Linear Secret Sharing Schemes from Error Correcting Codes and Universal Hash Functions

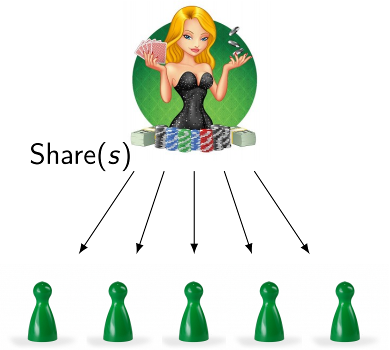
Ronald Cramer[1,2] Ivan Damgård[3] **Nico Döttling**[3]
Serge Fehr[1] Gabriele Spini[1,2,4]

[1]CWI Amsterdam [2]Mathematical Institute, Leiden University
[3]Aarhus University [4]Institut de Mathématiques de Bordeaux

Eurocrypt'15, April 28, 2015

Secret Sharing

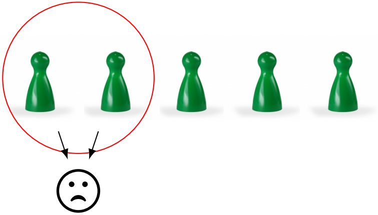
Distribute a secret s to players $P_1, \ldots, P_n$ such that



Secret Sharing

Distribute a secret s to players $P_1, \ldots, P_n$ such that

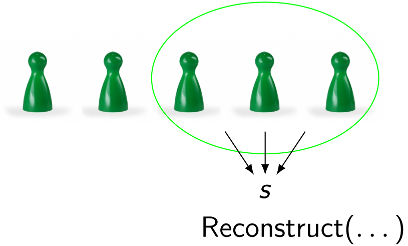
- t-**Privacy:** Any set of t players has no information about s.



Secret Sharing

Distribute a secret s to players $P_1, \ldots, P_n$ such that

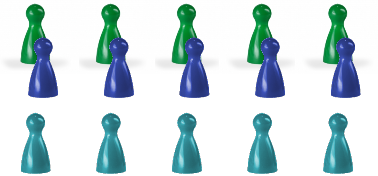
- t-**Privacy:** Any set of t players has no information about s.
- r-**Reconstruction:** Any set of r players can (efficiently) reconstruct s.



Secret Sharing

Distribute a secret s to players $P_1, \ldots, P_n$ such that

- t-**Privacy:** Any set of t players has no information about s.
- r-**Reconstruction:** Any set of r players can (efficiently) reconstruct s.
- **Linearity (LSSS):** $(c_1)_i$ and $(d_1)_i$ shares of s_1 and $s_2 \Rightarrow (\alpha c_i + \beta d_i)_i$ shares of $\alpha s_1 + \beta s_2$.



Design Goals for Secret Sharing

- **Structural Goals:**
  - Minimize size of shares (for given privacy and reconstruction thresholds)
  - Maximize the size of the secret (...)

Design Goals for Secret Sharing

- **Structural Goals:**
  - Minimize size of shares (for given privacy and reconstruction thresholds)
  - Maximize the size of the secret (...)
  - Additional algebraic/combinatorial properties (multiplicativity, sophisticated access structures,...)

Design Goals for Secret Sharing

- **Structural Goals:**
  - Minimize size of shares (for given privacy and reconstruction thresholds)
  - Maximize the size of the secret (...)
  - Additional algebraic/combinatorial properties (multiplicativity, sophisticated access structures,...)
- **Algorithmic Goals:** Optimize overhead of sharing and reconstruction/fancier reconstruction goals

Design Goals for Secret Sharing

- **Structural Goals:**
  - Minimize size of shares (for given privacy and reconstruction thresholds)
  - Maximize the size of the secret (...)
  - Additional algebraic/combinatorial properties (multiplicativity, sophisticated access structures,...)
- **Algorithmic Goals:** Optimize overhead of sharing and reconstruction/fancier reconstruction goals

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption
- Non-malleable codes
- ...

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption
- Non-malleable codes
- ...

LSSS construction paradigms

- Polynomials

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption
- Non-malleable codes
- ...

LSSS construction paradigms

- Polynomials
- Algebraic function fields

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption
- Non-malleable codes
- ...

LSSS construction paradigms

- Polynomials
- Algebraic function fields
- Random linear codes

Secret Sharing Schemes

Useful for:

- MPC
- 2PC via MPC-in-the-head
- Attribute based encryption
- Non-malleable codes
- ...

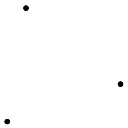
LSSS construction paradigms

- Polynomials
- Algebraic function fields
- Random linear codes

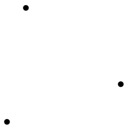
- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$

Error Correcting Codes

- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow$ $\mathrm{dist}(ECC(m_1), ECC(m_2)) > d$

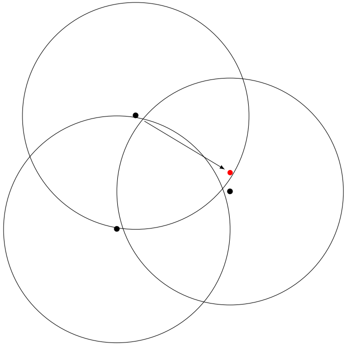


- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow$ $\text{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures



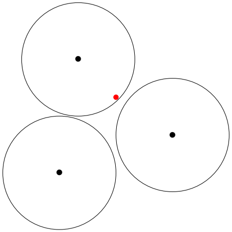
Error Correcting Codes

- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \rightarrow \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow$ $\text{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures



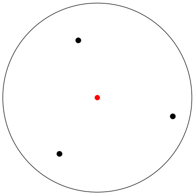
Error Correcting Codes

- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow$ $\mathrm{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures



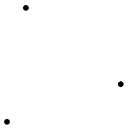
Error Correcting Codes

- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow \mathrm{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures



Error Correcting Codes

- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \to \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow$ $\text{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures
- Linear codes: ECC is $\mathbb{F}$-linear



- Redundant (distance-amplifying) encodings $ECC : \mathbb{F}^k \rightarrow \mathbb{F}^n$
- $m_1 \neq m_2 \Rightarrow \mathrm{dist}(ECC(m_1), ECC(m_2)) > d$
- This allows for correcting errors/erasures
- Linear codes: ECC is $\mathbb{F}$-linear
- **Algorithmic Goals:** Efficient encoding, decoding from errors/erasures, list-decoding.

LSSS and Codes (1)

- LSSS are *codes with privacy*
- Natural approach: Construct LSSS from codes [Mas96,CCGHV07,...]

LSSS and Codes (1)

- LSSS are *codes with privacy*
- Natural approach: Construct LSSS from codes [Mas96,CCGHV07,...]
- Privacy of LSSS established using properties of **specific** codes (e.g. dual distance)

LSSS and Codes (1)

- LSSS are *codes with privacy*
- Natural approach: Construct LSSS from codes [Mas96,CCGHV07,...]
- Privacy of LSSS established using properties of **specific** codes (e.g. dual distance)
- Properties irrelevant or detrimental for error correction

LSSS and Codes (1)

- LSSS are *codes with privacy*
- Natural approach: Construct LSSS from codes [Mas96,CCGHV07,...]
- Privacy of LSSS established using properties of **specific** codes (e.g. dual distance)
- Properties irrelevant or detrimental for error correction

LSSS and Codes (2)

- Coding theory made huge algorithmic progress the last $\approx 20y$.
- Linear time encoding, error/erasure correction [SS96,Spie96,GI04,...]

LSSS and Codes (2)

- Coding theory made huge algorithmic progress the last $\approx 20y$.
- Linear time encoding, error/erasure correction [SS96,Spie96,GI04,...]
- Efficient list decoding approaching the Singleton bound [GR07,GX13]

LSSS and Codes (2)

- Coding theory made huge algorithmic progress the last $\approx 20y$.
- Linear time encoding, error/erasure correction [SS96,Spie96,GI04,...]
- Efficient list decoding approaching the Singleton bound [GR07,GX13]
- Can we translate (algorithmic) progress in coding theory into realm of secret sharing?

LSSS and Codes (2)

- Coding theory made huge algorithmic progress the last $\approx 20y$.
- Linear time encoding, error/erasure correction [SS96,Spie96,GI04,...]
- Efficient list decoding approaching the Singleton bound [GR07,GX13]
- Can we translate (algorithmic) progress in coding theory into realm of secret sharing?
- ...for any code?

LSSS and Codes (2)

- Coding theory made huge algorithmic progress the last $\approx 20y$.
- Linear time encoding, error/erasure correction [SS96,Spie96,GI04,...]
- Efficient list decoding approaching the Singleton bound [GR07,GX13]
- Can we translate (algorithmic) progress in coding theory into realm of secret sharing?
- ...for any code?

Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**

Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**
- Construction preserves algorithmic efficiency

Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**
- Construction preserves algorithmic efficiency

FINEPRINT:

- * Ramp scheme: gap between privacy and reconstruction, large number of players, secret larger than shares

Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**
- Construction preserves algorithmic efficiency

FINEPRINT:

- * Ramp scheme: gap between privacy and reconstruction, large number of players, secret larger than shares
- ** For sufficiently large alphabet/shares (still constant)

NEW! Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**
- Construction preserves algorithmic efficiency

FINEPRINT:

- * Ramp scheme: gap between privacy and reconstruction, large number of players, secret larger than shares
- ** For sufficiently large alphabet/shares (still constant)
- Multiplicativity of a code not preserved by our constructions: No multiplicative SSS

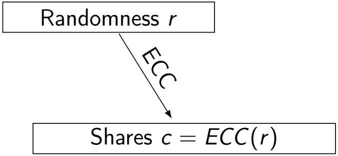
Our Results

- Black box construction of LSSS from any family of linear codes*
- Privacy close to best possible**
- Construction preserves algorithmic efficiency

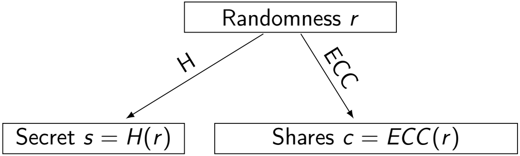
FINEPRINT:

- * Ramp scheme: gap between privacy and reconstruction, large number of players, secret larger than shares
- ** For sufficiently large alphabet/shares (still constant)
- **Multiplicativity of a code not preserved by our constructions**: No multiplicative SSS

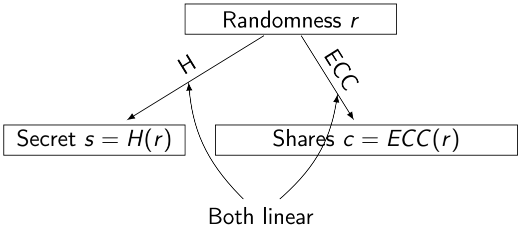
An information-theoretic look on secret sharing



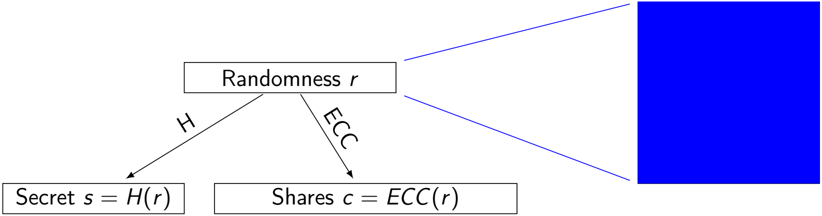
An information-theoretic look on secret sharing



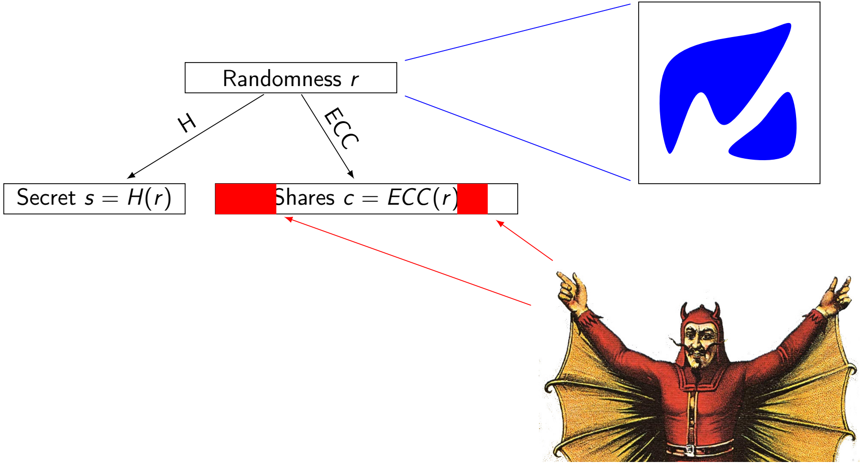
An information-theoretic look on secret sharing



An information-theoretic look on secret sharing

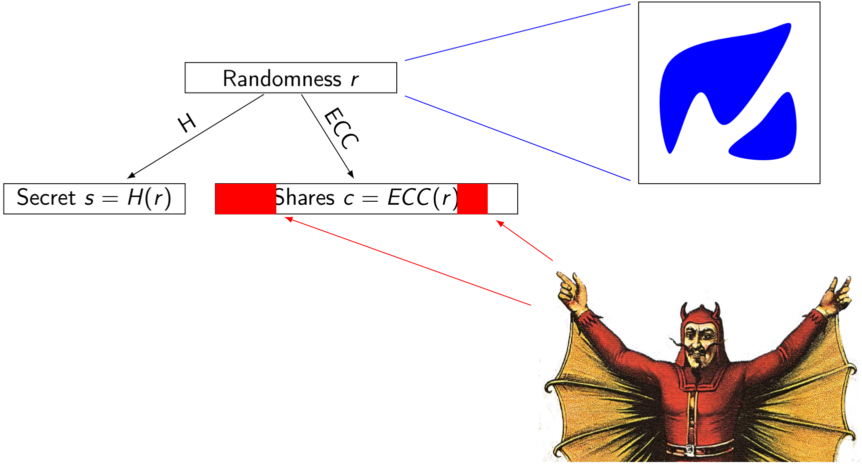


An information-theoretic look on secret sharing



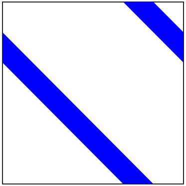
An information-theoretic look on secret sharing

Too Pessimistic!



An information-theoretic look on secret sharing

What really happens:



Randomness r

H ECC

Secret $s = H(r)$

Shares $c = ECC(r)$

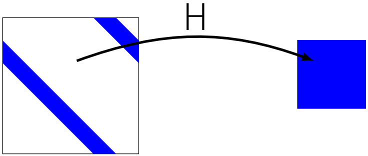
Both linear

What do we require from H?

- H should be linear (we want linear SSS)
- H should be surjective on all affine subspaces that correspond to sets in adversarial structure

What do we require from H?

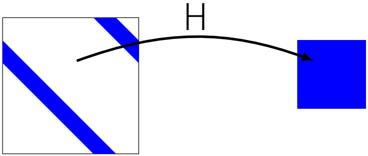
- H should be linear (we want linear SSS)
- H should be surjective on all affine subspaces that correspond to sets in adversarial structure



- How do we construct such a H for arbitrary codes ECC?

What do we require from H?

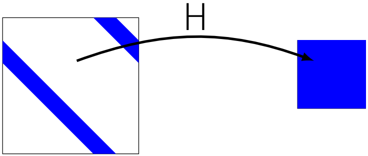
- H should be linear (we want linear SSS)
- H should be surjective on all affine subspaces that correspond to sets in adversarial structure



- How do we construct such a H for arbitrary codes ECC?
- Choose H at random from a small family

What do we require from H?

- H should be linear (we want linear SSS)
- H should be surjective on all affine subspaces that correspond to sets in adversarial structure



- How do we construct such a H for arbitrary codes ECC?
- **Choose H at random from a small family**

Universal Hash Functions

- Function Ensembles with statistical collision resistance
- $\mathcal{H} : \{H : X \to Y\}$ is universal iff $\forall x_1 \neq x_2$: $\Pr_{H \leftarrow_{\$} \mathcal{H}}[H(x_1) = H(x_2)] \leq 1/|Y|$

Subspace Surjectivity

Almost all functions of a family will be surjective on a fix subspace of some minimum dimension.

- $\mathcal{H}$ a family of **linear** universal hash functions $\mathbb{F}_q^k \to \mathbb{F}_q^l$
- V a subspace of $\mathbb{F}_q^k$ of dimension at least r

Subspace Surjectivity

Almost all functions of a family will be surjective on a fix subspace of some minimum dimension.

- $\mathcal{H}$ a family of **linear** universal hash functions $\mathbb{F}_q^k \to \mathbb{F}_q^l$
- V a subspace of $\mathbb{F}_q^k$ of dimension at least r
- $H \leftarrow_{\$} \mathcal{H}$ chosen uniformly at random

Subspace Surjectivity

Almost all functions of a family will be surjective on a fix subspace of some minimum dimension.

- $\mathcal{H}$ a family of **linear** universal hash functions $\mathbb{F}_q^k \to \mathbb{F}_q^l$
- V a subspace of $\mathbb{F}_q^k$ of dimension at least r
- $H \leftarrow_{\$} \mathcal{H}$ chosen uniformly at random
- Then $H(V) = \mathbb{F}_q^l$ (i.e. H is surjective on V), except with probability $q^{-(r-l)}$ over the choice of H

Subspace Surjectivity

Almost all functions of a family will be surjective on a fix subspace of some minimum dimension.

- $\mathcal{H}$ a family of **linear** universal hash functions $\mathbb{F}_q^k \to \mathbb{F}_q^l$
- V a subspace of $\mathbb{F}_q^k$ of dimension at least r
- $H \leftarrow_{\$} \mathcal{H}$ chosen uniformly at random
- Then $H(V) = \mathbb{F}_q^l$ (i.e. H is surjective on V), except with probability $q^{-(r-l)}$ over the choice of H

The Scheme

LSSS

Fix some linear hash function $H \in \mathcal{H}$, ECC linear code

Share(s):
- $r \leftarrow_{\$} H^{-1}(s)$
- $\mathbf{c} \leftarrow ECC(r)$
- Output share vector $\mathbf{c}$

Reconstruct($\tilde{c}$):
- $r \leftarrow ECC$.Decode($\tilde{c}$)
- If $r = \perp$
  - Output $\perp$
- $s \leftarrow H(r)$
- Output secret s

Main Theorem

Theorem

- *ECC an $\mathbb{F}_q$-linear code of length n, rate R and alphabet $\mathbb{F}_q^m$*
- *$\mathcal{H}$ a family of $\mathbb{F}_q$-linear universal hash functions $\mathbb{F}_q^{Rn} \to \mathbb{F}_q^{\rho n}$*

Main Theorem

Theorem

- *ECC an $\mathbb{F}_q$-linear code of length n, rate R and alphabet $\mathbb{F}_q^m$*
- *$\mathcal{H}$ a family of $\mathbb{F}_q$-linear universal hash functions $\mathbb{F}_q^{Rn} \to \mathbb{F}_q^{\rho n}$*
- *Constant $\eta > 0$*

Main Theorem

Theorem

- *ECC an $\mathbb{F}_q$-linear code of length n, rate R and alphabet $\mathbb{F}_q^m$*
- *$\mathcal{H}$ a family of $\mathbb{F}_q$-linear universal hash functions $\mathbb{F}_q^{Rn} \to \mathbb{F}_q^{\rho n}$*
- *Constant $\eta > 0$*
- *There exists a $H \in \mathcal{H}$ such that above scheme has τn-**privacy**., given that*

$$R \geq \rho + \eta + \tau + h(\tau)/(m \cdot \log(q)).$$ [a]

Main Theorem

Theorem

- *ECC an $\mathbb{F}_q$-linear code of length n, rate R and alphabet $\mathbb{F}_q^m$*
- *$\mathcal{H}$ a family of $\mathbb{F}_q$-linear universal hash functions $\mathbb{F}_q^{Rn} \to \mathbb{F}_q^{\rho n}$*
- *Constant $\eta > 0$*
- *There exists a $H \in \mathcal{H}$ such that above scheme has τn-**privacy.**, given that*

$$R \geq \rho + \eta + \tau + h(\tau)/(m \cdot \log(q)).^a$$

- *H can be chosen randomly with success-probability $1 - q^{-\eta nm}$.*

[a] $h(p) = -p\log(p) - (1-p)\log(1-p)$

Main Theorem

Theorem

- *ECC an $\mathbb{F}_q$-linear code of length n, rate R and alphabet $\mathbb{F}_q^m$*
- *$\mathcal{H}$ a family of $\mathbb{F}_q$-linear universal hash functions $\mathbb{F}_q^{Rn} \to \mathbb{F}_q^{\rho n}$*
- *Constant $\eta > 0$*
- *There exists a $H \in \mathcal{H}$ such that above scheme has τn-**privacy.**, given that*

$$R \geq \rho + \eta + \tau + h(\tau)/(m \cdot \log(q)).^a$$

- *H can be chosen randomly with success-probability $1 - q^{-\eta nm}$.*

[a] $h(p) = -p\log(p) - (1-p)\log(1-p)$

Applications

1. LSSS with linear time sharing and reconstruction.
2. Robust secret sharing

Applications

1. LSSS with linear time sharing and reconstruction.
2. Robust secret sharing

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function
- But: Can share random secrets without inverting hash function

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function
- But: Can share random secrets without inverting hash function
- Bootstrap this into a standard sharing algorithm via OTP encryption + dispersion (similar to [Kra93])

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function
- But: Can share random secrets without inverting hash function
- Bootstrap this into a standard sharing algorithm via OTP encryption + dispersion (similar to [Kra93])
- **Application of the Application:** (Amortized) Linear time commitments via MPC-in-the-head.

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function
- But: Can share random secrets without inverting hash function
- Bootstrap this into a standard sharing algorithm via OTP encryption + dispersion (similar to [Kra93])
- **Application of the Application:** (Amortized) Linear time commitments via MPC-in-the-head.

Linear Time Secret Sharing

- Use linear time en- and decodable codes [Spi96,GI04] + linear time computable hash functions [IKOS07,DI14]
- **Small annoyance:** Sharing algorithm inverts hash function
- But: Can share random secrets without inverting hash function
- Bootstrap this into a standard sharing algorithm via OTP encryption + dispersion (similar to [Kra93])
- **Application of the Application:** (Amortized) Linear time commitments via MPC-in-the-head.

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction
- Robust reconstruction easy for $t \leq n/3$ and impossible for $t \geq n/2$. **For** $n/3 \leq t < n/2$ **only statistical robustness.**

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction
- Robust reconstruction easy for $t \leq n/3$ and impossible for $t \geq n/2$. **For $n/3 \leq t < n/2$ only statistical robustness.**
- Efficient reconstruction!

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction
- Robust reconstruction easy for $t \leq n/3$ and impossible for $t \geq n/2$. **For $n/3 \leq t < n/2$ only statistical robustness.**
- Efficient reconstruction!
- Best construction so far ([CFOR12]): $t \leq n/2 - 1$, shares of size $O(\lambda + n \cdot \log(n))$

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction
- Robust reconstruction easy for $t \leq n/3$ and impossible for $t \geq n/2$. **For** $n/3 \leq t < n/2$ **only statistical robustness.**
- Efficient reconstruction!
- Best construction so far ([CFOR12]): $t \leq n/2 - 1$, shares of size $O(\lambda + n \cdot \log(n))$
- **This work:** $t < (1 - \epsilon)n/2$ for any constant ϵ, shares of size $O(1 + \lambda/n)$

Robust Secret Sharing

- Stronger reconstruction property: Adversary returns corrupted shares.
- Standard reconstruction $\sim$ erasure correction
- Robust reconstruction $\sim$ error correction
- Robust reconstruction easy for $t \leq n/3$ and impossible for $t \geq n/2$. **For $n/3 \leq t < n/2$ only statistical robustness.**
- Efficient reconstruction!
- Best construction so far ([CFOR12]): $t \leq n/2 - 1$, shares of size $O(\lambda + n \cdot \log(n))$
- **This work:** $t < (1 - \epsilon)n/2$ for any constant ϵ, shares of size $O(1 + \lambda/n)$

Robust Secret Sharing

Idea

- **List Decodable Code $\Rightarrow$ List Decodable SSS**
- List Decodable SSS + AMD Codes $\Rightarrow$ Robust Secret Sharing

Robust Secret Sharing

Idea

- **List Decodable Code $\Rightarrow$ List Decodable SSS**
- List Decodable SSS + AMD Codes $\Rightarrow$ Robust Secret Sharing

Thank You!