Linear Repairing Codes and Side-Channel Attacks

Hervé CHABANNE, Houssem MAGHREBI and Emmanuel PROUFF

IDEMIA, UL, ANSSI

Partially funded by REASSURE H2020 Project (ID 731591)

TCHES, Setember 2018

Secure implementations with secret sharing techniques.

Secure implementations with secret sharing techniques.

- First Ideas in *GoubinPatarin99* and *ChariJutlaRaoRohatgi99*.

Secure implementations with secret sharing techniques.

- First Ideas in *GoubinPatarin99* and *ChariJutlaRaoRohatgi99*.
- Soundness based on the following remark:
  - Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
  - Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
  - The number of leakage samples to test $\big((L_i)_i | x = 0\big) \stackrel{?}{=} \big((L_i)_i | x = 1\big)$ is lower bounded by $O(1)\sigma^d$.

Secure implementations with secret sharing techniques.

- First Ideas in *GoubinPatarin99* and *ChariJutlaRaoRohatgi99*.
- Soundness based on the following remark:
  - Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
  - Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
  - The number of leakage samples to test $\left((L_i)_i | x = 0\right) \stackrel{?}{=} \left((L_i)_i | x = 1\right)$ is lower bounded by $O(1)\sigma^d$.
- Theory available to prove the security in (relatively) sound models *DucDziembowskiFaust14*.

Secure implementations with secret sharing techniques.

- First Ideas in *GoubinPatarin99* and *ChariJutlaRaoRohatgi99*.
- Soundness based on the following remark:
  - Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
  - Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
  - The number of leakage samples to test $\left((L_i)_i | x = 0\right) \overset{?}{=} \left((L_i)_i | x = 1\right)$ is lower bounded by $O(1)\sigma^d$.
- Theory available to prove the security in (relatively) sound models *DucDziembowskiFaust14*.
- Tools have been developed to automatize the proofs (e.g. *BartheBelaidDupressoirFouqueGrégoireStrub15*)

- First Issue: how to share sensitive data?

- Second Issue: how to securely process on shared data?

- **First Issue:** how to share sensitive data?
- **Related to:**
  - secret sharing *Shamir79*
  - design of error correcting codes with large dual distance *Massey93, CastagnosRennerZémor13*
  - etc.

- **Second Issue:** how to securely process on shared data?
- **Related to:**
  - secure multi-party computation *NikovaRijmenSchläffer2008 ProuffRoche2011*
  - circuit processing in presence of leakage *e.g. GoldwasserRothblum2012*
  - efficient polynomial evaluation *e.g. CarletGoubinProuffQuisquater-Rivain2012, CoronProuffRoche2012, CoronRoyVivek2014*
  - etc.

- (n, d)-SSS: polynomial formulation;
  - generate a random degree-d polynomial

$$P_Z(X) = Z + R_1 X + R_2 X^2 + \ldots + R_d X^d \ ,$$

with R_1, ..., R_d chosen at random in the base field.

- (n, d)-SSS: polynomial formulation;
  - generate a random degree-d polynomial

    $$P_Z(X) = Z + R_1 X + R_2 X^2 + \ldots + R_d X^d \ ,$$

    with R_1, …, R_d chosen at random in the base field.
  - build the shares Z_i such that

    $$\boxed{Z_i = P_Z(\alpha_i)}$$

    for n different public constant values α_i.

- (n,d)-SSS: polynomial formulation;
  - generate a random degree-d polynomial

$$P_Z(X) = Z + R_1X + R_2X^2 + \ldots + R_dX^d \quad ,$$

  with R_1, …, R_d chosen at random in the base field.
  - build the shares Z_i such that

$$\boxed{Z_i = P_Z(\alpha_i)}$$

  for n different public constant values α_i.
- Reconstruction with Lagrange's Formula and a subset U of $d+1$:

$$\boxed{Z = \sum_{Z_i \in U} Z_i \times \beta_i \quad ,}$$

where the constants β_i are defined as

$$\beta_i = \prod_{k=1, k\neq i}^{n} \frac{\alpha_k}{\alpha_i + \alpha_k} \quad .$$

Choice of the Public Points α_i

Does the choice of the public points impact the security of SSS in the context of Side-Channel Analysis?

Optimal Number of Shares to Observe

In a Side-Channel Anlaysis context, what is the optimal number of shares to observe?

Choice of the Public Points α_i

Does the choice of the public points impact the security of SSS in the context of Side-Channel Analysis?

No influence on the effectiveness of Lagrange's reconstruction BUT the mutual information $(d+1)$-tuple of shares Z_i and Z seems to depend on the α_i *BalashFaustGierlichs15, WangStandaertYu+16*.

Optimal Number of Shares to Observe

In a Side-Channel Anlaysis context, what is the optimal number of shares to observe?

Choice of the Public Points α_i

Does the choice of the public points impact the security of SSS in the context of Side-Channel Analysis?

Optimal Number of Shares to Observe

In a Side-Channel Anlaysis context, what is the optimal number of shares to observe?

Since the knowledge of $d+1$ shares Z_i is sufficient to recover Z, it is commonly assumed that the optimal number is $d+1$.

Test of template attacks against a $(5, 2)$-SSS $(Z_0, Z_1, \ldots, Z_4)$ of Z

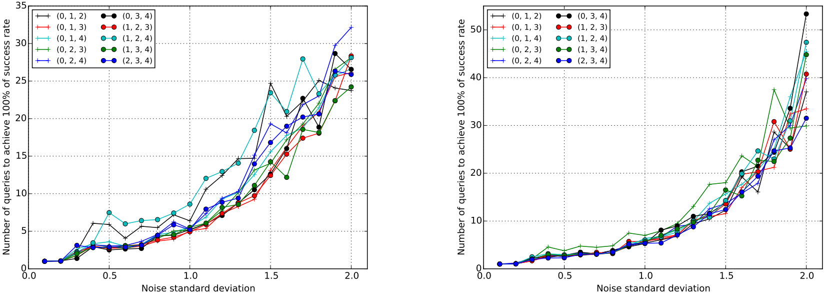


Figure: Number of observations to achieve a success rate of 100%wrt noise standard deviation for two different sets of public points.

Test of template attacks against a $(5, 2)$-SSS $(Z_0, Z_1, \ldots, Z_4)$ of Z

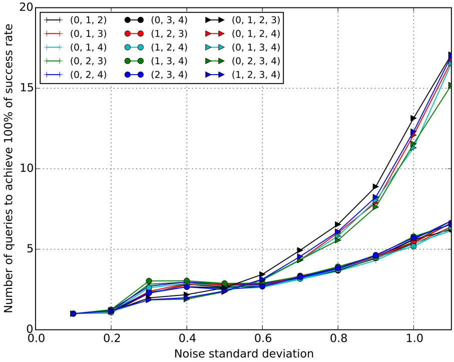


Figure: For different choices of tuples of shares, the number of observations required to achieve a 100% success rate vs the standard deviation of the noise.

Experiments Conclusions

- Observation 1: the choice of the public points impacts the attack efficiency!

Experiments Conclusions

- Observation 1: the choice of the public points impacts the attack efficiency!
- Observation 2: for some SNR, it is better to target strictly more than the sufficient number of shares needed to recover Z!

Experiments Conclusions

- Observation 1: the choice of the public points impacts the attack efficiency!
- Observation 2: for some SNR, it is better to target strictly more than the sufficient number of shares needed to recover Z!
- Rest of this talk: explain this phenomenon.

- Actually, we have to change the question:
  - ~~how many shares do I need to rebuild Z?~~
  - how much information do I need to rebuild Z?

- Actually, we have to change the question:
  - ~~how many shares do I need to rebuild Z?~~
  - how much information do I need to rebuild Z?

Guruswami & Wootters's Result *GuruswamiWootters16*

The number of bits needed to recover $Z \in \mathrm{GF}(2^m)$ from its (n, d)-sharing can be **much lower** than $(d+1) \times m$!

- Actually, we have to change the question:
  - ~~how many shares do I need to rebuild Z~~?
  - how much information do I need to rebuild Z?

Guruswami & Wootters's Result *GuruswamiWootters16*

The number of bits needed to recover $Z \in \mathrm{GF}(2^m)$ from its (n, d)-sharing can be **much lower** than $(d+1) \times m$!

- Recall that Lagrange's formula needs exactly $(d+1) \times m$ bits (or equiv. $d+1$ shares Z_i).

- Actually, we have to change the question:
  - ~~how many shares do I need to rebuild Z?~~
  - how much information do I need to rebuild Z?

Guruswami & Wootters's Result *GuruswamiWootters16*

The number of bits needed to recover $Z \in \mathrm{GF}(2^m)$ from its (n, d)-sharing can be **much lower** than $(d+1) \times m$!

- Recall that Lagrange's formula needs exactly $(d+1) \times m$ bits (or equiv. $d+1$ shares Z_i).
- Example *GuruswamiWootters16*:
  - for some $(14, 9)$-SSS sharing
  - Z can be recovered with only 64 bits of information on the Z_i
  - instead of $80 = 10 \times 8$ bits (if 10 shares are targeted)

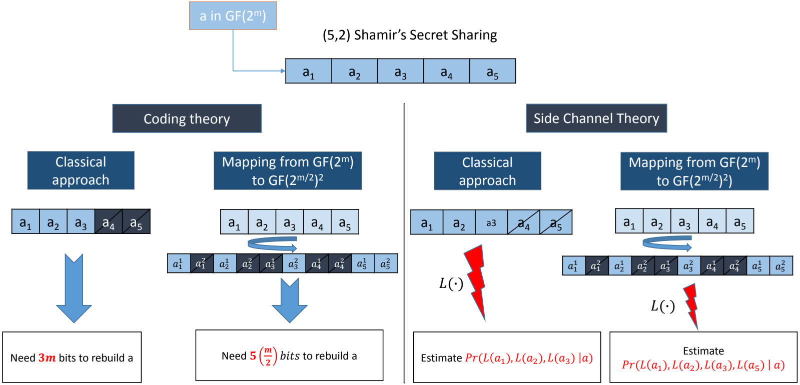


Figure: Side-channel and linear repairing codes for Shamir's sharing.

Z shared into $(Z_1, ..., Z_n)$ s.t. $Z_i = P_Z(\alpha_i)$ and $Z = P_Z(0)$.

$$Z = \sum_{i=1}^{n} \beta_i \times Z_i =$$

Z shared into $(Z_1, ..., Z_n)$ s.t. $Z_i = P_Z(\alpha_i)$ and $Z = P_Z(0)$.

$$Z = \sum_{i=1}^{n} \beta_i \times Z_i = \begin{cases} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_1 \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_1 \times \beta_i \times Z_i) \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_2 \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_2 \times \beta_i \times Z_i) \\ \vdots \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_t \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_t \times \beta_i \times Z_i) \end{cases}$$

Z shared into $(Z_1, \ldots, Z_n)$ s.t. $Z_i = P_Z(\alpha_i)$ and $Z = P_Z(0)$.

$$Z = \sum_{i=1}^{n} \beta_i \times Z_i = \begin{cases} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_1 \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_1 \times \beta_i \times Z_i) \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_2 \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_2 \times \beta_i \times Z_i) \\ \vdots \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_t \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(\mu_t \times \beta_i \times Z_i) \end{cases}$$

- Main Idea in *GuruswamiWootters16*: change the projections and, for each coordinate, **interpolate** $p_j(X) \times P_Z(X)$ **instead of** $P_Z(X)$ for well chosen polynomials $p_j(X)$.

Z shared into $(Z_1, ..., Z_n)$ s.t. $Z_i = P_Z(\alpha_i)$ and $Z = P_Z(0)$.

$$Z = \sum_{i=1}^{n} \beta_i \times Z_i = \begin{cases} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_1(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_1(\alpha_i) \times \beta_i \times Z_i) \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_2(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_2(\alpha_i) \times \beta_i \times Z_i) \\ \vdots \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_t(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_t(\alpha_i) \times \beta_i \times Z_i) \end{cases}$$

- Main Idea in *GuruswamiWootters16*: change the projections and, for each coordinate, **interpolate** $p_j(X) \times P_Z(X)$ **instead of** $P_Z(X)$ for well chosen polynomials $p_j(X)$.

Z shared into $(Z_1, \dots, Z_n)$ s.t. $Z_i = P_Z(\alpha_i)$ and $Z = P_Z(0)$.

$$Z = \sum_{i=1}^{n} \beta_i \times Z_i = \begin{cases} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_1(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_1(\alpha_i) \times \beta_i \times Z_i) \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_2(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_2(\alpha_i) \times \beta_i \times Z_i) \\ \vdots \\ \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_t(0) \times Z) = \sum_{i=1}^{n} \mathrm{tr}_{\mathbb{K}/\mathbb{F}}(p_t(\alpha_i) \times \beta_i \times Z_i) \end{cases}$$

- Main Idea in *GuruswamiWootters16*: change the projections and, for each coordinate, **interpolate $p_j(X) \times P_Z(X)$ instead of $P_Z(X)$** for well chosen polynomials $p_j(X)$.
- Necessary Condition: $p_1(0)$, $p_2(0)$, ..., $p_t(0)$ spans vector space of dimension t.

- Illustration for $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$

- Illustration for $n = 14$, $d = 9$, $GF(2^m) = GF(256)$ and $t = 2$
- Values obtained for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$p_1(\alpha_i)$	0	0	76	68	0	238	57	157	220	80	115	204	131
$p_2(\alpha_i)$	248	21	120	0	127	0	211	56	0	171	33	147	45

- Illustration for $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values obtained for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$p_1(\alpha_i)$	0	0	76	68	0	238	57	157	220	80	115	204	131
$p_2(\alpha_i)$	248	21	120	0	127	0	211	56	0	171	33	147	45

- in **Grey**, values linearly dependent over $\mathrm{GF}(16)$

- Illustration for $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values obtained for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$p_1(\alpha_i)$	0	0	76	68	0	238	57	157	220	80	115	204	131
$p_2(\alpha_i)$	248	21	120	0	127	0	211	56	0	171	33	147	45

- in **Grey**, values linearly dependent over GF(16)
- Total number of required bits on the shares: $\mathbf{64} = 16 * 4$ bits

- Illustration for $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values obtained for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$p_1(\alpha_i)$	0	0	76	68	0	238	57	157	220	80	115	204	131
$p_2(\alpha_i)$	248	21	120	0	127	0	211	56	0	171	33	147	45

- in **Grey**, values linearly dependent over GF(16)
- Total number of required bits on the shares: $\mathbf{64} = 16 * 4$ bits
- For Lagrange's interpolation formula: $\mathbf{80} = 10 * 8$ bits

- Illustration for $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values obtained for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$p_1(\alpha_i)$	0	0	76	68	0	238	57	157	220	80	115	204	131
$p_2(\alpha_i)$	248	21	120	0	127	0	211	56	0	171	33	147	45

- in **Grey**, values linearly dependent over GF(16)
- Total number of required bits on the shares: **64** $= 16 * 4$ bits
- For Lagrange's interpolation formula: **80** $= 10 * 8$ bits
- Conclusion: more shares are needed (10 instead of 8) but less information is needed (64 bits instead of 80 bits)

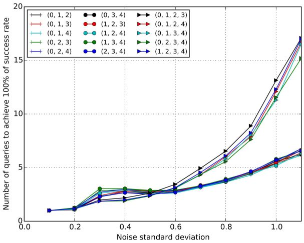


Figure: # of observations to achieve a 100% success rate vs the noise std.

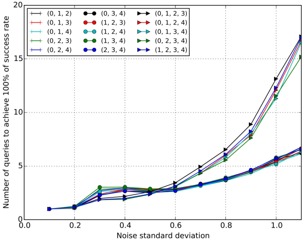


Figure: # of observations to achieve a 100% success rate vs the noise std.

- Theoretically: full knowledge of 3 shares (i.e. 24 bits) is enough to rebuild Z

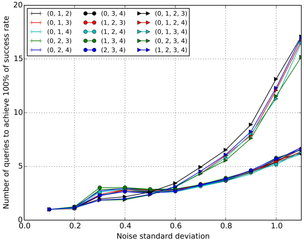


Figure: # of observations to achieve a 100% success rate vs the noise std.

- Theoretically: full knowledge of 3 shares (i.e. 24 bits) is enough to rebuild Z
- In practice: some 4-tuple of shares leeds to recover Z more efficiently than with 3 shares

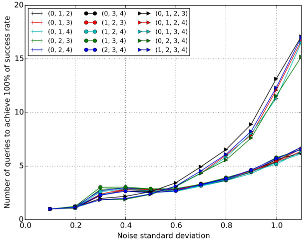


Figure: # of observations to achieve a 100% success rate vs the noise std.

- Theoretically: full knowledge of 3 shares (i.e. 24 bits) is enough to rebuild Z
- In practice: some 4-tuple of shares leeds to recover Z more efficiently than with 3 shares
- Explanation: from those 4 shares, the attack needs to recover strictly less than 24 bits

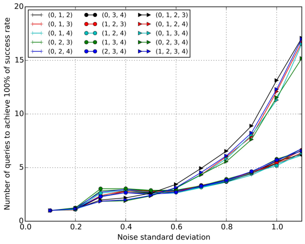


Figure: # of observations to achieve a 100% success rate vs the noise std.

- Theoretically: full knowledge of 3 shares (i.e. 24 bits) is enough to rebuild Z
- In practice: some 4-tuple of shares leeds to recover Z more efficiently than with 3 shares
- Explanation: from those 4 shares, the attack needs to recover strictly less than 24 bits
- Only effective till' some noise amount!

- $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$

- $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values of the reconstruction coefs for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$\mu_{i,1}$	0	0	76	68	0	238	57	157	220	80	115	204	131
$\mu_{i,2}$	248	21	120	0	127	0	211	56	0	171	33	147	45

- $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values of the reconstruction coefs for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$\mu_{i,1}$	0	0	76	68	0	238	57	157	220	80	115	204	131
$\mu_{i,2}$	248	21	120	0	127	0	211	56	0	171	33	147	45

- To enable reconstruction, only 64 bits are required instead of 80 (in state of the art)

- $n = 14$, $d = 9$, $\mathrm{GF}(2^m) = \mathrm{GF}(256)$ and $t = 2$
- Values of the reconstruction coefs for some polynomials $p_1(X)$ and $p_2(X)$ found by exhaustive search:

	1	2	3	4	5	6	7	8	9	10	11	12	13
$\mu_{i,1}$	0	0	76	68	0	238	57	157	220	80	115	204	131
$\mu_{i,2}$	248	21	120	0	127	0	211	56	0	171	33	147	45

- To enable reconstruction, only 64 bits are required instead of 80 (in state of the art)
- In the paper, we combine this property with *GoubinMartinelli11* and *CastagnosRennerZémor13* to improve the efficiency of the secure multiplication over data shared with SSS *Ben-OrGoldwasserWigderson88*.

Conclusions

Conclusions

- Shamir's Sharing Scheme is interesting to get implementations secure against HoSCA in the presence of glitches

Conclusions

- Shamir's Sharing Scheme is interesting to get implementations secure against HoSCA in the presence of glitches
- Because of the algebraic complexity of the sharing (polynomial evaluation/interpolation), the relation between the shares and the shared datum is difficult to analyze

Conclusions

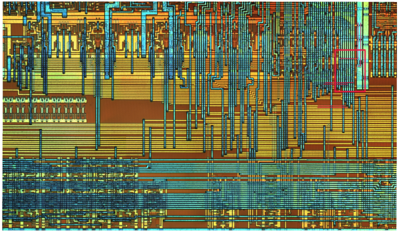
- Shamir's Sharing Scheme is interesting to get implementations secure against HoSCA in the presence of glitches
- Because of the algebraic complexity of the sharing (polynomial evaluation/interpolation), the relation between the shares and the shared datum is difficult to analyze
- We confirmed previous observations and exhibited new ones related to the difference with Boolean Sharing:
  - the choice of the public points matters from a security point of view
  - it can be sound to target more shares than strictly necessary
  - it exists more efficient reconstruction schemes than Lagrange's interpolation *GuruswamiWootters16*

Conclusions

- Shamir's Sharing Scheme is interesting to get implementations secure against HoSCA in the presence of glitches
- Because of the algebraic complexity of the sharing (polynomial evaluation/interpolation), the relation between the shares and the shared datum is difficult to analyze
- We confirmed previous observations and exhibited new ones related to the difference with Boolean Sharing:
  - the choice of the public points matters from a security point of view
  - it can be sound to target more shares than strictly necessary
  - it exists more efficient reconstruction schemes than Lagrange's interpolation *GuruswamiWootters16*
- We used the theory of Linear Exact Repairing Schemes (LERS) to improve the secure multiplication between data shared with SSS

Conclusions

- Shamir's Sharing Scheme is interesting to get implementations secure against HoSCA in the presence of glitches
- Because of the algebraic complexity of the sharing (polynomial evaluation/interpolation), the relation between the shares and the shared datum is difficult to analyze
- We confirmed previous observations and exhibited new ones related to the difference with Boolean Sharing:
  - the choice of the public points matters from a security point of view
  - it can be sound to target more shares than strictly necessary
  - it exists more efficient reconstruction schemes than Lagrange's interpolation *GuruswamiWootters16*
- We used the theory of Linear Exact Repairing Schemes (LERS) to improve the secure multiplication between data shared with SSS
- More works needed to study how to design efficient LERS for given n and d



Thank you for your attention!
Questions/Remarks?