KU LEUVEN

Overdrive$_{2^k}$: Efficient Secure MPC over $\mathbb{Z}_{2^k}$ from Somewhat Homomorphic Encryption

Emmanuela Orsini **Nigel Smart** Fredrik Vercauteren

imec-COSIC, KU Leuven

MPC setting in this work

Goal:
Multiparty computation for circuits over $\mathbb{Z}_{2^k}$

Adversary:
Dishonest majority, malicious adversary

- ⋆ Impossible without computational assumption
- ⋆ No guaranteed termination

MPC setting in this work

Goal:
Multiparty computation for circuits over $\mathbb{Z}_{2^k}$

Adversary:
Dishonest majority, malicious adversary

- ⋆ Impossible without computational assumption
- ⋆ No guaranteed termination

MPC setting in this work

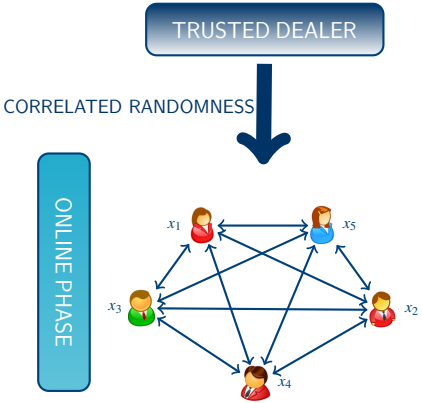
Goal:
Multiparty computation for circuits over $\mathbb{Z}_{2^k}$

Adversary:
Dishonest majority, malicious adversary

- ⋆ Impossible without computational assumption
- ⋆ No guaranteed termination

- Allows direct use of CPU's arithmetic
  - No need for reduction mod p
- Bitwise operations
  - Easier to do basic arithmetic for applications

MPC in the preprocessing model



Implementing the trusted dealer - Finite field

- SPDZ-like protocols ([DPSZ12] ([KPR18] **Overdrive-HighGear**)
- BeDOZa-like protocols ([BDOZ10] ([KPR18] **Overdrive-LowGear**)

- TinyOT-like protocols

Implementing the trusted dealer - Efficiency in the field case

- Large **computational** overhead
- Small **communication**
  - LowGear allows better SHE parameters
  - HighGear requires less communication
- Better for arithmetic modulo p

- Small **computational** overhead
- Requires pairwise **communication**
- Better for boolean and binary circuits

MPC over $\mathbb{Z}_{2^k}$

- `Cramer et al. EUROCRYPT 2003`: actively secure MPC over black-box rings
- `Bogdanov et al. ESORICS 2008`: (3,1) and passive security; `[AFLNO16], [FLNO17]`
- `Damgård et al. CRYPTO 2018`: compiler from passive to active with small # corruption

Homomorphic Encryption

Oblivious Transfer

- Cramer et al. [CDESX18] (**SPD**$\mathbb{Z}_{2^k}$)
- Damgård et al. [DEFKSV19]

MPC over $\mathbb{Z}_{2^k}$

- Cramer et al. EUROCRYPT 2003: actively secure MPC over black-box rings
- Bogdanov et al. ESORICS 2008: (3,1) and passive security; [AFLNO16], [FLN017]
- Damgård et al. CRYPTO 2018: compiler from passive to active with small # corruption

Homomorphic Encryption

- Catalano et al. [CDFG19] (**MON**$\mathbb{Z}_{2^k}$**A**)
  - Two party case only

Oblivious Transfer

- Cramer et al. [CDESX18] (**SPD**$\mathbb{Z}_{2^k}$)
- Damgård et al. [DEFKSV19]

MPC over $\mathbb{Z}_{2^k}$

- `Cramer et al. EUROCRYPT 2003`: actively secure MPC over black-box rings
- `Bogdanov et al. ESORICS 2008`: (3,1) and passive security; `[AFLNO16], [FLNO17]`
- `Damgård et al. CRYPTO 2018`: compiler from passive to active with small # corruption

Homomorphic Encryption

- Catalano et al. [CDFG19] (**MON**$\mathbb{Z}_{2^k}$**A**)
  - Two party case only
- **Our result – Overdrive**$_{2^k}$
  - Any number of parties

Oblivious Transfer

- Cramer et al. [CDESX18] (**SPD**$\mathbb{Z}_{2^k}$)
- Damgård et al. [DEFKSV19]

Our results

- We use BGV (Brakerski et al. 2011) to implement the SPD$\mathbb{Z}_{2^k}$ preprocessing
  1. We introduce a special packing technique for BGV operating over $\mathbb{Z}_{2^k}$
  2. Adapt the SPDZ preprocessing (distributed decryption and ZK) to work on $\mathbb{Z}_{2^k}$
- Introduce a new primitive for SPD$\mathbb{Z}_{2^k}$: bit-decomposition
  (this has also been independently developed in [DEFKSV19])

Our results

- We use BGV (Brakerski et al. 2011) to implement the SPD$\mathbb{Z}_{2^k}$ preprocessing
  1. We introduce a special packing technique for BGV operating over $\mathbb{Z}_{2^k}$
  2. Adapt the SPDZ preprocessing (distributed decryption and ZK) to work on $\mathbb{Z}_{2^k}$
- Introduce a new primitive for SPD$\mathbb{Z}_{2^k}$: bit-decomposition
  (this has also been independently developed in [DEFKSV19])

Batch computation (Traditional)

Use a ring defined by power-of-two cyclotomic $\Phi_m(X) = \Phi_{2^n}(X)$.

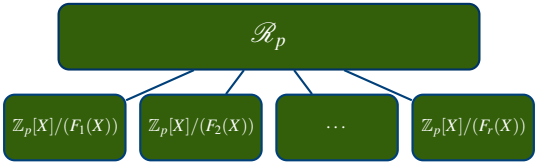
- $\mathscr{R} = \mathbb{Z}[X]/(\Phi_m(X))$, where $\deg(\Phi_m(X)) = \phi(m) = N$
- $\mathscr{R}_p = R/pR = \mathbb{Z}_p[X]/(\Phi_m(X))$, m and p coprime

$$\implies \quad \Phi_m(X) \equiv \prod_{i=1}^{r} F_i(X) \pmod{p}$$

- Each $F_i(X)$ has degree $d = \phi(m)/r = N/r$

$$\mathscr{R}_p \cong \mathbb{Z}_p[X]/(F_1(X)) \times \cdots \times \mathbb{Z}_p[X]/(F_r(X)) \cong \mathbb{F}_{p^d}^r.$$

Batch computation



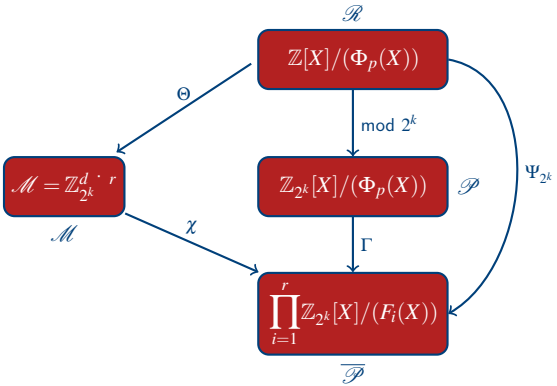
- We can have up to N isomorphisms

$$\psi_i : \mathbb{Z}_p[X]/F_i(X) \to \mathbb{F}_p$$

$\Rightarrow$ we can represent N plaintext elements of $\mathbb{F}_p$ as a single element in R_p.

BGV setting in our work

Use a ring defined by prime cyclotomic $\Phi_p(X)$.



BGV setting - Packing mod 2^k

Traditional setting...

Our setting...

Our packing technique

$\mathbb{I} = \{i_1, \ldots, i_t\}$ and $\mathbb{J} = \{j_1, \ldots, j_t\}$ such that $j_\ell = 2 \cdot i_\ell \quad \forall i_\ell \in \mathbb{I}$

$$\omega_{\mathbb{I}} : \begin{cases} (\mathbb{Z}_{2^k})^t & \longrightarrow & \mathbb{Z}_{2^k}[X] \\ (m_1, \ldots, m_t) & \longmapsto & m_1 \cdot X^{i_1} + \ldots + m_t \cdot X^{i_t}, \end{cases}$$

Our packing technique

$\mathbb{I} = \{i_1, \ldots, i_t\}$ and $\mathbb{J} = \{j_1, \ldots, j_t\}$ such that $j_\ell = 2 \cdot i_\ell \quad \forall i_\ell \in \mathbb{I}$

$$\omega_{\mathbb{I}} : \begin{cases} (\mathbb{Z}_{2^k})^t & \longrightarrow & \mathbb{Z}_{2^k}[X] \\ (m_1, \ldots, m_t) & \longmapsto & m_1 \cdot X^{i_1} + \ldots + m_t \cdot X^{i_t}, \end{cases}$$

$$\omega_{\mathbb{J}} : \begin{cases} (\mathbb{Z}_{2^k})^t & \longrightarrow & \mathbb{Z}_{2^k}[X] \\ (m_1, \ldots, m_t) & \longmapsto & m_1 \cdot X^{j_1} + \ldots + m_t \cdot X^{j_t}, \end{cases}$$

We encode $r \cdot |\mathbb{I}|$ $\mathbb{Z}_{2^k}$-elements into a single ciphertext

Our packing technique

$\mathbb{I} = \{i_1, \ldots, i_t\}$ and $\mathbb{J} = \{j_1, \ldots, j_t\}$ such that $j_\ell = 2 \cdot i_\ell \quad \forall i_\ell \in \mathbb{I}$

$$\omega_{\mathbb{I}} : \begin{cases} (\mathbb{Z}_{2^k})^t & \longrightarrow & \mathbb{Z}_{2^k}[X] \\ (m_1, \ldots, m_t) & \longmapsto & m_1 \cdot X^{i_1} + \ldots + m_t \cdot X^{i_t}, \end{cases}$$

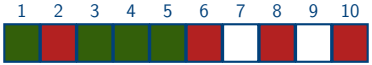
$$\omega_{\mathbb{J}} : \begin{cases} (\mathbb{Z}_{2^k})^t & \longrightarrow & \mathbb{Z}_{2^k}[X] \\ (m_1, \ldots, m_t) & \longmapsto & m_1 \cdot X^{j_1} + \ldots + m_t \cdot X^{j_t}, \end{cases}$$

We encode $r \cdot |\mathbb{I}|$ $\mathbb{Z}_{2^k}$-elements into a single ciphertext

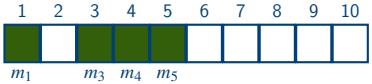
$\mathbb{I}$ is used to encode the intial packing
$\mathbb{J}$ is used to encode data after one multiplication

Our packing technique

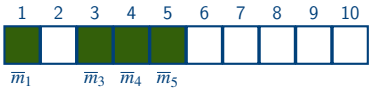
- $\mathbb{I} = \{1,3,4,5\}, \quad \mathbb{J} = \{2,6,8,10\}$



$m_1 \cdot X^1 + m_3 \cdot X^3 + m_4 \cdot X^4 + m_5 \cdot X^5$



$\overline{m}_1 \cdot X^1 + \overline{m}_3 \cdot X^3 + \overline{m}_4 \cdot X^4 + \overline{m}_5 \cdot X^5$



$\times \Longrightarrow$

$m_1\overline{m}_1 \cdot X^2 + m_3\overline{m}_3 \cdot X^6 + m_4\overline{m}_4 \cdot X^8 + m_5\overline{m}_5 \cdot X^{10}$

Our packing technique

$$R = \mathbb{Z}[X]/\Phi_p(X), \quad R_2 = (\mathbb{F}_{2^d})^r$$

$$\forall i \in \mathbb{I}, \quad 2 \cdot i < d$$

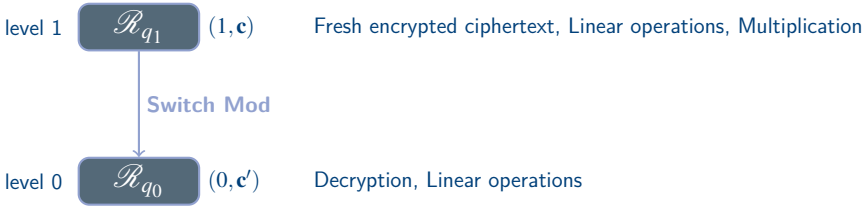
$$i_1 + i_2 \neq j, \quad i_1 \neq j, i_2 \neq j$$

$$\pi_p = \frac{r \cdot |\mathbb{I}|}{p-1}$$

p	r	d	$\lvert\mathbb{I}\rvert$	$r \cdot \lvert\mathbb{I}\rvert$	π_p
9719	226	43	8	1808	.186
11119	218	51	8	1744	.156
11447	118	97	16	1888	.164
13367	326	41	8	2608	.195
14449	172	84	16	2752	.190
20857	316	66	12	3792	.181
23311	518	45	8	4144	.177
26317	387	68	12	4644	.176
29191	278	105	16	4448	.152
30269	329	92	16	5264	.173
32377	568	57	10	5680	.175
38737	538	72	13	6994	.180
43691	1285	34	8	10280	.235
61681	1542	40	8	12336	.200

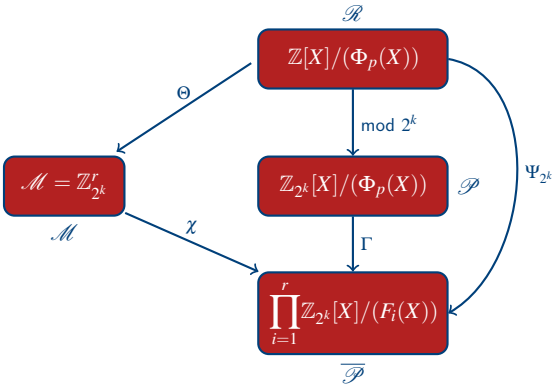
Two-level BGV encryption scheme

- $q_1 = p_0 \cdot p_1$ and $q_0 = p_0$

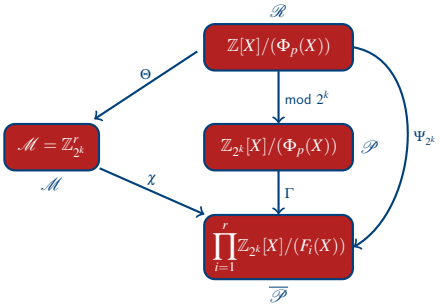


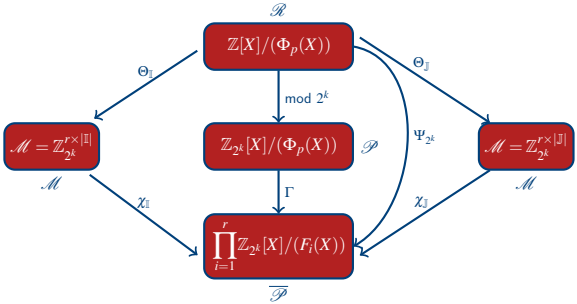
We can multiply ciphertexts at level 1, obtaining a ciphertext at level 0

Our packing technique



Our packing technique





MPC over $\mathbb{Z}_{2^k}$ from SHE

- Overdrive/HighGear preprocessing phase
  1. Packing technique should not leak any private information
  2. Modify the ZK proofs to check correctness of double-encrypted plaintexts

Efficiency

Protocol	k	s	sec	Triple Cost
This paper	32	32	26	**76.8**
SPDZ2k	32	32	26	79.87
This paper	64	64	57	**153.3**
SPDZ2k	64	64	57	319.488
This paper	128	64	57	**212.2**
SPDZ2k	128	64	57	557.06

k is the size of integers supported, i.e. MPC works modulo 2^k natively.

s is the expansion to support statistical security, i.e. internally SPDZ2k works modulo 2^{s+k}.

sec is the actual statistical security obtained.

Any Questions?