Leakage Resilient Secret Sharing

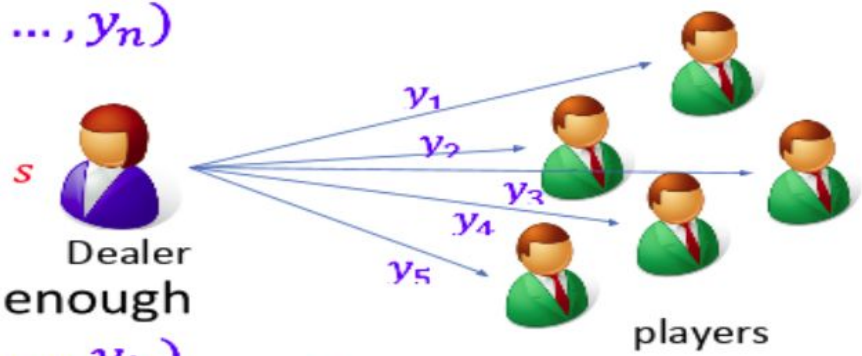
Sabyasachi Dutta

Shamir's (t,n) secret sharing

Adi Shamir

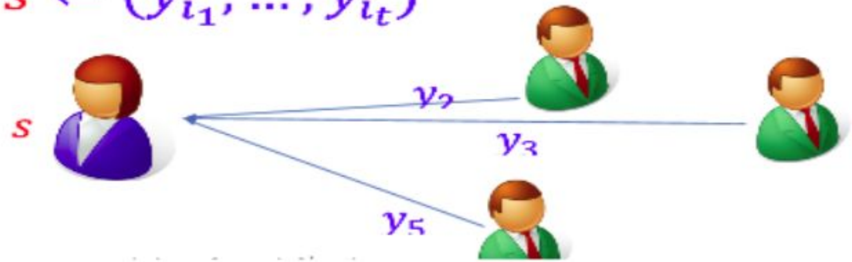
- Sharing secret s with n players

$$s \rightarrow (y_1, \dots, y_n)$$



- To recover s, t shares are enough

$$s \leftarrow (y_{i_1}, \dots, y_{i_t})$$



Sharing Phase for t = 3

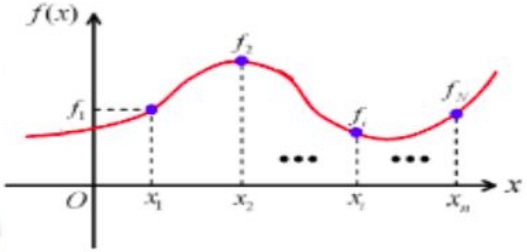
- Dealer chooses a degree $t-1$ polynomial over $\mathbb{Z}/p\mathbb{Z}$
  - s (secret to be shared) : Constant term
  - a_1, a_2 : Other coefficients chosen at random from $\mathbb{Z}/p\mathbb{Z}$ (Field)

$$f(x) = s + a_1 x + a_2 x^2 \bmod p$$

- Dealer computes shares

$$y_i := f(x_i), i = 1, \ldots, n$$

- Dealer distributes shares to n players

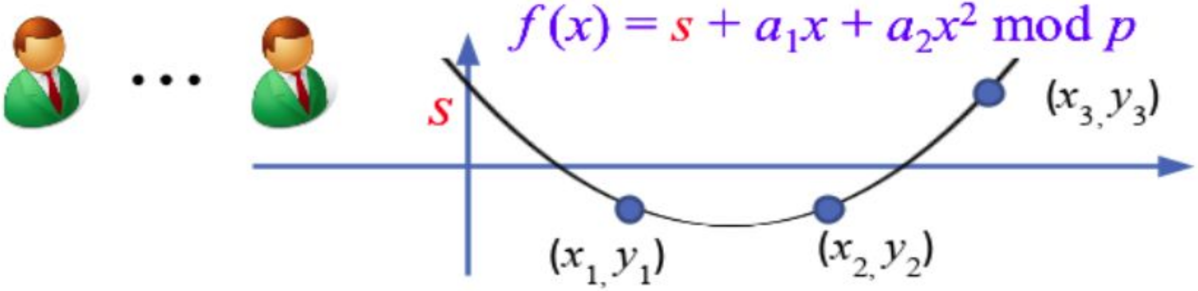


Recovery Phase t = 3

- Idea: From $t = 3$ points, compute the degree $t-1$ curve
  - $t = 3$ players are identified by x–values, $x_1 < x_2 < x_3$
  - $t = 3$ shares are y–values, $y_1,\ y_2,\ y_3$
  - Unknown, degree $t-1$ curve $y = f(x)$ can be determined from $t = 3$ points, $(x_1, y_1), (x_2, y_2), (x_3, y_3)$

Secret s is determined as the constant term!

$$f(x) = s + a_1 x + a_2 x^2 \bmod p$$



Two main properties:

- **Correctness :** Any **t** shares must recover the secret **s**

- **Secrecy :** Any **t-1** shares **must not reveal** any information about the **secret s**

- **Secrecy :** Any **t-1** shares **must not reveal** any information about the **secret s**

$$Sh[i_1] , Sh[i_2] , \ldots\ldots\ldots\ldots , Sh[i_{t-1}]$$

S = 0 **???**	**S = 1** **???**	....	**S = p-1** **???**

- **Secrecy :** Any **t-1** shares **must not reveal** any information about the **secret s**

$Sh[i_1]$, $Sh[i_2]$, …..

All values are equally probable as secret

S = 0 ???	S = 1 ???	……… ……….	S = p-1 ???

Threshold Secret Sharing

- Numerous Applications
  - ➢ Secure multiparty computation [GMW87, BGW88, CCD88,...]
  - ➢ Threshold cryptographic primitives [DF90,Fra90, ….]

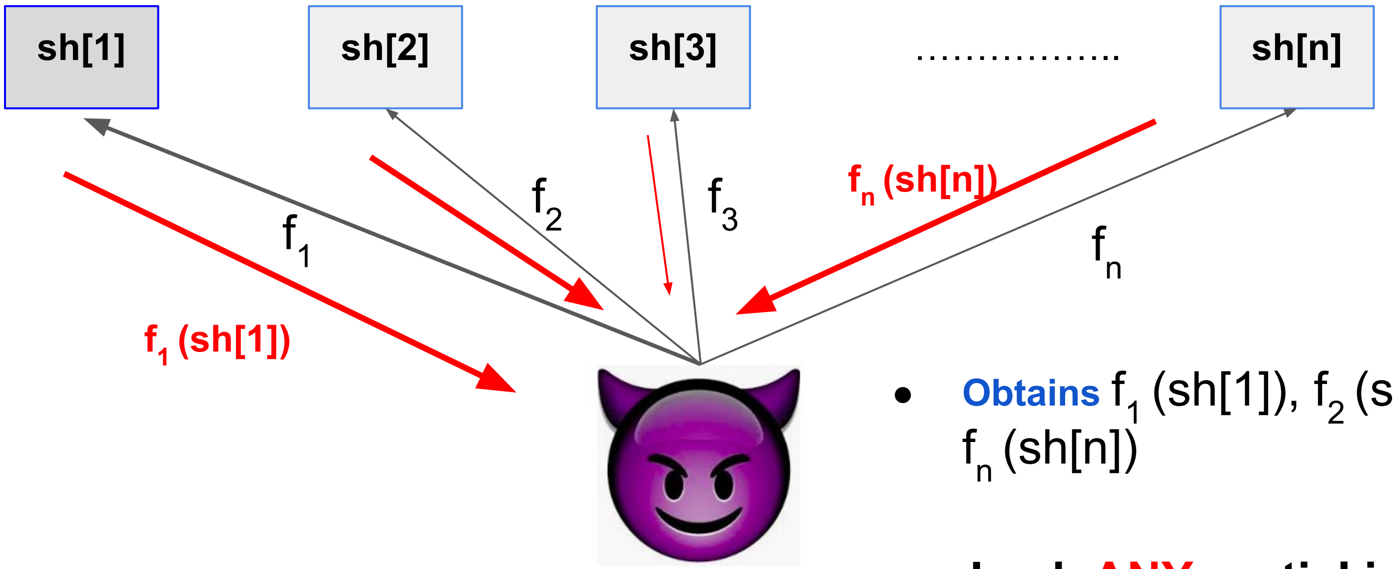
Security of these applications crucially depends on the SECRECY property of secret sharing

- **n-out-of-n** secret sharing scheme ensures even if **n-1** shares are obtained by adversary, it cannot gain any information about the secret value [very strong guarantee]

- What if all the shares are obtained by adversary? [No hope]

- What if adversary learns some partial information about (honest) all shares ?

Twist in the story (Introducing leakage)



- **Obtains** f_1 (sh[1]), f_2 (sh[2]) , … f_n (sh[n])

- **Leak ANY partial information**
- **Output of each f_i is SMALL**

Is this model of (LOCAL) leakage reasonable?

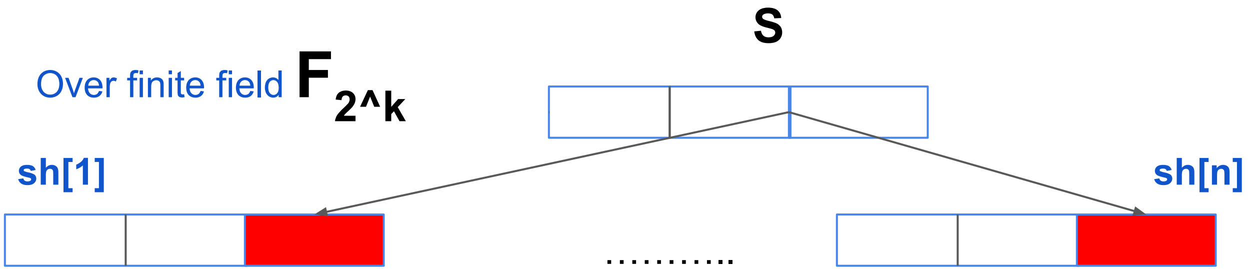
- Physical Separation of servers where the shares are stored

- Shrinked output of leakage

- Adversarial leakage i.e. the adversary gets to choose the leakage functions independent of each other

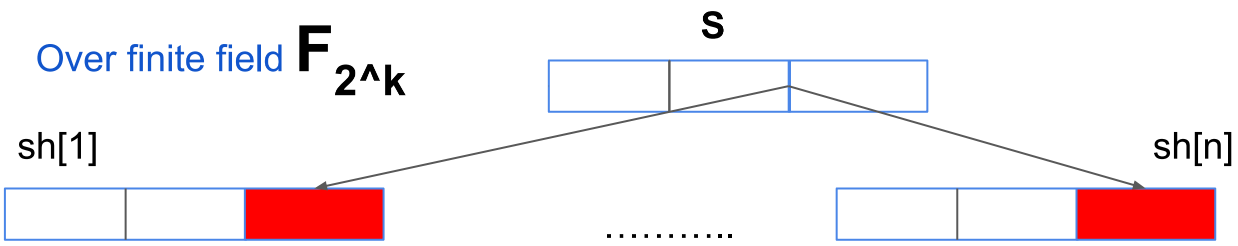
Shamir scheme not leakage resilient

Over finite field F_{2^k}



Shamir scheme not leakage resilient

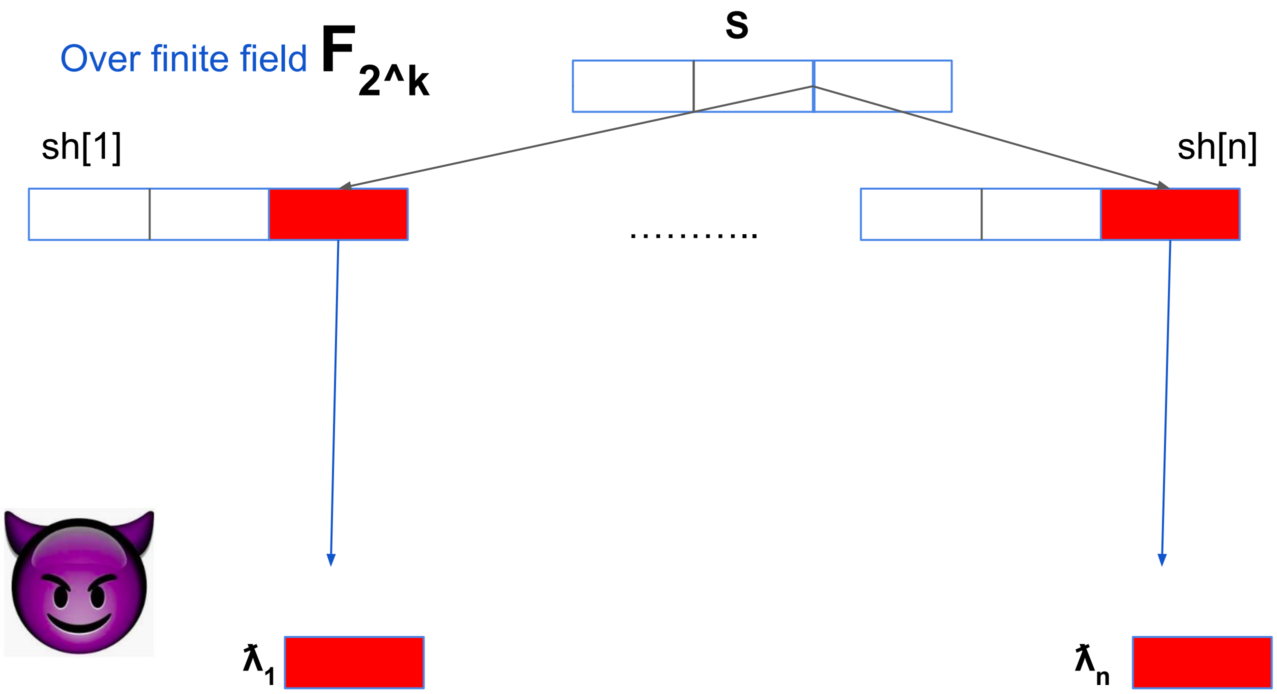
Over finite field $\mathbf{F}_{2^k}$



Lagrange interpolation for recovery

$$\mathbf{S = \lambda_1 sh[1] + \ldots + \lambda_n sh[n]}$$

Shamir scheme not leakage resilient



Lagrange interpolation

$$S = ƛ_1 sh[1] + \ldots\ldots + ƛ_n sh[n]$$

Modelling the leakage

- **Local / Independent leakage [GW 2016, BDS+ 2018, SV 2019]**

Guruswami-Wootters 2016 : One bit leakage from every server can reconstruct the secret

- **Joint leakage [SV 2019]**

- **Adaptive leakage [KMS 2019]**

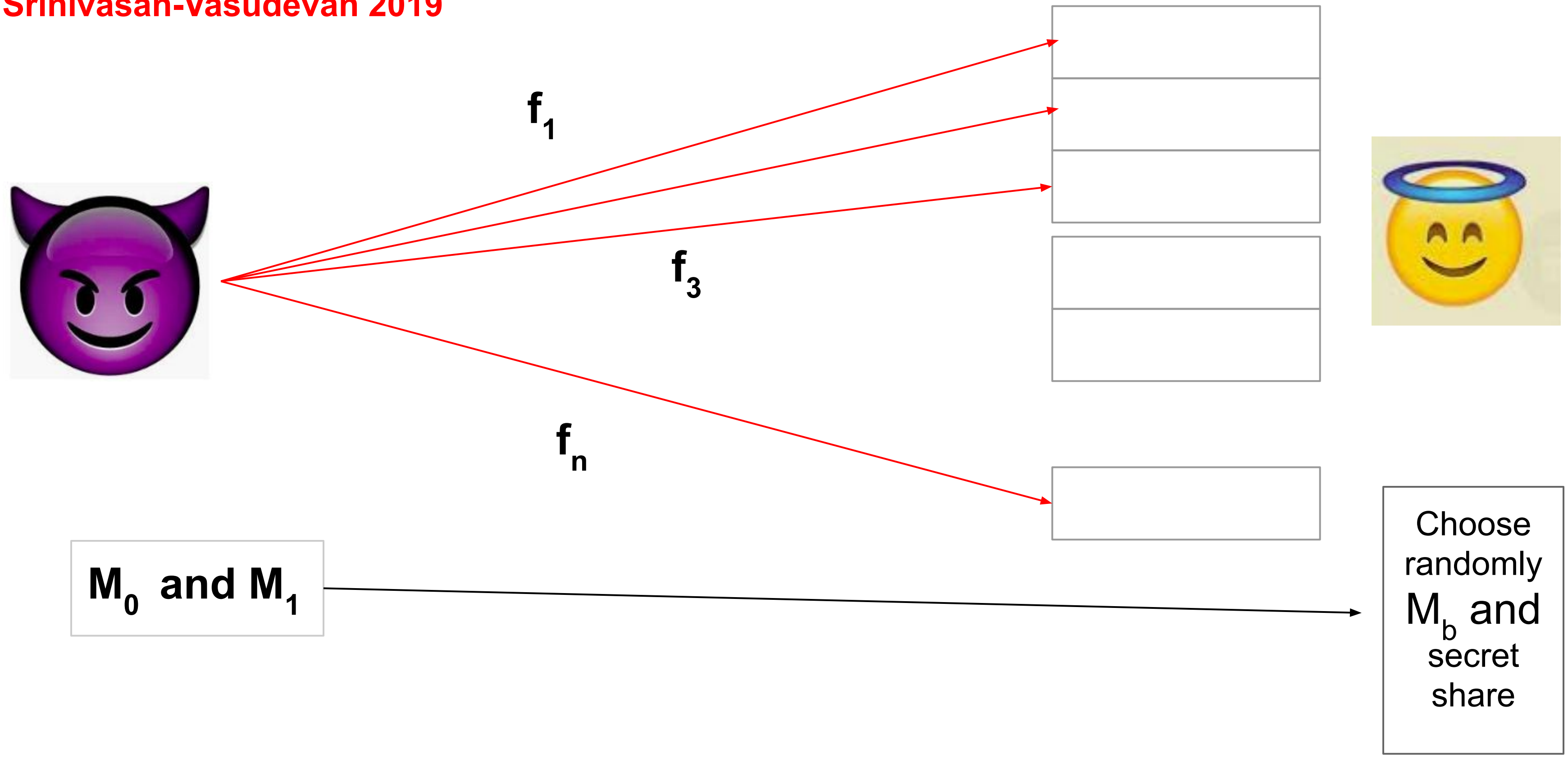
Stronger models of leakage

Results with respect to Local Leakage

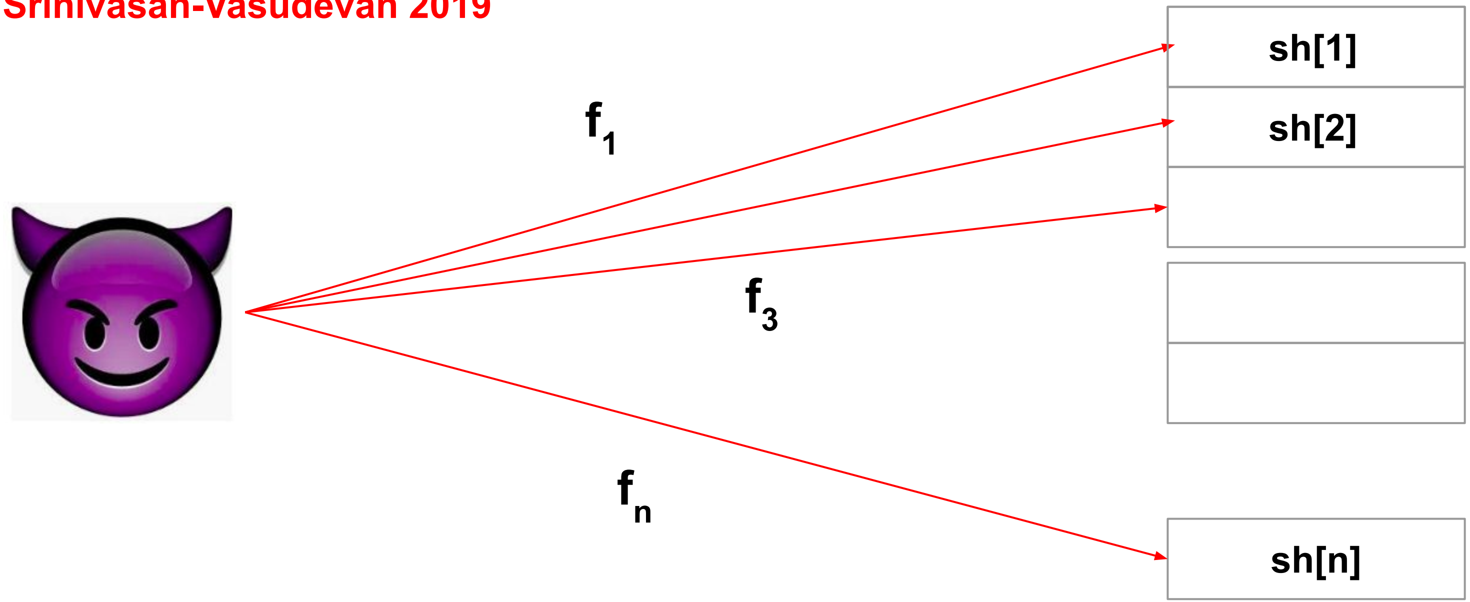
- **Benhamouda et al. 2018** :
  - ➢ Shamir scheme is LR if field is of size large prime p
  - ➢ Threshold is high **n - o(log n)**
  - ➢ Leakage bound **Ω (log p)** bits

- **Srinivasan-Vasudevan 2019**:
  - ➢ Compiler to make (t,n) Shamir scheme leakage resilient where t > 1
  - ➢ Uses average case strong seeded Extractor

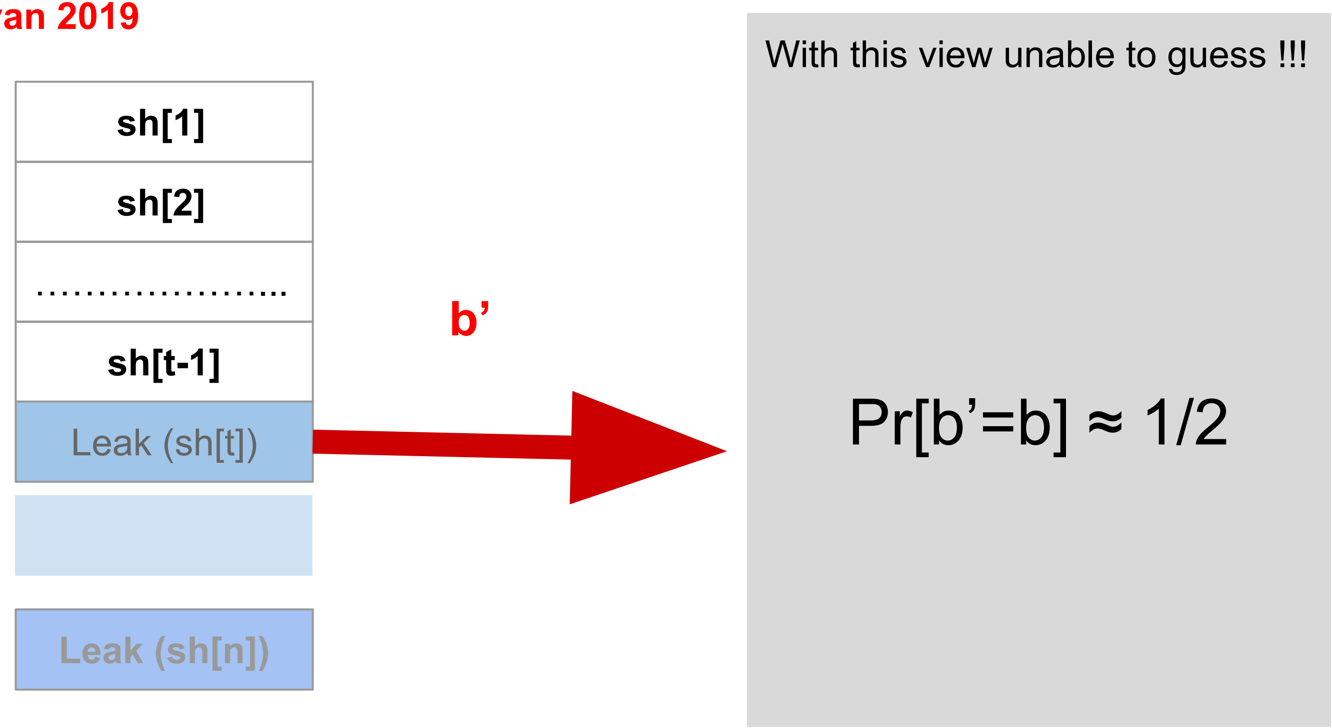
Srinivasan-Vasudevan 2019



Srinivasan-Vasudevan 2019



Srinivasan-Vasudevan 2019



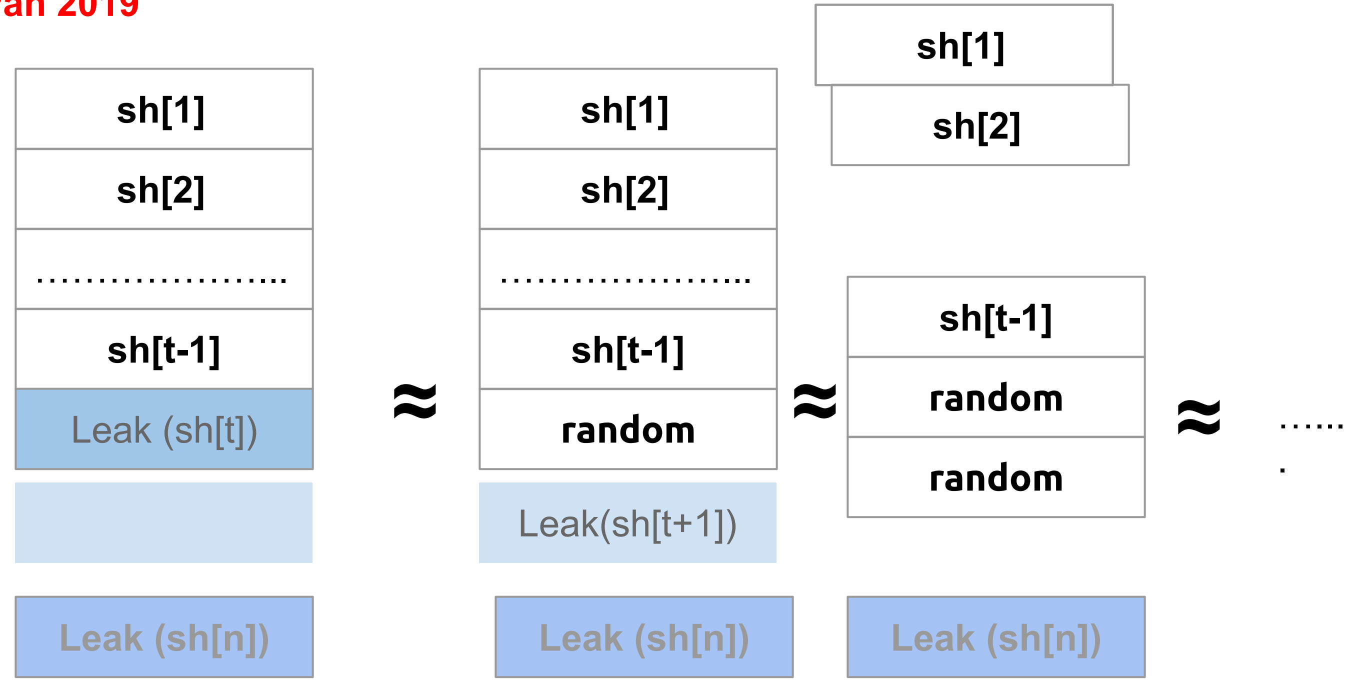
Srinivasan-Vasudevan 2019

With this view unable to guess !!!

- The secret is (statistically) hidden even when the adversary has leakage information from all shares

- View of Adv. when M_0 is secret shared ≈ View of Adv. when M_1 is secret shared

Srinivasan-Vasudevan 2019



Main component of the construction

Extractors are used to act like "one-time pad"

Definition (Strong seeded extractor). *A function* Ext $: \{0,1\}^n \times \{0,1\}^d \longrightarrow \{0,1\}^m$ *is called a strong seeded extractor for sources with min-entropy* k *and error* ϵ *if for any* (n,k)*-source* X *and an independently & uniformly chosen random string* U_d*, we must have*

$$\text{Ext}(X, U_d)||U_d \approx_\epsilon U_m||U_d,$$

where U_m *and* U_d *are independent.*

- **Initial Setup & Input :** secret message m & a (t,n)-threshold access structure. Also suppose that $(\mathsf{ShareGen}_{(t,n)}, \mathsf{Rec}_{(t,n)})$ denote a perfect (t,n)-threshold Shamir secret sharing scheme and let $\mathsf{Ext} : \{0,1\}^\eta \times \{0,1\}^d \longrightarrow \{0,1\}^\rho$ be a $(\eta - \mu, \epsilon)$ average-case, strong seeded extractor.

- **Share Generation ($\mathsf{LRShareGen}$):**
  1. Run $\mathsf{ShareGen}_{(t,n)}(m) \longrightarrow (\mathsf{sh}[1], \ldots, \mathsf{sh}[n])$.
  2. Choose a uniform seed $s \in_R \{0,1\}^d$ and a masking string $r \in_R \{0,1\}^\rho$.
  3. For each $i = 1, 2, \ldots, n$ do:
     $w_i \in_R \{0,1\}^\eta$
     Compute: $\mathsf{sh}[i] \oplus Ext(w_i, s) \oplus r$
  4. Run $\mathsf{ShareGen}_{(2,n)}(s, r) \longrightarrow (s_1, \ldots, s_n)$
  5. Output: $\mathsf{sh}_i = (w_i, \mathsf{sh}[i] \oplus Ext(w_i, s) \oplus r, s_i)$

Joint leakage model

sh[1]
sh[2]
....
sh[t-1]
Leak (sh[t])
Leak (sh[n])

Leaks depend on any t-2 shares

sh[1], sh[2], …. ,sh[t-2]

cannot depend on t-1 shares !!! (Trivial Attack)

Modelling Adaptive Leakage [KMS 2019]

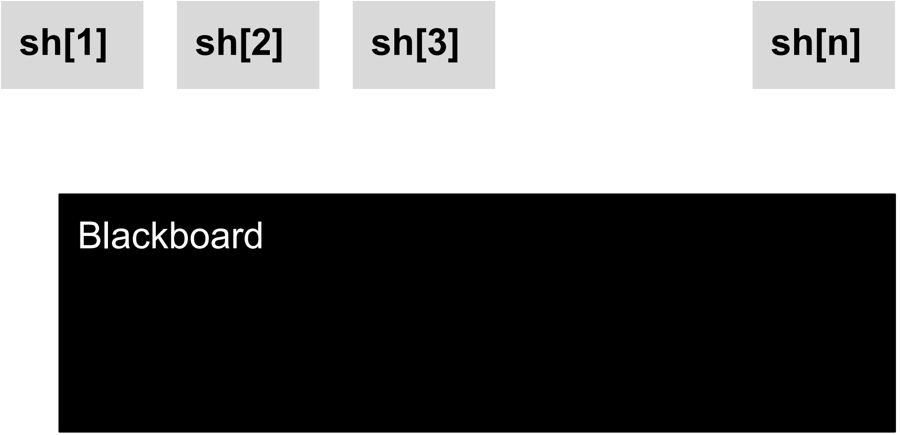
Adversary runs a multi party communication protocol and learns "transcript"

- **Total number of bits communicated is bounded**
- **Certain types of protocols are allowed (**Bounded collusion protocols**)**

Bounded Collusion Protocols (**BCP**)

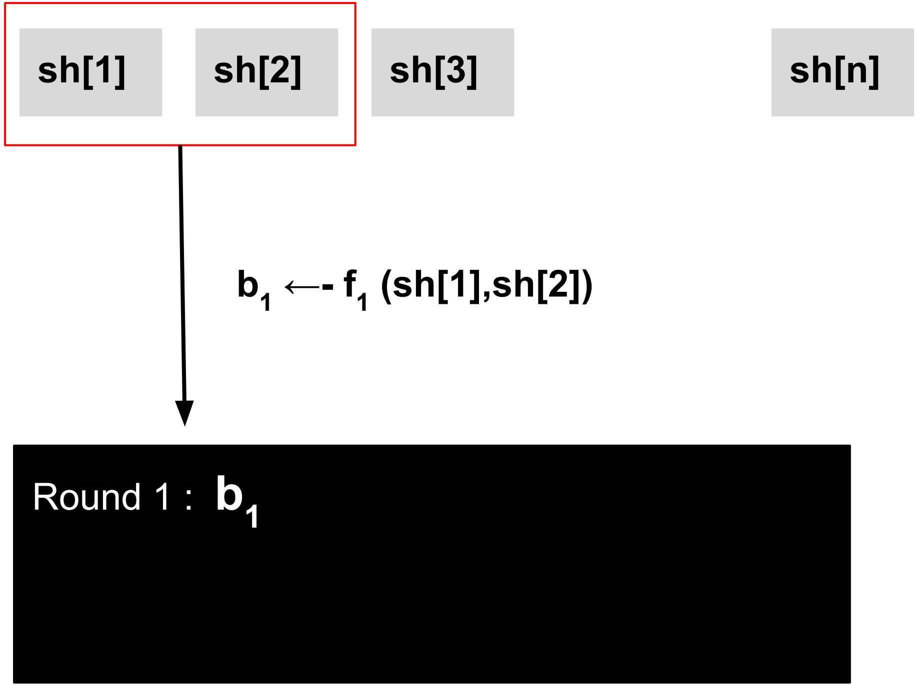
p -- party Collusion Protocol

Each round p parties collude and write a bit on the public board



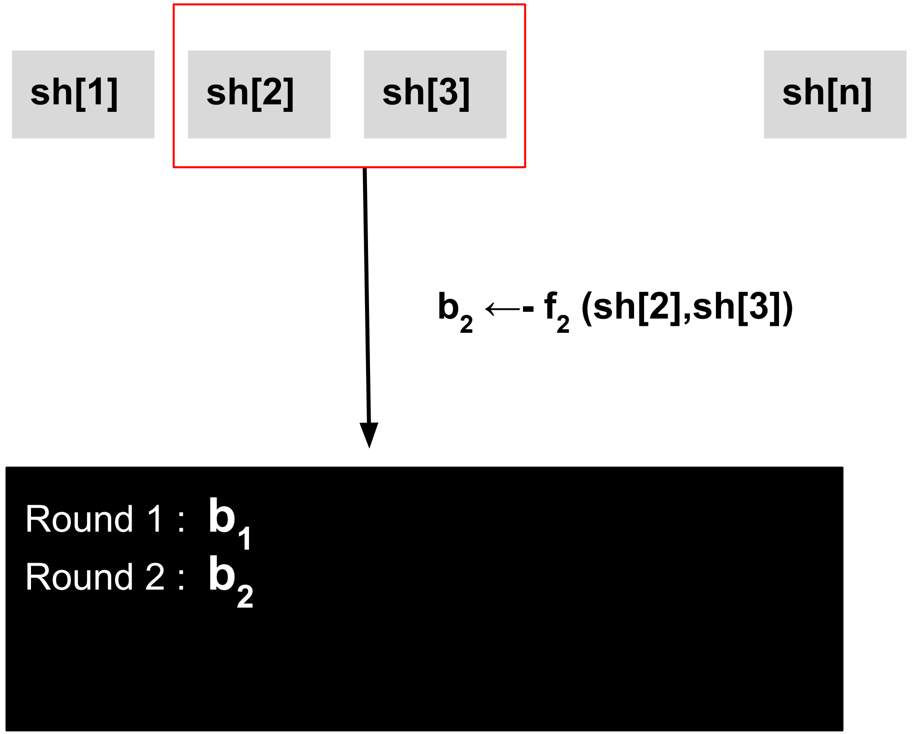
- **p = collusion bound**
- **μ = leakage bound**

Each round p = 2 parties collude and write a bit on the public board



- p = collusion bound
- μ = leakage bound

Each round p = 2 parties collude and write a bit on the public board



- p = collusion bound
- μ = leakage bound

Each round p = 2 parties collude and write a bit on the public board

sh[1] | sh[2] | sh[3] | sh[n]

- p = collusion bound
- μ = leakage bound

Round 1 : b_1
Round 2 : b_2

Round μ : bμ

Each round p = 2 parties collude and write a bit on the public board

sh[1] | sh[2] | sh[3] | sh[n]

- p = collusion bound
- μ = leakage bound

Round 1 : b_1
Round 2 : b_2

Round μ : b_μ

Advantages:

- **Joint leakage**
- **Overlapping leakage**
- **Adaptive**

BCP in communication complexity

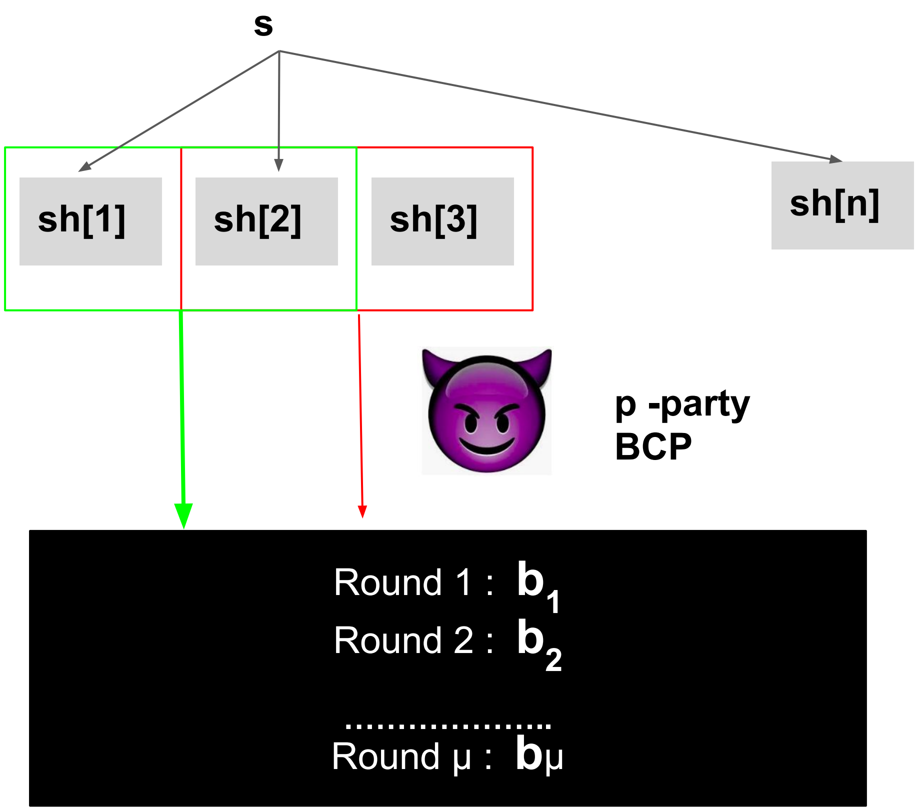
- 1 - party collusion protocol : **Number in hand** (NIH)

- (n-1) - party collusion protocol : **Number on forehead** (NoF)

  [Chandra-Furst-Lipton 1983]

Leakage resilient secret sharing w.r.t p-party BCP ??

Leakage resilience against BCPs



- **(p,t,n)-LRSS**

- **Any t can recover s**
- **t-1 can not**

Leakage Resilience

Secret statistically hidden given p- party BCP transcript

p= t-1 is the worst possible adversary

- Main technique : Choose a function f : ({0,1}^b)^n ----------> {0,1} such that communication complexity (**NoF**) of f > **μ**

1. **Share generation:** On input a secret bit m
   - sample uniformly & independently $r_i \in \{0,1\}^b$ for all $i = 1, \ldots, \rho$
   - compute the bit $r \longleftarrow f(r_1, \ldots, r_\rho)$
   - compute s such that $s \oplus r = m$
   - sample uniformly and independently $s_1, \ldots, s_\rho \in \{0,1\}$ and find s_ρ such that $s_1 \oplus \cdots \oplus s_\rho = s$
   - Output $\mathsf{share}_i = (r_i, s_i)$ for all $i = 1, \ldots, \rho$

Main Results

- When t-1 parties are under Adversarial control

➢ Compiler to convert (t,n) Shamir scheme to LR (t,n) secret sharing scheme [**SV19**, **ADK+19**]

➢ Construction of LR (t,t) secret sharing scheme
➢ LR (t,n) secret sharing scheme
➢ LR t-monotone general access structure

[KMS19]

Main Results

- Wh… …trol

Local Leakage, Joint Leakage

➢ Compiler to convert (t,n) Shamir scheme to LR (t,n) secret sharing scheme [**SV19**]

Adaptive leakage through **BCP**

[KMS19]

➢ Construction of LR (t,t) secret sharing scheme
➢ LR (t,n) secret sharing scheme
➢ LR t-monotone general access structure

Our work

Extend the classes of leakage functions for general access structure

[General Access Structure does not have any particular form for qualified sets or forbidden sets]

- **Extend the idea of joint leakage model [Adv can control any forbidden set of parties/ shares]**
- **Extend the idea of (t-1) - party CP to F - party CP**
- **Compilers and scheme that are secure against these classes**

धन्यवाद

多謝

ขอบคุณ

Спасибо

Gracias

شكراً

Obrigado

Thank You

Diolch

Grazie

Danke

Merci

நன்றி

多谢

감사합니다

ありがとうございました