Correlated Pseudorandomness from the Hardness of Decoding Quasi-Abelian Codes

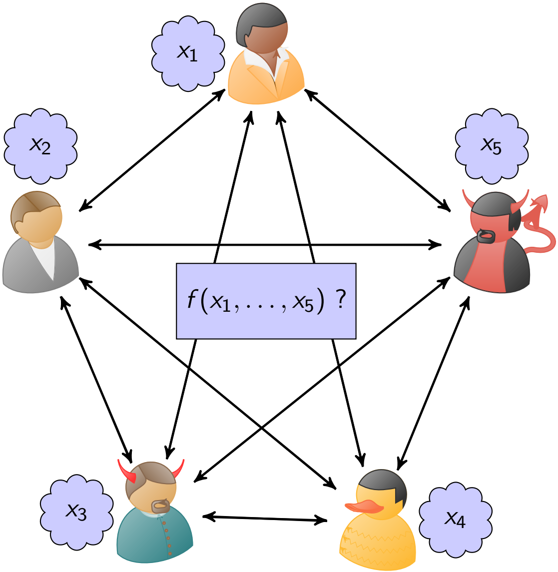
Maxime Bombar, Geoffroy Couteau, Alain Couvreur, Clément Ducros

CWI, Amsterdam

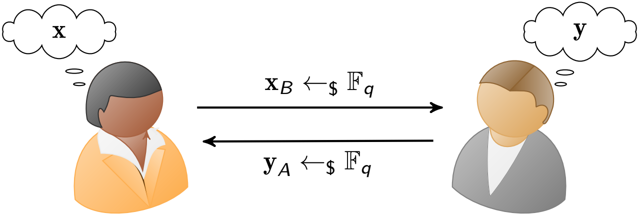
Journées C2, Najac

October, 16 2023

Secure Multiparty Computation



Additive Secret Sharing



$$\mathbf{x}_A \stackrel{\text{def}}{=} \mathbf{x} - \mathbf{x}_B \approx \$$$

$$\mathbf{y}_A$$

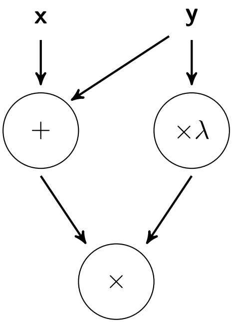
$$\mathbf{x}_B$$

$$\mathbf{y}_B \stackrel{\text{def}}{=} \mathbf{y} - \mathbf{y}_A \approx \$$$

$$\mathbf{x}_A + \mathbf{x}_B = \mathbf{x}$$
$$\mathbf{y}_A + \mathbf{y}_B = \mathbf{y}$$

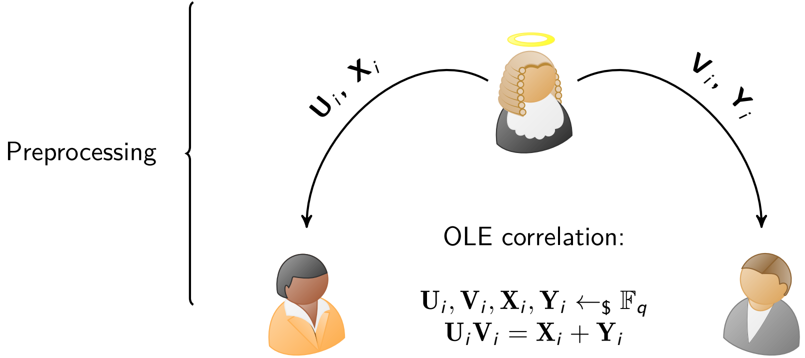
Secure multiparty Computation over $\mathbb{F}_q$

Goal: Compute some function $f(\mathbf{x}, \mathbf{y})$ without revealing $\mathbf{x}, \mathbf{y}$.
Idea: Compute $\textsc{Shares}(f(\mathbf{x}, \mathbf{y}))$ from $\textsc{Shares}(\mathbf{x}, \mathbf{y})$ and reveal at the end.



- $\textsc{Shares}(\mathbf{x} + \mathbf{y}) = \textsc{Shares}(\mathbf{x}) + \textsc{Shares}(\mathbf{y}) \Rightarrow$ free ✓
- $\textsc{Shares}(\lambda \mathbf{x}) = \lambda \textsc{Shares}(\mathbf{x}) \Rightarrow$ free ✓
- Multiplications $\Rightarrow$ Require communication $\Rightarrow$ Costly ✗.

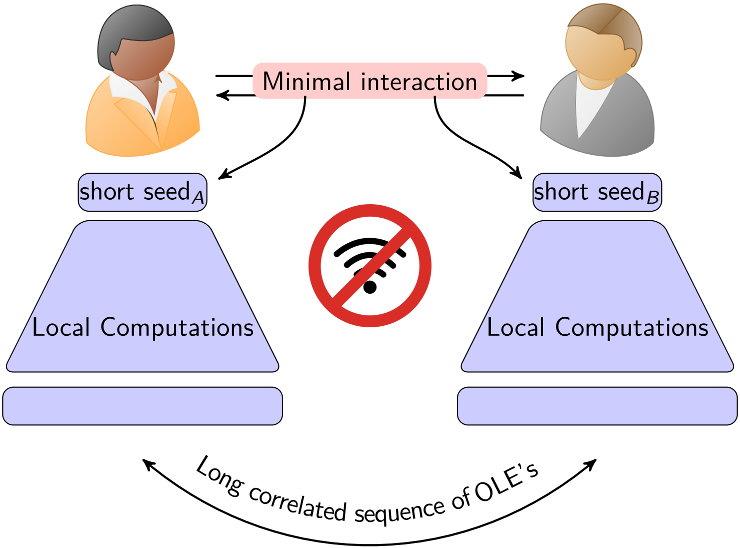
The Correlated Randomness Model



Fast online protocol consumming two OLE's per multiplication

How to efficiently distribute many ($\approx 2^{20}, 2^{30}$) OLE's?

Pseudorandom Correlation Generator (PCG)



One OLE to Rule them All

Goal: Distribute **a lot** of random OLE's over $\mathbb{F}_q$.

Wishful thinking. ([BCGIKS20][1]) Take a ring $\mathcal{R} \simeq \mathbb{F}_q \times \cdots \times \mathbb{F}_q$

ONE OLE over $\mathcal{R}$ $\quad$ $\mathbf{U} \cdot \mathbf{V} = \mathbf{X} + \mathbf{Y}$

↓

Many OLE over $\mathbb{F}_q$ $\quad$ $\mathbf{u}_i \cdot \mathbf{v}_i = \mathbf{x}_i + \mathbf{y}_i$

[1] *Efficient Pseudorandom Correlation Generators from Ring-LPN*, Boyle, Couteau, Gilboa, Ishai, Kohl, Sholl - CRYPTO '20

PCG for OLE [BCGIKS20]

There exists an efficient protocol to distribute additive shares of **sparse** vectors.[2]

Idea: Take $\mathcal{R} = \mathbb{F}_q[X]/(F(X))$ where $F(X)$ splits completely.

- Sample randomly $\mathbf{a} \leftarrow \mathcal{R}$.
- Set $\mathbf{U} \stackrel{\text{def}}{=} \mathbf{a} \cdot \mathbf{e}_1 + \mathbf{f}_1 \approx^? \$$
- Set $\mathbf{V} \stackrel{\text{def}}{=} \mathbf{a} \cdot \mathbf{e}_2 + \mathbf{f}_2 \approx^? \$$

Where $\mathbf{e}_i, \mathbf{f}_i$ are **sparse** polynomials.

$$\begin{aligned} \mathbf{U} \cdot \mathbf{V} &= \mathbf{a}^2(\mathbf{e}_1\mathbf{e}_2) + \mathbf{a}(\mathbf{e}_1\mathbf{f}_2 + \mathbf{e}_2\mathbf{f}_1) + \mathbf{f}_1\mathbf{f}_2 \\ &= \text{Linear combination of } \textit{somewhat} \text{ sparse polynomials.} \end{aligned}$$

[2] *Function secret sharing*, Boyle, Gilboa, Ishai - EUROCRYPT '15

PCG for OLE [BCGIKS20]

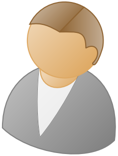
$\mathcal{R} = \mathbb{F}_q[X]/(F(X)) \simeq \mathbb{F}_q \times \cdots \mathbb{F}_q$

$$\mathbf{U} = \mathbf{a} \cdot \mathbf{e}_1 + \mathbf{f}_1$$
$$\mathbf{V} = \mathbf{a} \cdot \mathbf{e}_2 + \mathbf{f}_2$$

$\textsc{Seed}_A = (\mathbf{a}, \mathbf{e}_1, \mathbf{f}_1, \textsc{Shares}(\mathbf{e}_i \mathbf{f}_j))$

↓

Locally compute $\mathbf{U}, \textsc{Share}(\mathbf{UV})$
$\Rightarrow$ OLE's over $\mathbb{F}_q$ via CRT



$\textsc{Seed}_B = (\mathbf{a}, \mathbf{e}_2, \mathbf{f}_2, \textsc{Shares}(\mathbf{e}_i \mathbf{f}_j))$

↓

Locally Compute $\mathbf{V}, \textsc{Share}(\mathbf{UV})$
$\Rightarrow$ OLE's over $\mathbb{F}_q$ via CRT

PCG for OLE [BCGIKS20]

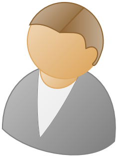
$\mathcal{R} = \mathbb{F}_q[X]/(F(X)) \simeq \mathbb{F}_q \times \cdots \mathbb{F}_q$ $\Rightarrow$ Only works for large q

$$\mathbf{U} = \mathbf{a} \cdot \mathbf{e}_1 + \mathbf{f}_1$$
$$\mathbf{V} = \mathbf{a} \cdot \mathbf{e}_2 + \mathbf{f}_2$$

$\textsc{Seed}_A = (\mathbf{a}, \mathbf{e}_1, \mathbf{f}_1, \textsc{Shares}(\mathbf{e}_i\mathbf{f}_j))$

$\downarrow$

Locally compute $\mathbf{U}, \textsc{Share}(\mathbf{UV})$
$\Rightarrow$ OLE's over $\mathbb{F}_q$ via CRT



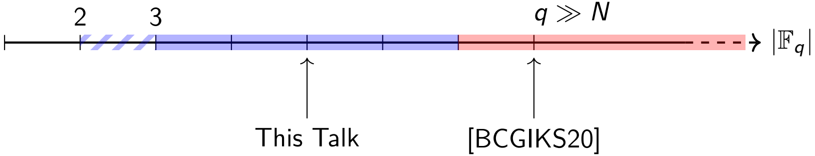
$\textsc{Seed}_B = (\mathbf{a}, \mathbf{e}_2, \mathbf{f}_2, \textsc{Shares}(\mathbf{e}_i\mathbf{f}_j))$

$\downarrow$

Locally Compute $\mathbf{V}, \textsc{Share}(\mathbf{UV})$
$\Rightarrow$ OLE's over $\mathbb{F}_q$ via CRT

This Talk

Goal: Produce N OLE's over $\mathbb{F}_q$.



Group algebras

Finite (abelian) group G, $\mathbb{F}_q[G] = \left\{\sum_{g\in G} a_g g \mid a_g \in \mathbb{F}_q\right\} \simeq \mathbb{F}_q^{|G|}$

$$\left(\sum_{g\in G} a_g g\right)\left(\sum_{g\in G} b_g g\right) \stackrel{\text{def}}{=} \sum_{g\in G}\left(\sum_{h\in G} a_h b_{h^{-1}g}\right) g.$$

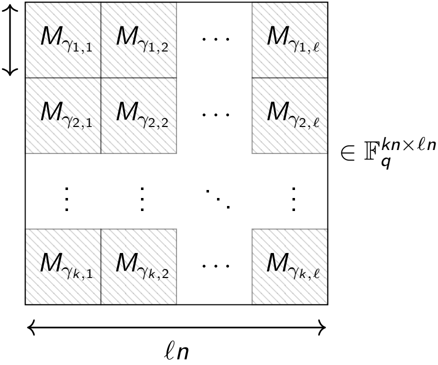
$G = \{1\}$	$\mathbb{F}_q[G] = \mathbb{F}_q,$
$G = \mathbb{Z}/N\mathbb{Z}$	$\mathbb{F}_q[G] = \mathbb{F}_q[X]/(X^N - 1),$
$G = \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/M\mathbb{Z}$	$\mathbb{F}_q[G] = \mathbb{F}_q[X, Y]/(X^N - 1, Y^M - 1).$

Quasi-abelian codes $\simeq$ Module lattices

A quasi-abelian code is an $\mathbb{F}_q[G]$-submodule of $\mathbb{F}_q[G]^\ell$

$$n \stackrel{\text{def}}{=} |G|.$$

$$\mathbf{\Gamma} = \begin{pmatrix} \gamma_{1,1} & \cdots & \gamma_{1,\ell} \\ \vdots & \ddots & \vdots \\ \gamma_{k,1} & \cdots & \gamma_{k,\ell} \end{pmatrix} \in \mathbb{F}_q[G]^{k\times\ell}$$



$$\mathcal{C} \stackrel{\text{def}}{=} \{\mathbf{m}\mathbf{\Gamma} \mid \mathbf{m} \in \mathbb{F}_q[G]^k\}.$$

Example: Quasi-cyclic codes

$G = \mathbb{Z}/n\mathbb{Z} \qquad \mathcal{R} = \mathbb{F}_q[G] \simeq \mathbb{F}_q[X]/(X^n - 1)$

$$\mathbf{a} \in \mathbb{F}_q[G] \longleftrightarrow \begin{pmatrix} a_0 & a_1 & \dots & \dots & a_{n-1} \\ a_{n-1} & a_0 & \dots & \dots & a_{n-2} \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_{n-1} & a_0 \end{pmatrix}$$

$$\mathbf{m} \begin{pmatrix} \mathbf{a}^{(1)} & \mathbf{a}^{(2)} \\ \circlearrowleft & \circlearrowleft \end{pmatrix} + \begin{pmatrix} \mathbf{e}^{(1)} & \mathbf{e}^{(2)} \end{pmatrix} \xrightarrow{\simeq} \begin{cases} \mathbf{m}(X)\mathbf{a}^{(1)}(X) + \mathbf{e}^{(1)}(X) \in \mathcal{R} \\ \mathbf{m}(X)\mathbf{a}^{(2)}(X) + \mathbf{e}^{(2)}(X) \in \mathcal{R} \end{cases}$$

Quasi-Abelian (Syndrome) Decoding

Search version

Data. Random $\mathbf{H} \leftarrow \mathbb{F}_q[G]^{(\ell-k)\times\ell}$, a target weight $t \leqslant n$ and $\mathbf{s} \in \mathbb{F}_q[G]^{\ell-k}$.

Goal. Find $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_\ell) \in \mathbb{F}_q[G]^\ell$ with $|\mathbf{e}_i| = t$ and $\mathbf{H}\mathbf{e}^\top = \mathbf{s}$.

Decision version

Data. Random $\mathbf{H} \leftarrow \mathbb{F}_q[G]^{(\ell-k)\times\ell}$, a target weight $t \leqslant n$ and $\mathbf{y} \in \mathbb{F}_q[G]^{\ell-k}$.

Question. Is $\mathbf{y}$ uniform or of the form $\mathbf{H}\mathbf{e}^\top$ with $|\mathbf{e}_i| = t$?

Hardness of decision version $\iff$ Pseudorandomness of $(\mathbf{H}, \mathbf{H}\mathbf{e}^\top)$.

Quasi-cyclic versions used in BIKE and HQC (NIST 4th round).

Security ?

Why should we believe in pseudorandomness of $(\mathbf{H}, \mathbf{H}\mathbf{e}^\top)$?

No decoding algorithm (50+ years of research)

But search-to-decision reduction only for particular cases ([**B**CD22][3]).

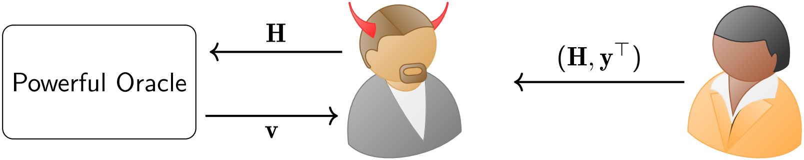
Roughly all known generic attacks[a] fit in the *linear tests* framework.

[a]Not grobner based

[3] *On Codes and Learning With Errors over Function Fields*, B., Couvreur, Debris-Alazard - CRYPTO '22.

The linear test framework

Essentially all known [4] distinguishers can be expressed as a *linear* function $\mathbf{v} \cdot \mathbf{y}^\top$.



$\mathbf{v} \cdot \mathbf{H}\mathbf{e}^\top = \langle \mathbf{v}\mathbf{H}, \mathbf{e} \rangle$ is biased towards 0 if $\mathbf{v}\mathbf{H}$ is *sparse*.

[4] Information Set Decoding, Statistical Decoding, folding ...

Security against linear attacks

No low-weight (non-zero) $\mathbf{vH} \Longleftrightarrow \mathcal{C}^{\perp}$ has good minimum distance

Gilbert-Varshamov bound [FL15][5]

Random QA codes have minimum distance linear in their length.

[5] *Thresholds of Random Quasi-Abelian Codes*, Fan, Lin - IEEE-IT

A multivariate setting

Goal. Find G such that $\mathbb{F}_q[G] \simeq \underbrace{\mathbb{F}_q \times \cdots \times \mathbb{F}_q}_{N \text{ copies}}$ with $N >> 1$.

Idea. Take $G = (\mathbb{Z}/(q-1)\mathbb{Z})^t$ for some $t \geqslant 1$.

$$\begin{aligned}
\mathbb{F}_q[G] = &\quad \mathbb{F}_q[X_1, \ldots, X_t]/(X_1^{q-1} - 1, \ldots, X_t^{q-1} - 1) \\
= &\quad \prod_{(\zeta_1,\ldots,\zeta_t)\in(\mathbb{F}_q^\times)^t} \mathbb{F}_q[X_1, \ldots, X_t]/(X_1 - \zeta_1, \ldots, X_t - \zeta_t) \\
= &\quad \underbrace{\mathbb{F}_q \times \cdots \times \mathbb{F}_q}_{(q-1)^t \text{ copies}}
\end{aligned}$$

With $q = 3$, choose $t = 20$ to get $N = 2^{20}$ OLE correlations over $\mathbb{F}_3$.

The curious case of $\mathbb{F}_2$

- Is it possible to go to $\mathbb{F}_2$?
- Obviously, we cannot set $q = 2$ in the above construction.
- Most natural approach would be using the ring of boolean functions

$$\mathcal{R} = \mathbb{F}_2[X_1, \ldots, X_t]/(X_1^2 - X_1, \ldots, X_t^2 - X_t).$$

⚠This is NOT a group algebra.

Vulnerable to a very simple linear attack.

The curious case of $\mathbb{F}_2$ (cont'd)

In fact we have the following theorem

There is no group G such that $\mathbb{F}_2[G] = \underbrace{\mathbb{F}_2 \times \cdots \times \mathbb{F}_2}_{N \text{ times}}$ unless $G = \{1\}$ and $N = 1$.

Proof. $G \subset \mathbb{F}_2[G]^\times$ and $|(\mathbb{F}_2 \times \cdots \times \mathbb{F}_2)^\times| = 1$.

Towards $\mathbb{F}_2$?

- There exists G and a ring $\mathcal{R}$ endowed with an action of G such that

$$\mathbb{F}_2[G] \underbrace{\simeq}_{\textit{As modules}} \mathcal{R} \underbrace{\simeq}_{\textit{As algebras}} \mathbb{F}_2 \times \cdots \times \mathbb{F}_2$$

- Construction based on number theory in function fields
- Needs more work on the MPC side....

Conclusion and perspectives

What I did not talk about

- Concrete security
- Practical parameters relevant for MPC
- From 2 to N party computation.
- Efficiency

Open questions:

- Are there other secure structured variants of the Decoding Problem ?
- Characterise secure instances ? (Uncertainty principle ?)
- Possibility to fix the protocol for $\mathbb{F}_2$?
- ...

;

Thank You!