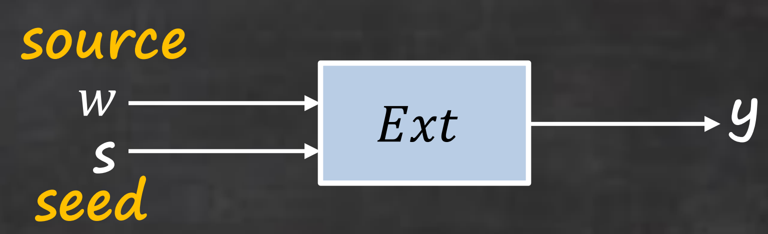
Building Blocks

Linear Extractors: Invertibility

$w \leftarrow U_\eta$

$H_\infty(U_\eta | Z) \geq k$

$s \leftarrow U_d$



LINEAR RANDOMNESS EXTRACTORS

For the above Ext, there exists efficient $InvExt$ such that:

1. $InvExt(Ext(U_\eta; U_d), U_d), U_d, Ext(U_\eta, U_d) \equiv U_\eta, U_d, Ext(U_\eta, U_d)$

Can invert and get a "correct" source string w, given a seed s and an extractor output y.

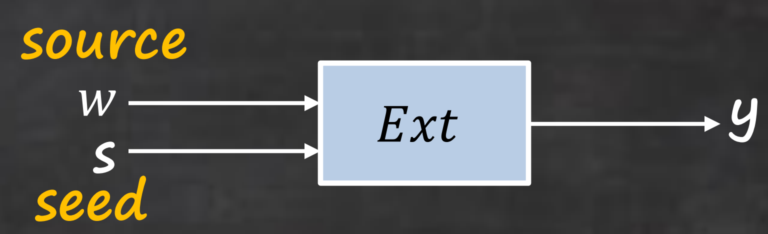
Building Blocks

Linear Extractors: Invertibility

$w \leftarrow U_\eta$

$H_\infty(U_\eta | Z) \geq k$

$s \leftarrow U_d$



LINEAR RANDOMNESS EXTRACTORS

For the above Ext, there exists efficient $InvExt$ such that:

1. $InvExt(Ext(U_\eta; U_d), U_d), U_d, Ext(U_\eta, U_d) \equiv U_\eta, U_d, Ext(U_\eta, U_d)$

2. For each (s,y) $\in \{0,1\}^d \times \{0,1\}^l$:

   - If there exists w s.t. $Ext(w;s) = y$,

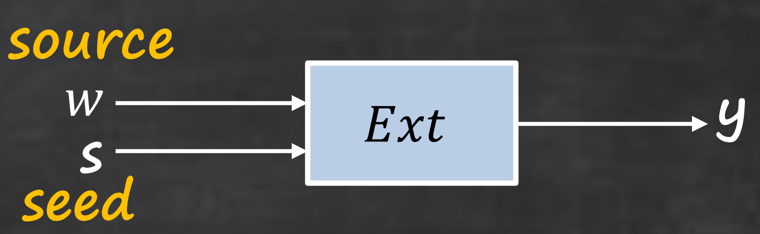
Building Blocks

Linear Extractors: Invertibility

$w \leftarrow U_\eta$

$H_\infty(U_\eta | Z) \geq k$

$s \leftarrow U_d$



LINEAR RANDOMNESS EXTRACTORS

For the above Ext, there exists efficient $InvExt$ such that:

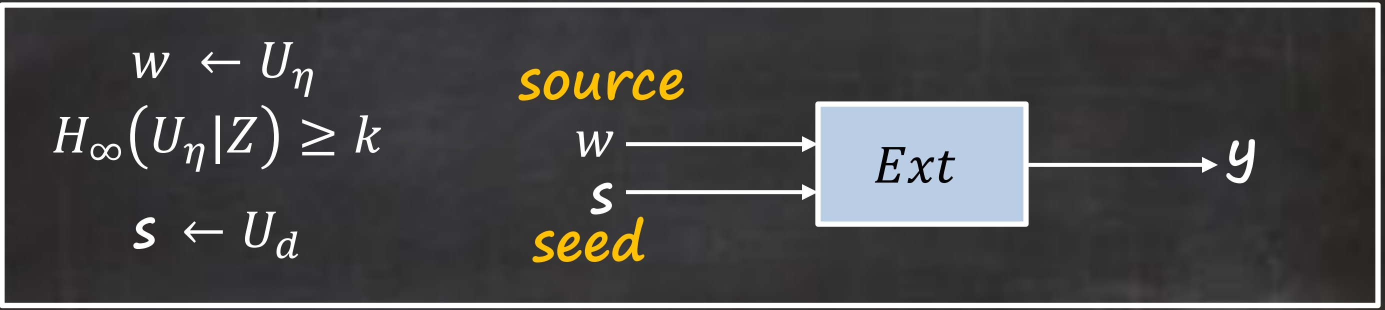
1. $InvExt(Ext(U_\eta; U_d), U_d), U_d, Ext(U_\eta, U_d) \equiv U_\eta, U_d, Ext(U_\eta, U_d)$

2. **For each (s,y)** $\in \{0,1\}^d \times \{0,1\}^l$:

   - **If there exists** w **s.t.** $Ext(w; s) = y$, $Ext(InvExt(y, s); s) = y$ **w.p. 1.**

Building Blocks

Linear Extractors: Invertibility



LINEAR RANDOMNESS EXTRACTORS

For the above Ext, there exists efficient $InvExt$ such that:

1. $InvExt(Ext(U_\eta; U_d), U_d), U_d, Ext(U_\eta, U_d) \equiv U_\eta, U_d, Ext(U_\eta, U_d)$

2. **For each (s,y)** $\in \{0,1\}^d \times \{0,1\}^l$:

   - **If there exists** w **s.t.** $Ext(w;s) = y$, $Ext(InvExt(y,s)\,;\,s) = y$ **w.p. 1.**
   - **Else** $InvExt(y,s) = \perp$ **w.p. 1.**

Our Construction

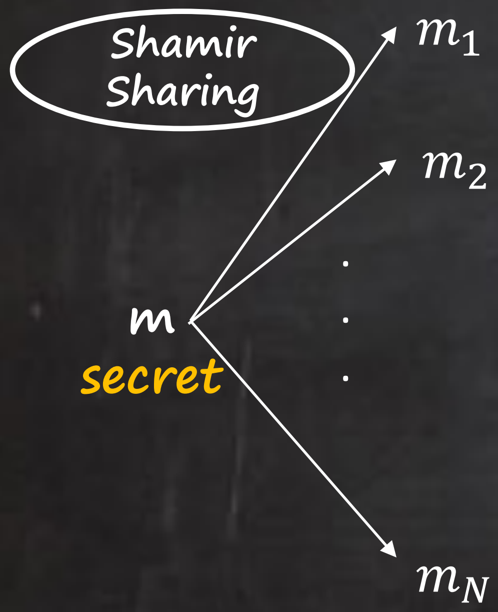
Optimal Threshold LRSS

m

secret

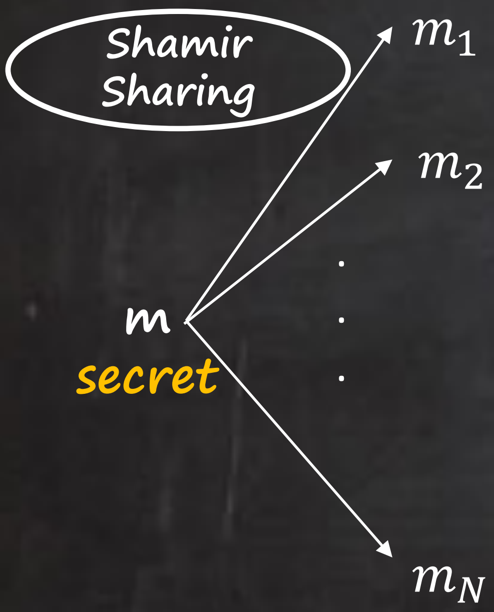
Our Construction

Optimal Threshold LRSS



Our Construction

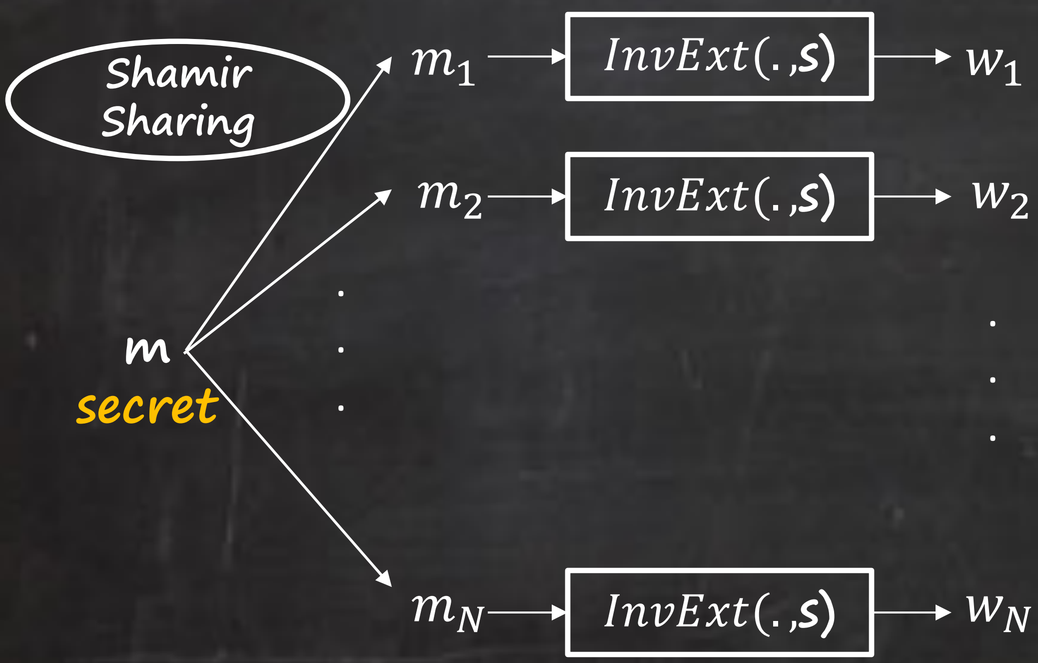
Optimal Threshold LRSS



Sample $s \leftarrow U_d$

Our Construction

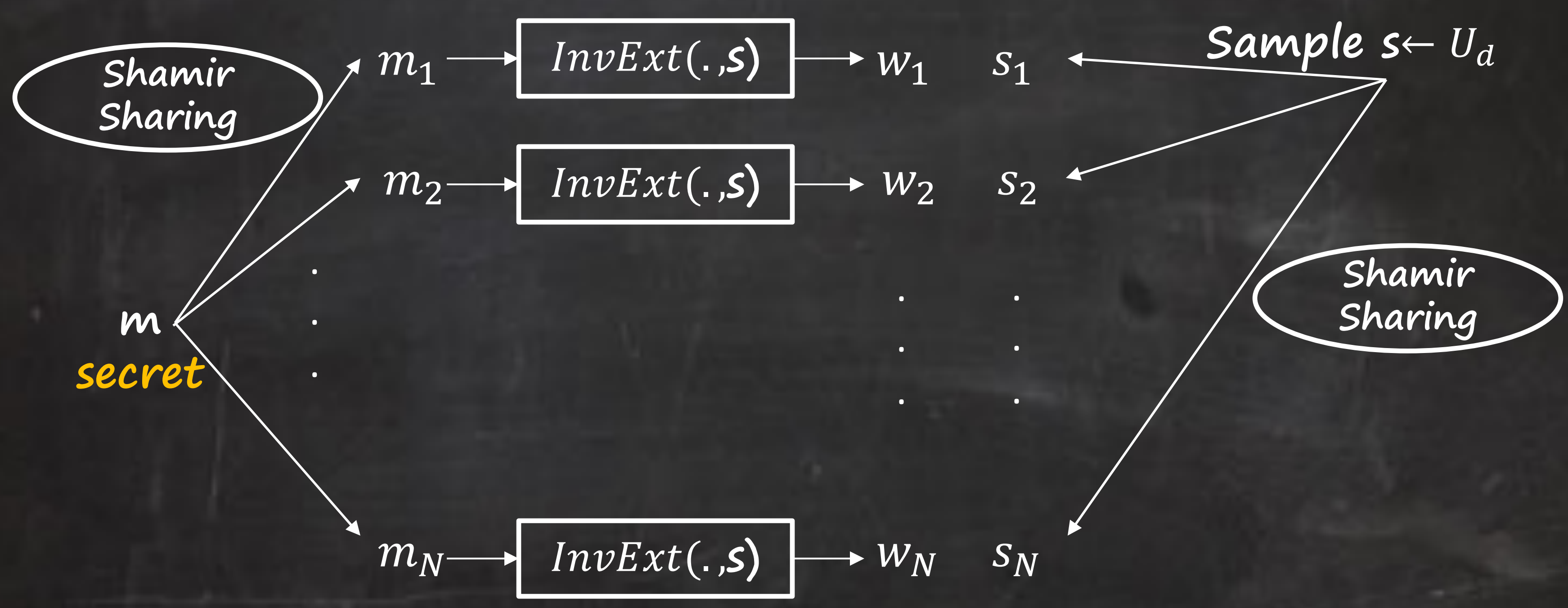
Optimal Threshold LRSS



Sample $s \leftarrow U_d$

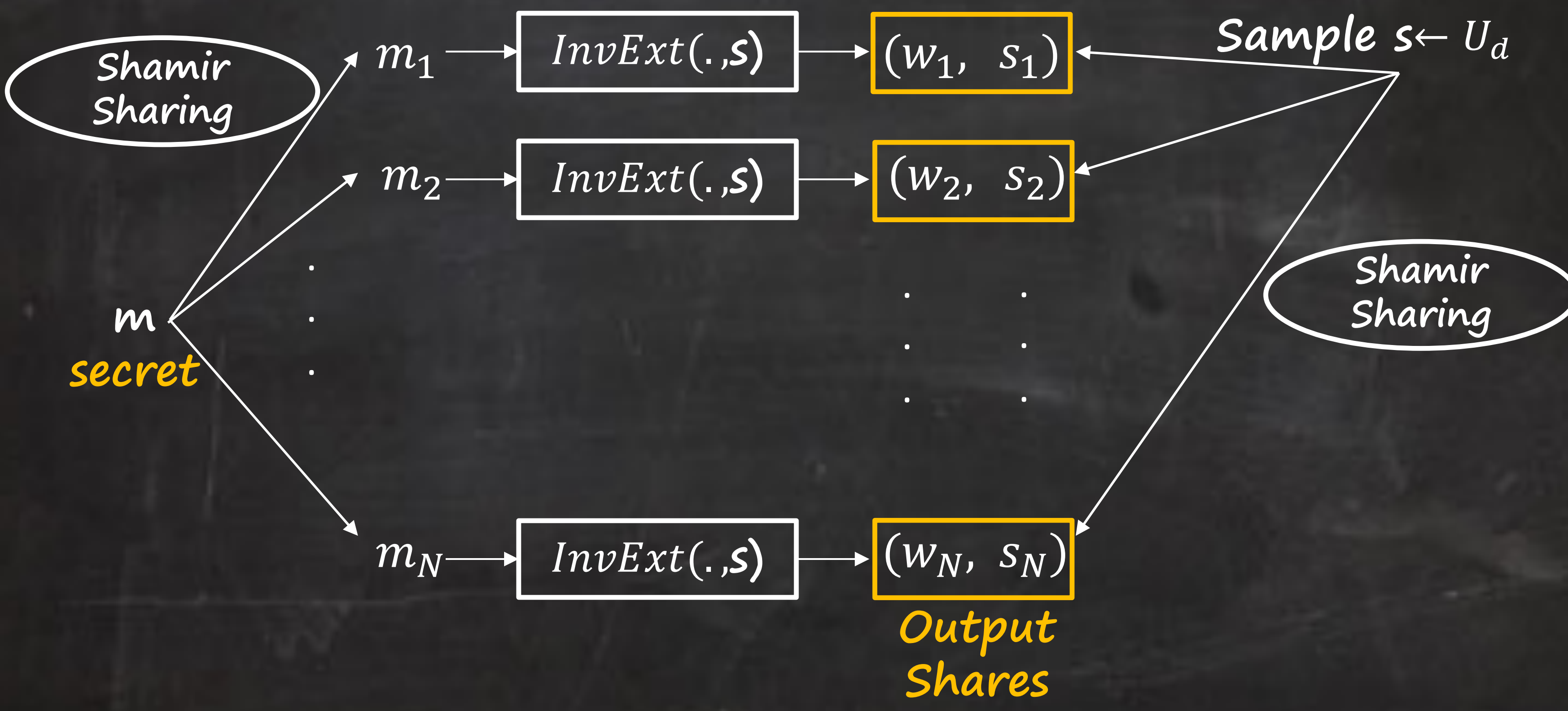
Our Construction

Optimal Threshold LRSS



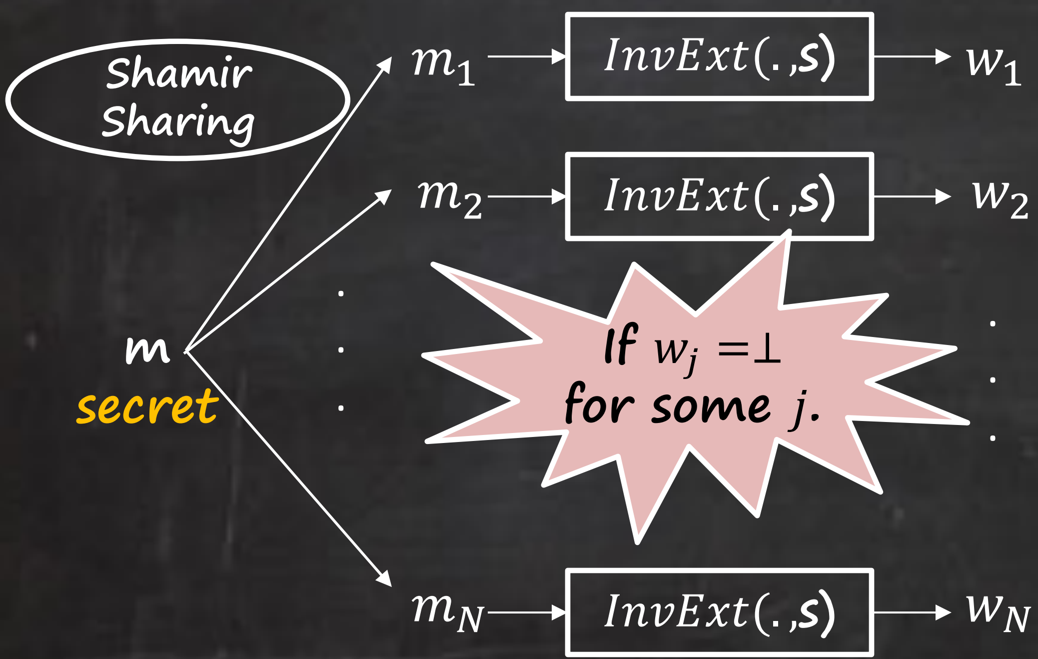
Our Construction

Optimal Threshold LRSS



Our Construction

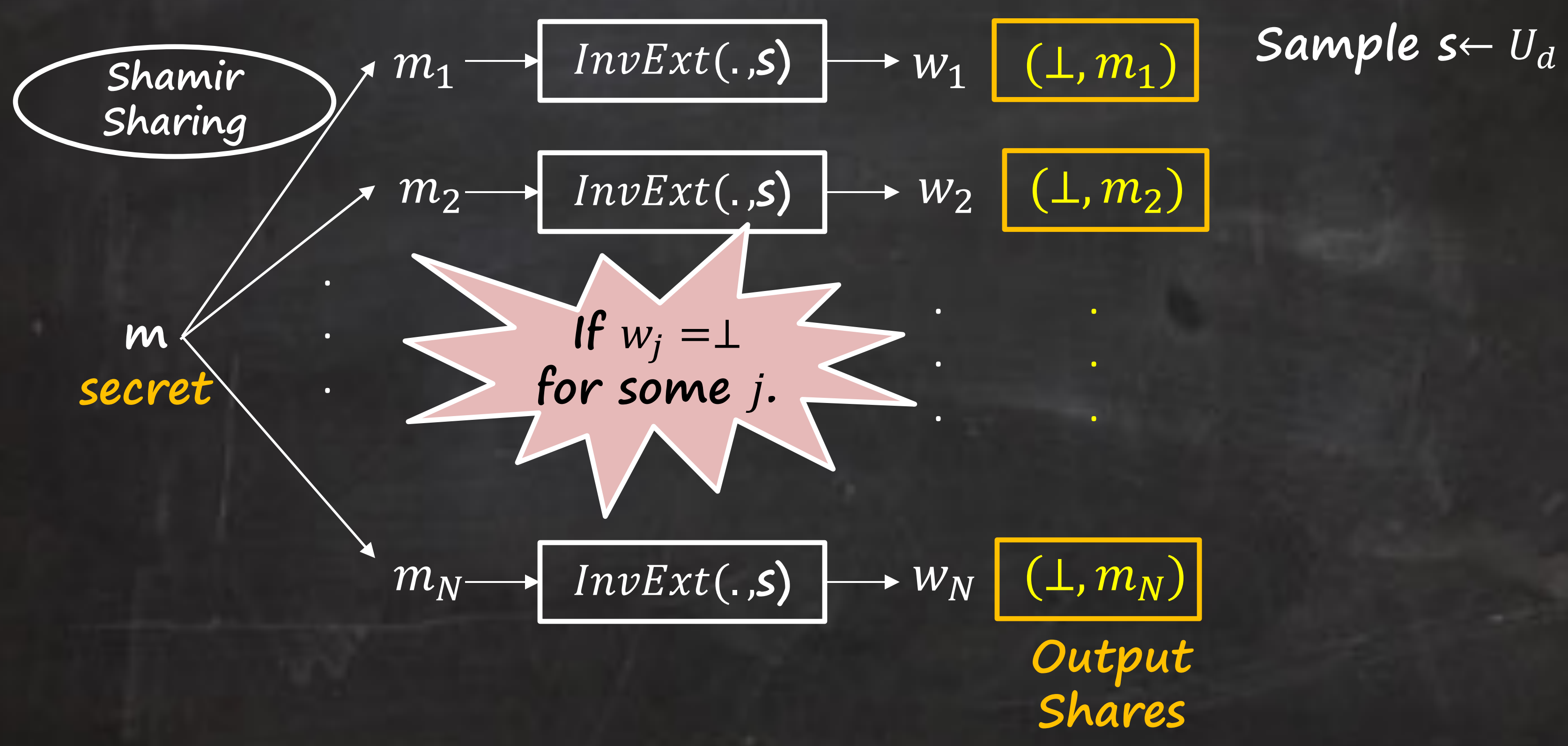
Optimal Threshold LRSS



Sample $s \leftarrow U_d$

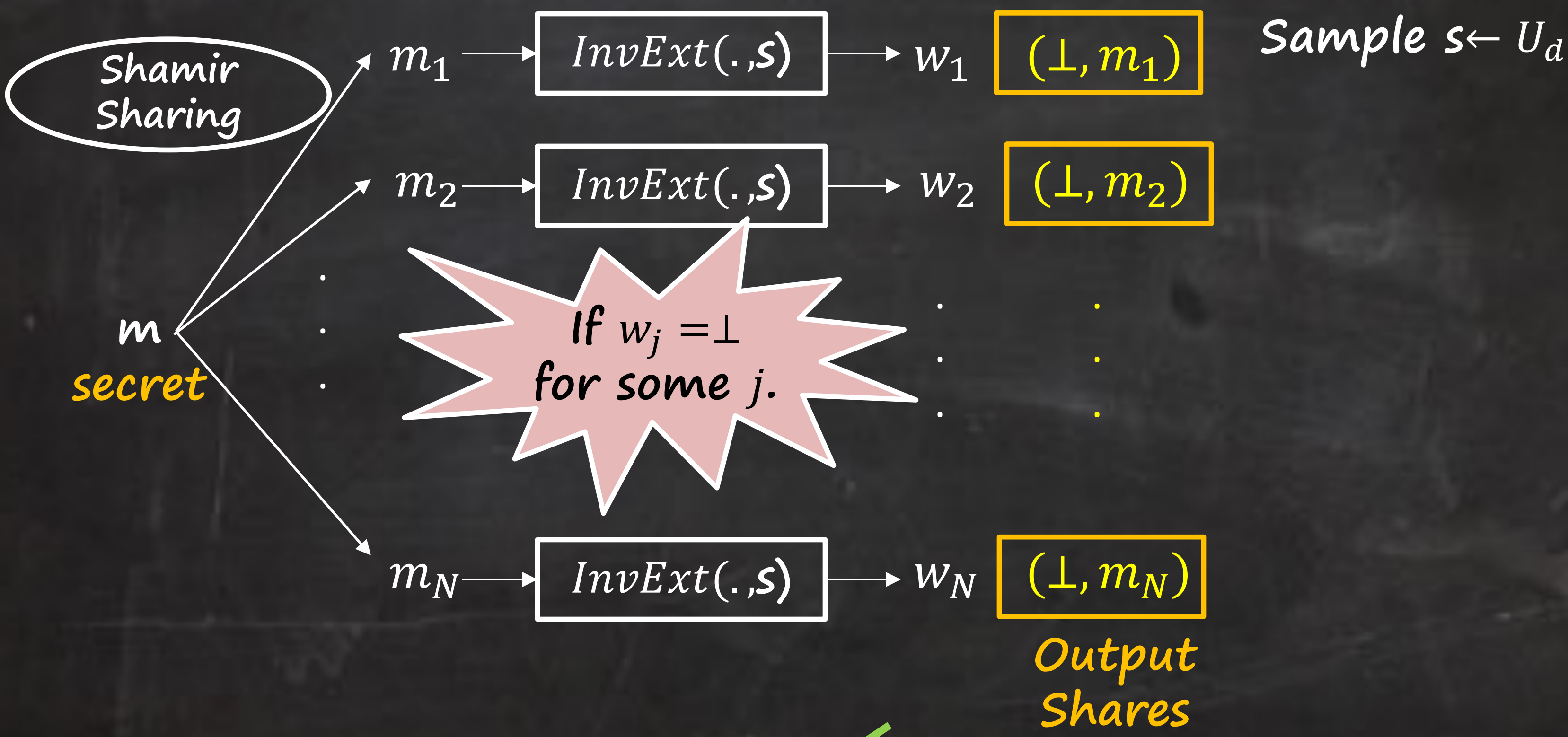
Our Construction

Optimal Threshold LRSS



Our Construction

Optimal Threshold LRSS



CORRECTNESS ✓

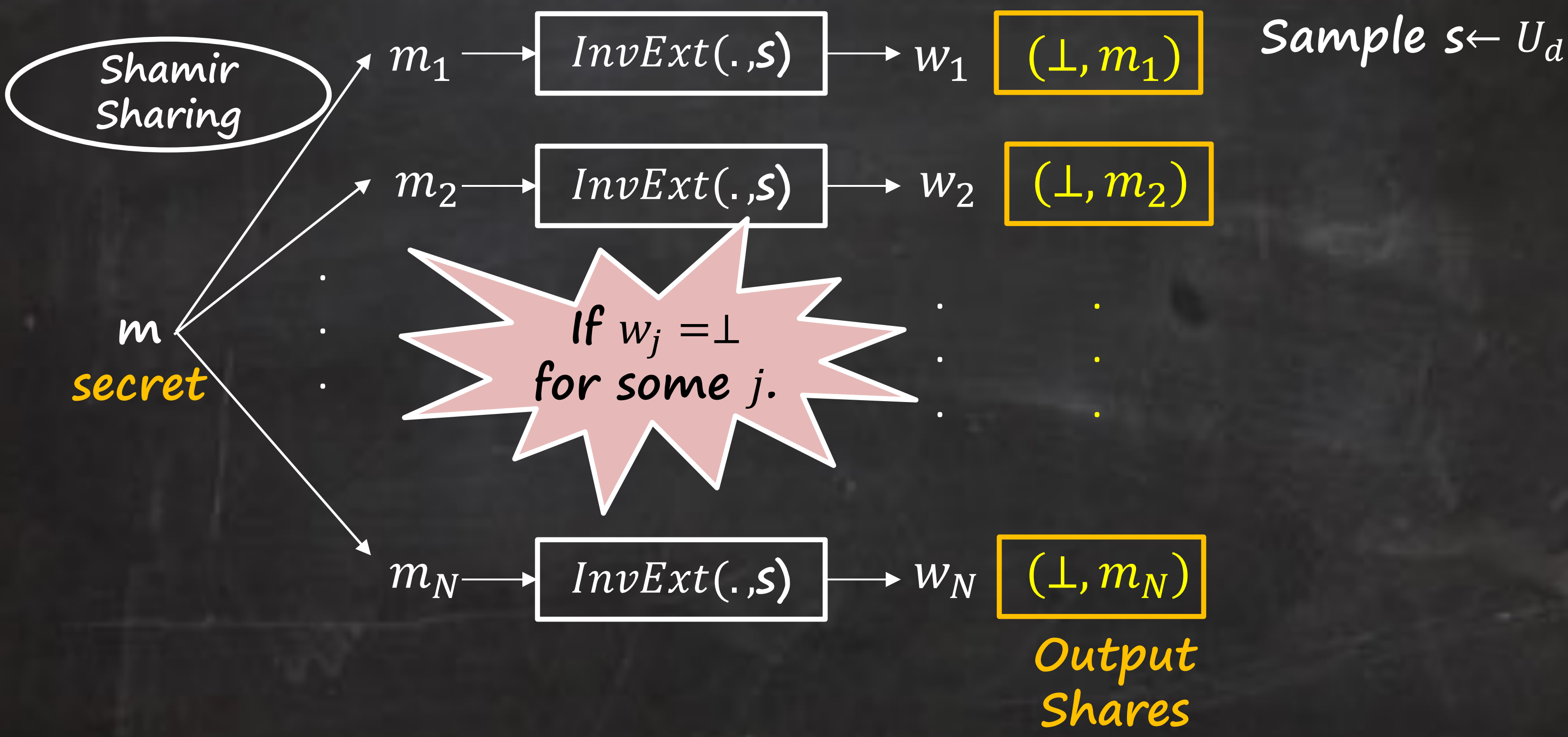
Our Construction

Optimal Threshold LRSS: Leakage Resilience

Our Construction

Optimal Threshold LRSS: Leakage Resilience

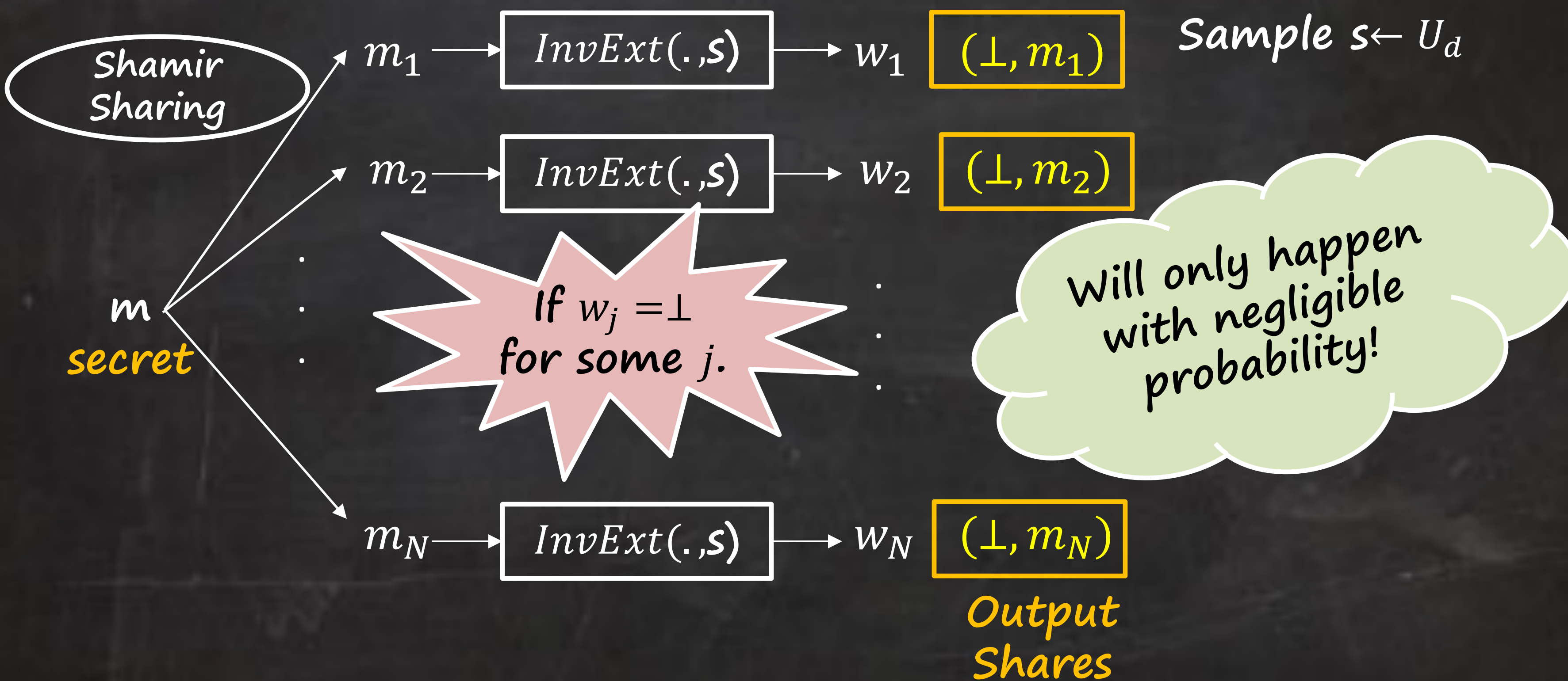
CASE I



Our Construction

Optimal Threshold LRSS: Leakage Resilience

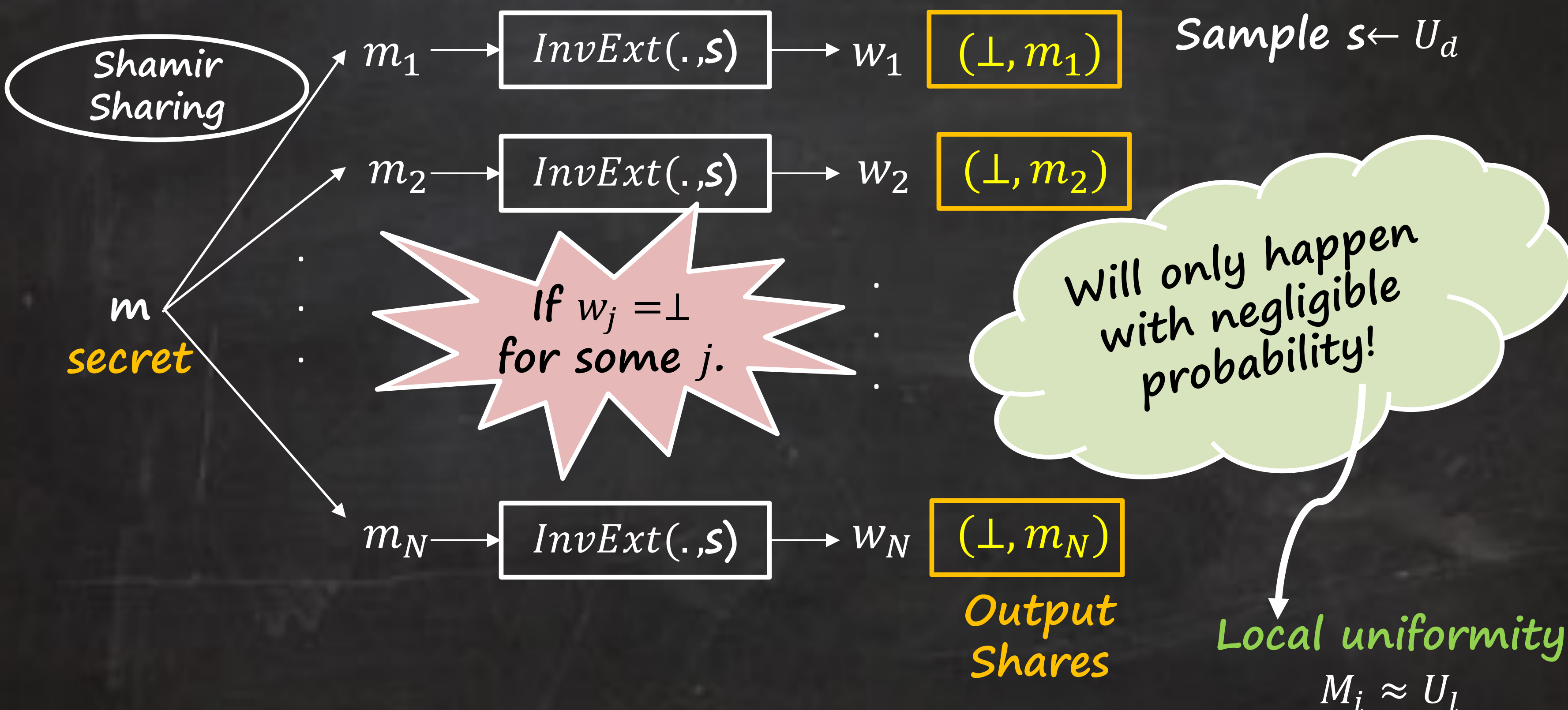
CASE I



Our Construction

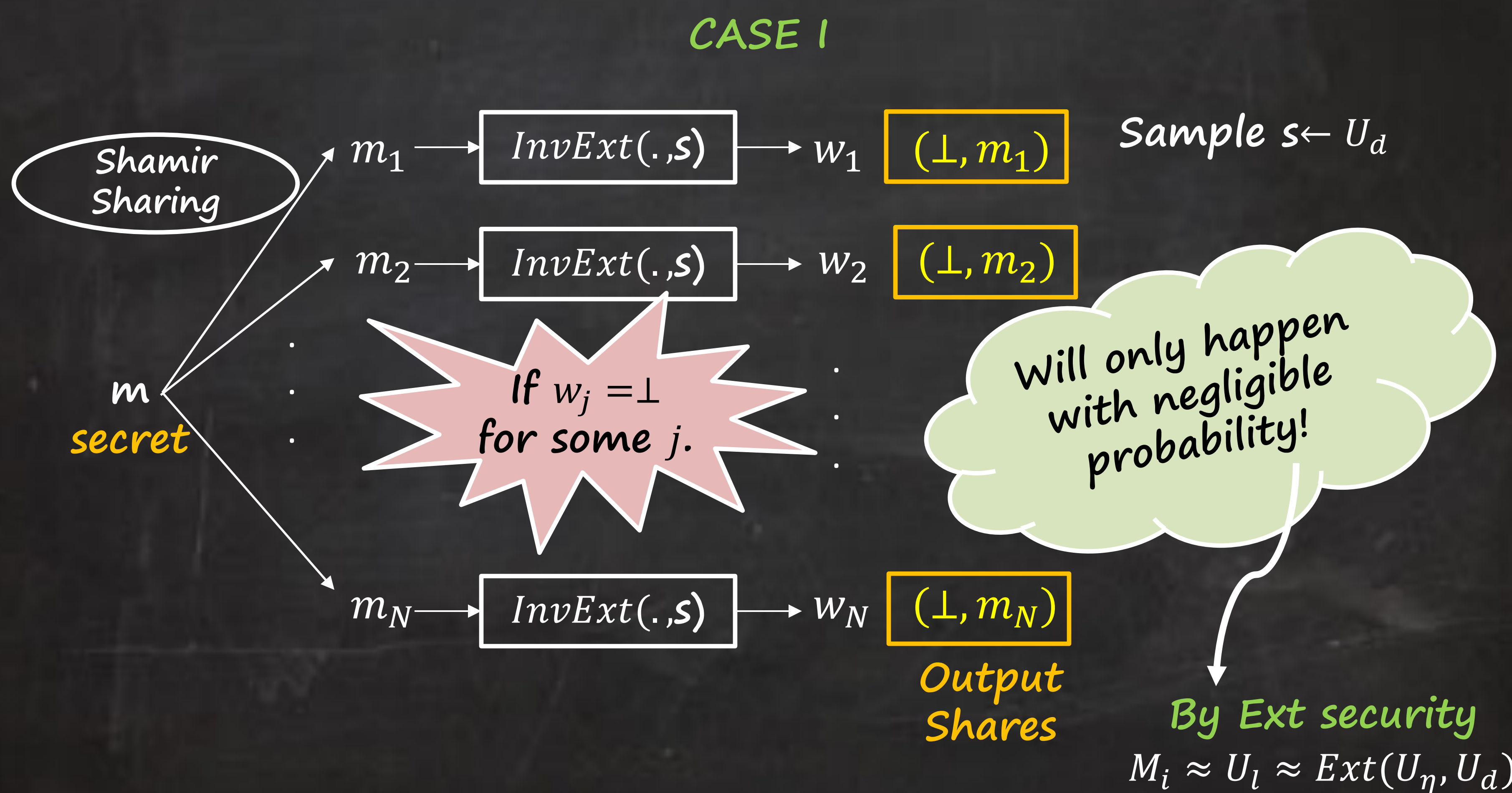
Optimal Threshold LRSS: Leakage Resilience

CASE I



Our Construction

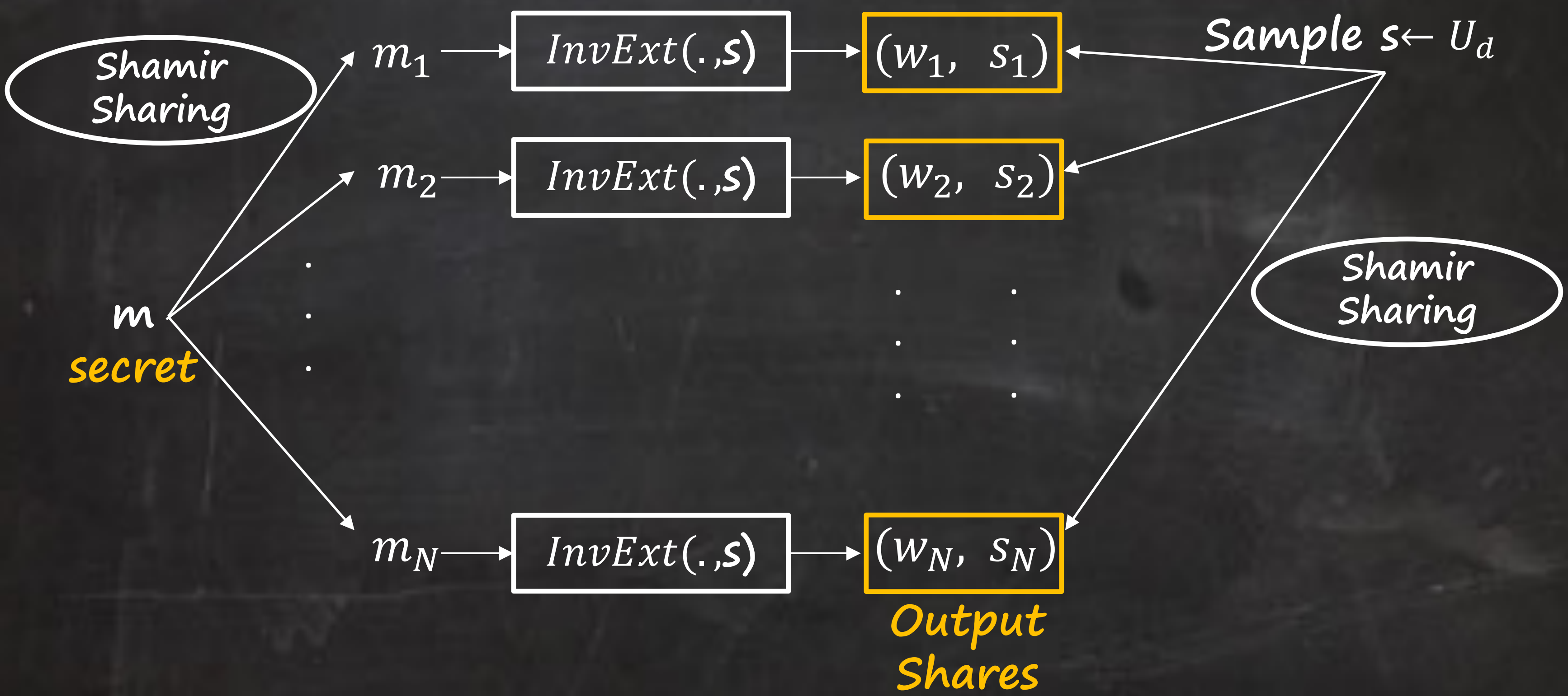
Optimal Threshold LRSS: Leakage Resilience



Our Construction

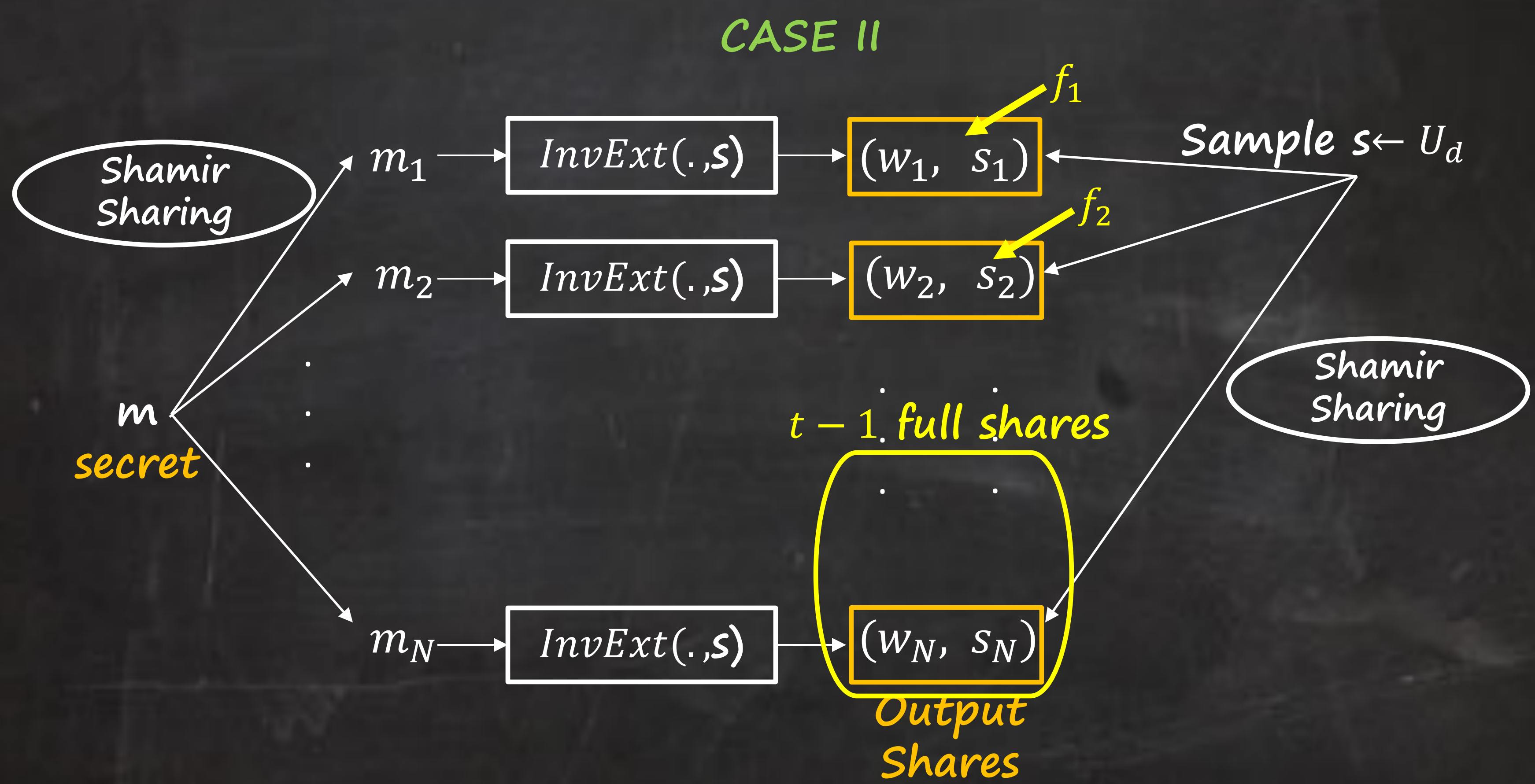
Optimal Threshold LRSS: Leakage Resilience

CASE II



Our Construction

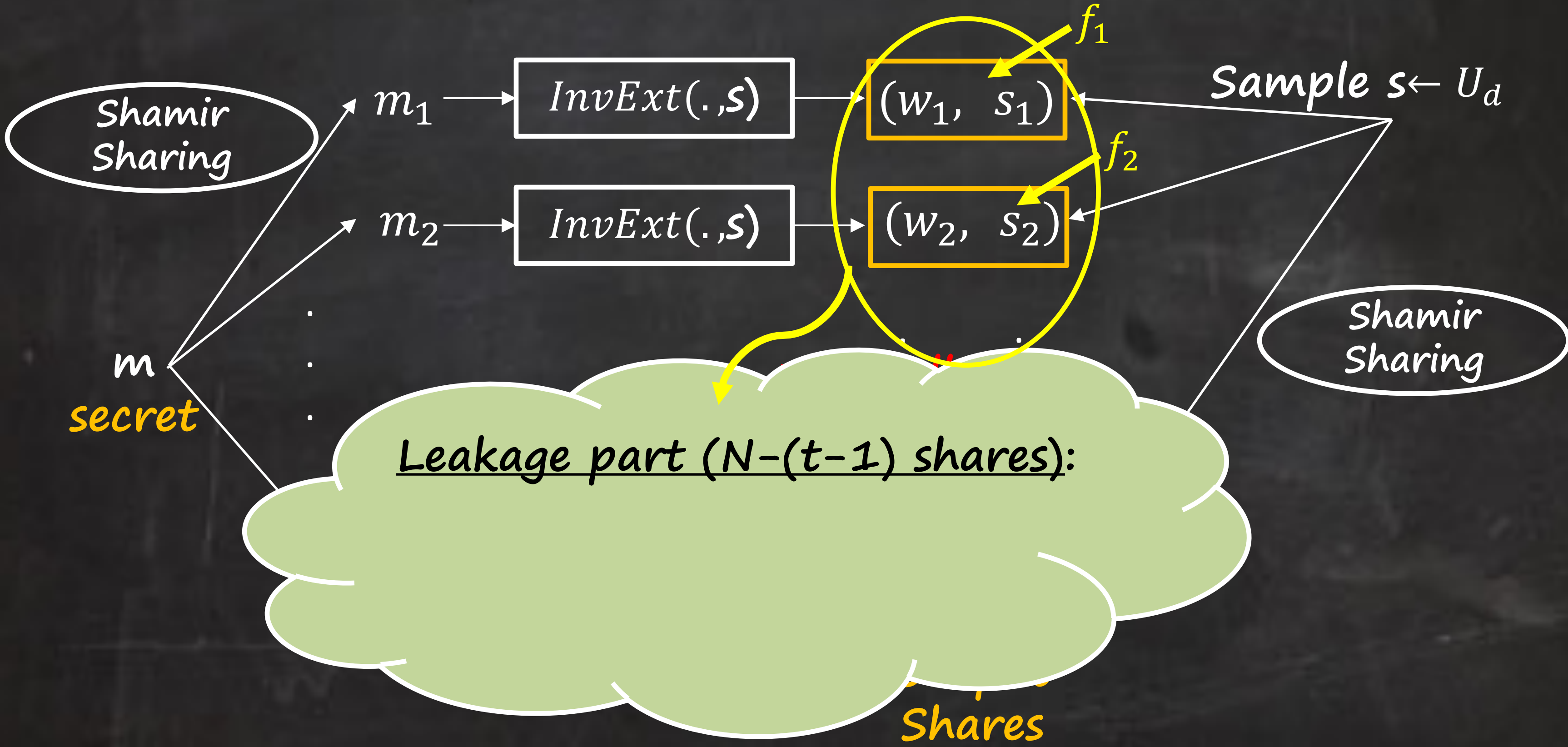
Optimal Threshold LRSS: Leakage Resilience



Our Construction

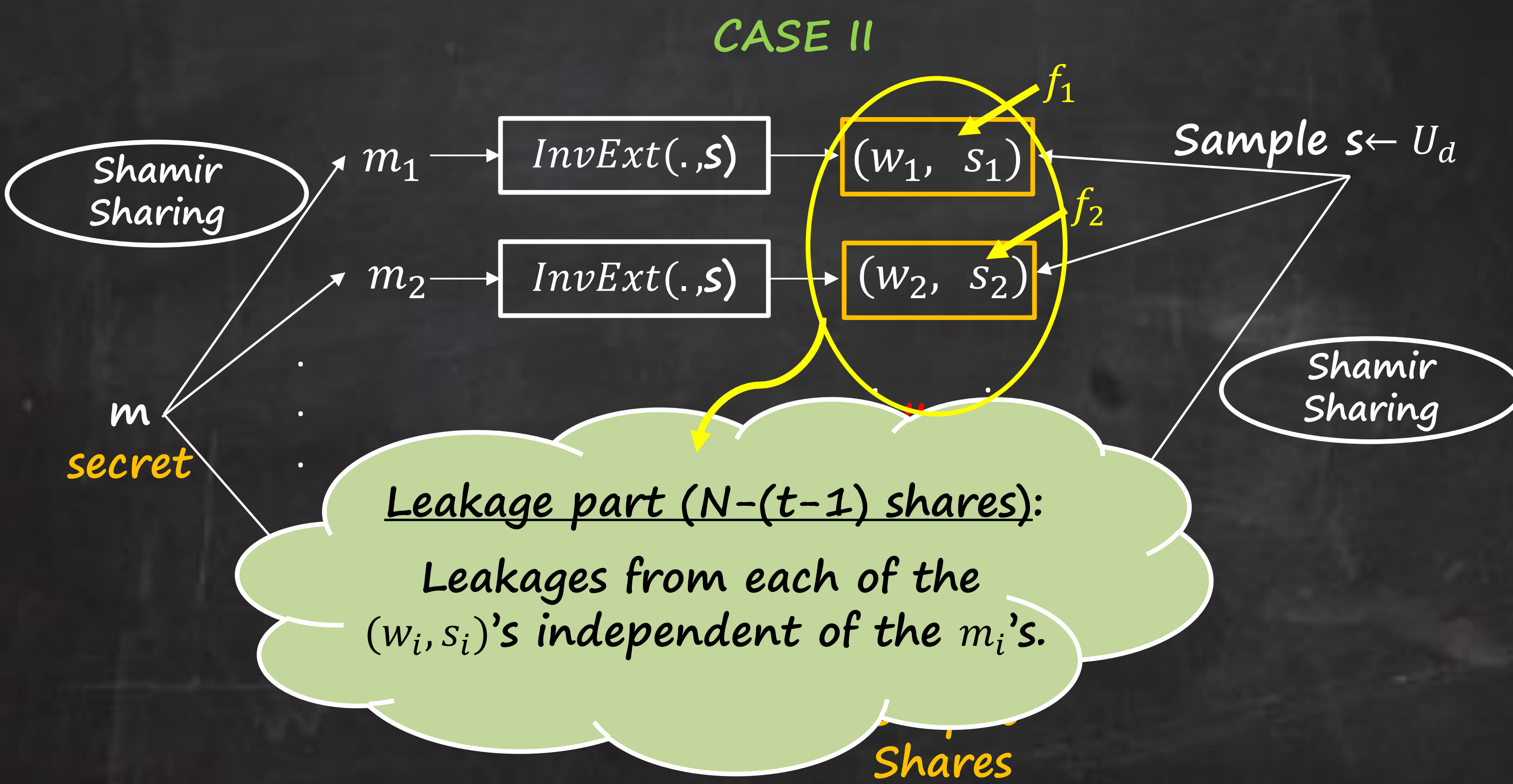
Optimal Threshold LRSS: Leakage Resilience

CASE II



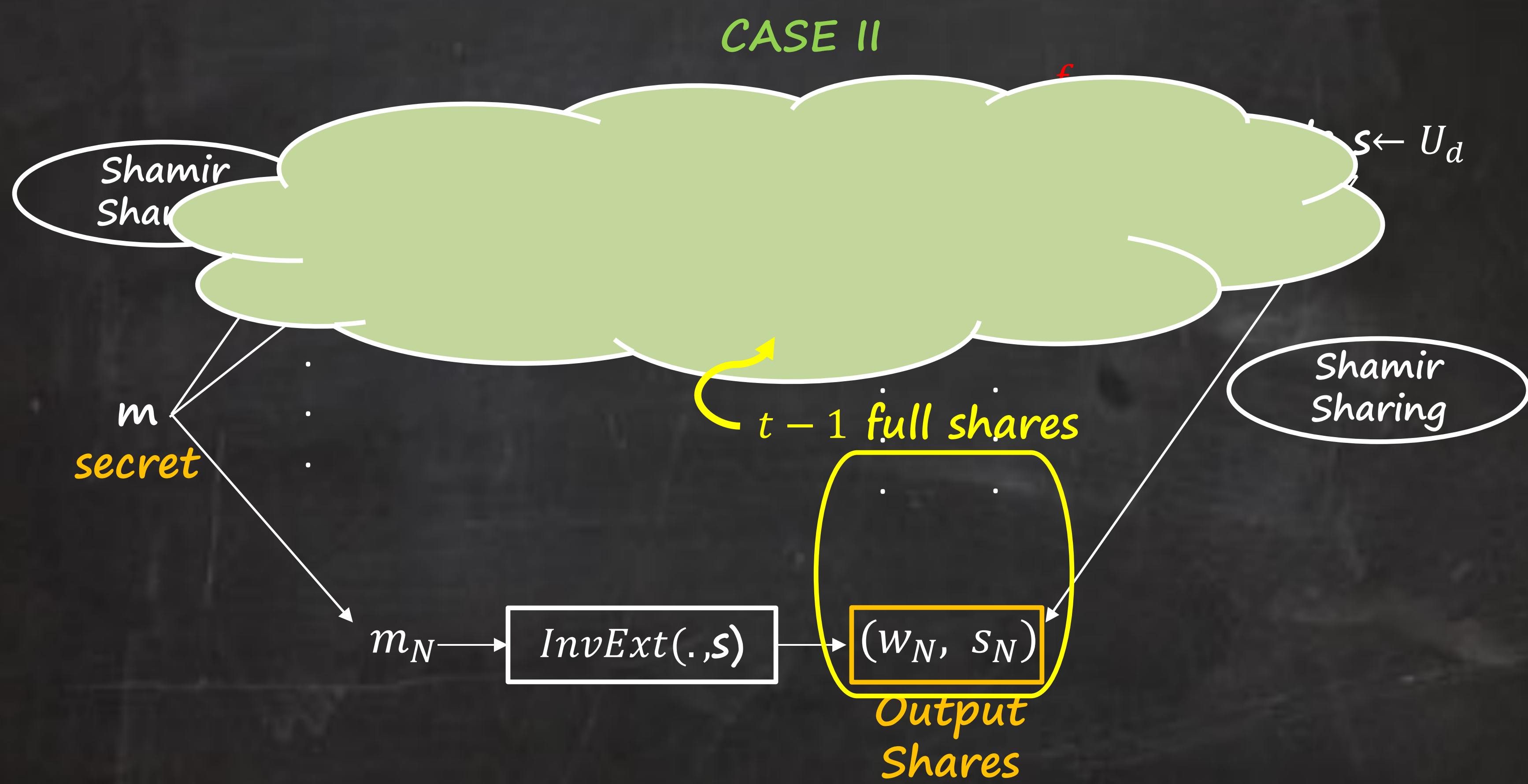
Our Construction

Optimal Threshold LRSS: Leakage Resilience



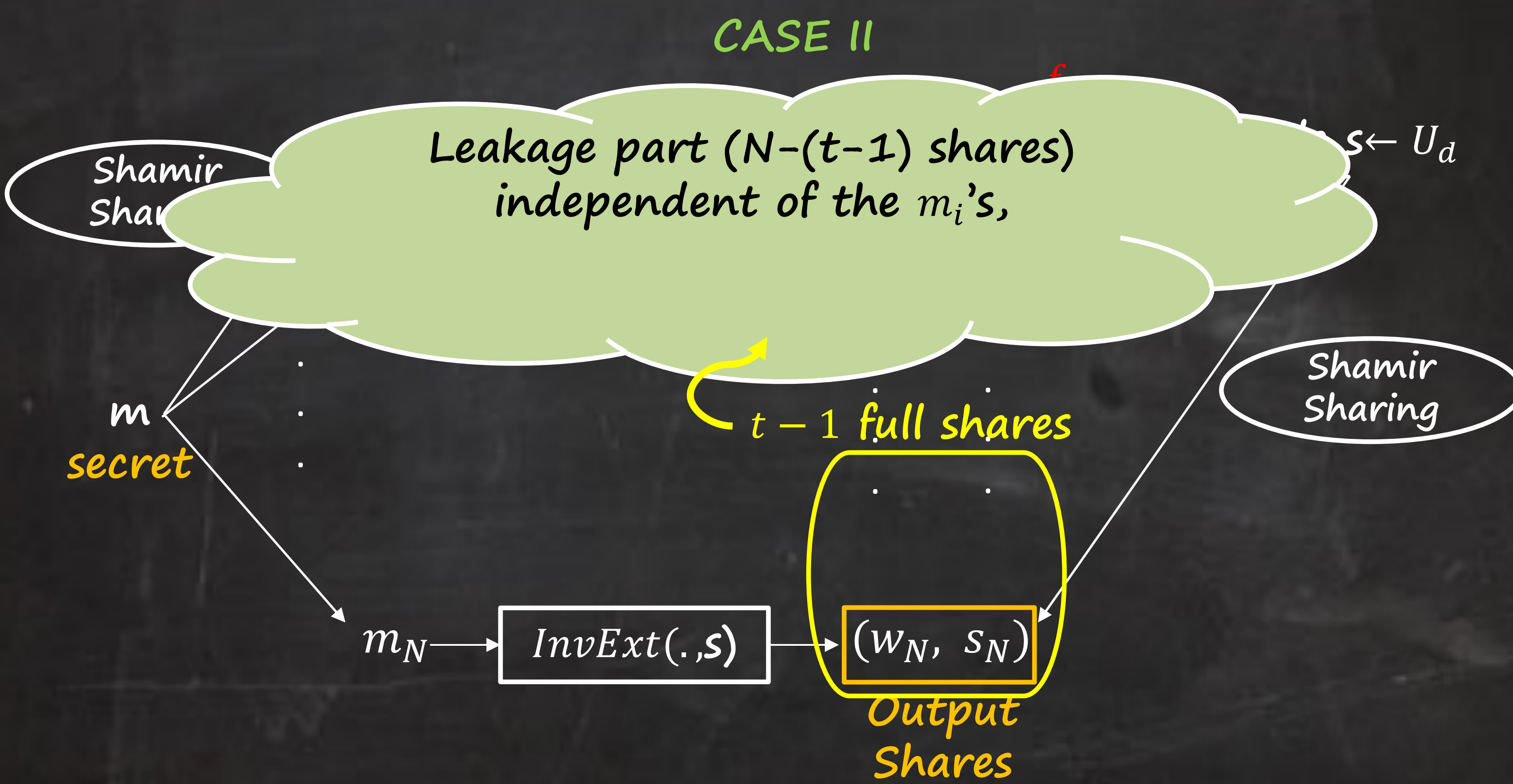
Our Construction

Optimal Threshold LRSS: Leakage Resilience



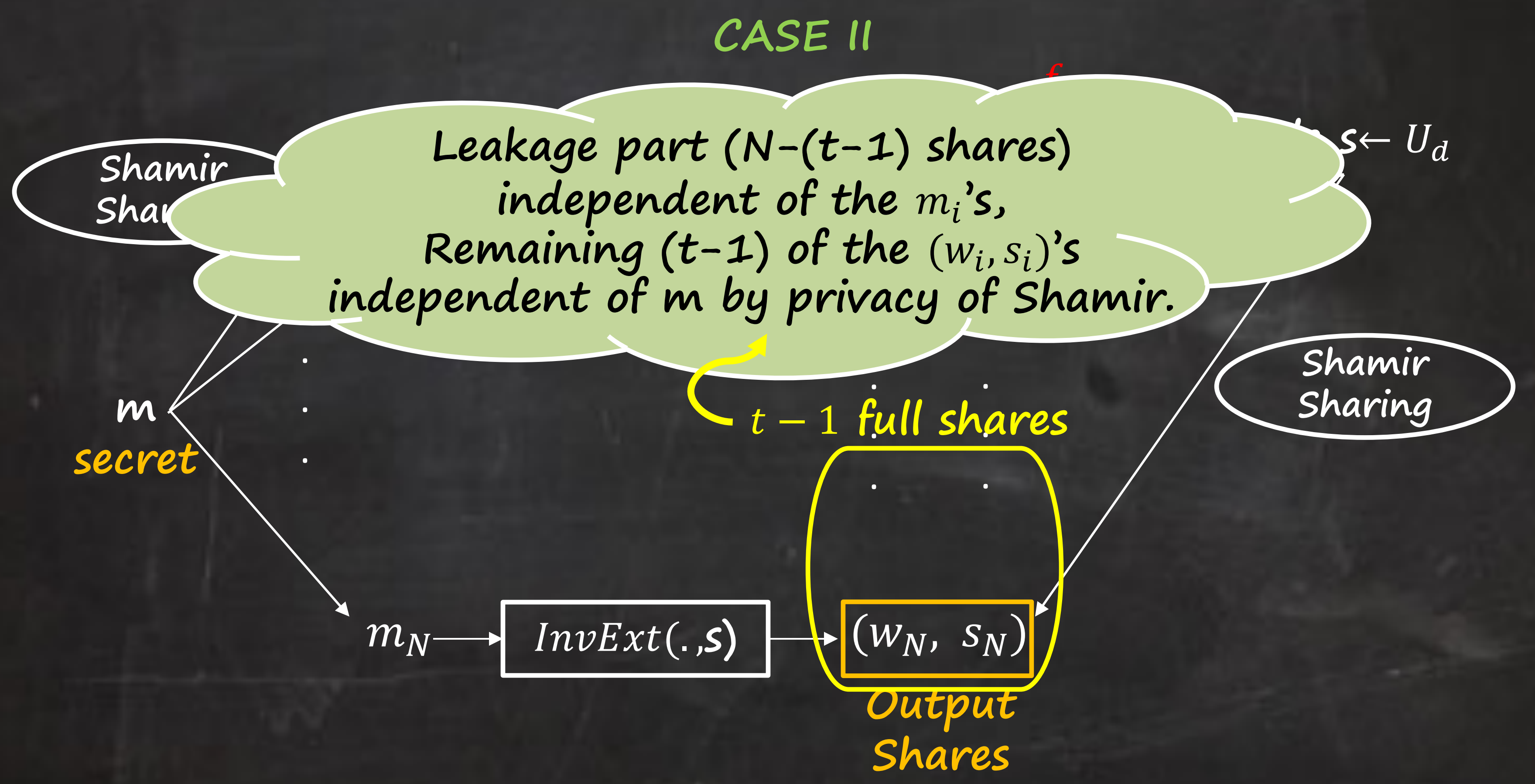
Our Construction

Optimal Threshold LRSS: Leakage Resilience



Our Construction

Optimal Threshold LRSS: Leakage Resilience



Summary

Summary

- LRSS for threshold access structure in local leakage model with optimal share size!

Summary

- LRSS for threshold access structure in local leakage model with optimal share size!
- Our compiler preserves the rate for general access structures too.

Summary

- LRSS for threshold access structure in local leakage model with optimal share size!
- Our compiler preserves the rate for general access structures too.
- We show applications to non-malleable secret sharing schemes with improved share size.

Summary

- LRSS for threshold access structure in local leakage model with optimal share size!
- Our compiler preserves the rate for general access structures too.
- We show applications to non-malleable secret sharing schemes with improved share size.
- Open- Can we get this for stronger leakage models?

Summary

- LRSS for threshold access structure in local leakage model with optimal share size!
- Our compiler preserves the rate for general access structures too.
- We show applications to non-malleable secret sharing schemes with improved share size.
- Open- Can we get this for stronger leakage models?

THANK YOU

eprint/2022/216