De l'utilisation du chiffrement homomorphe pour calculer dans le domaine chiffré

Simon Fau (CEA/Lab-STICC)
avec R. Sirdey (CEA), C. Fontaine (Lab-STICC),
C. Aguilar-Melchor (XLIM) et G. Gogniat
(Lab-STICC)

12 Octobre 2012

Sommaire

Sommaire

Introduction : le chiffrement homomorphe

Definition

Un cryptosystème E est dit **homomorphe** si une opération algébrique effectuée sur les chiffrés équivaut à une opération algébrique sur les clairs. En particulier, si E est stable pour un nombre arbitraire d'additions et de multiplications, il est dit **complètement homomorphe**.

Introduction : le chiffrement homomorphe

Definition

Un cryptosystème E est dit **homomorphe** si une opération algébrique effectuée sur les chiffrés équivaut à une opération algébrique sur les clairs. En particulier, si E est stable pour un nombre arbitraire d'additions et de multiplications, il est dit **complètement homomorphe**.

Applications possibles :

- Cloud computing
  - traitement de données sensibles
  - Requêtes masquées sur base de données chiffrée

Introduction : le chiffrement homomorphe

Definition

Un cryptosystème E est dit **homomorphe** si une opération algébrique effectuée sur les chiffrés équivaut à une opération algébrique sur les clairs. En particulier, si E est stable pour un nombre arbitraire d'additions et de multiplications, il est dit **complètement homomorphe**.

Applications possibles :

- Cloud computing
  - traitement de données sensibles
  - Requêtes masquées sur base de données chiffrée
- algorithmes ou fonctions privés et données confidentielles

Introduction : le chiffrement homomorphe

Definition

Un cryptosystème E est dit **homomorphe** si une opération algébrique effectuée sur les chiffrés équivaut à une opération algébrique sur les clairs. En particulier, si E est stable pour un nombre arbitraire d'additions et de multiplications, il est dit **complètement homomorphe**.

Applications possibles :

- Cloud computing
  - traitement de données sensibles
  - Requêtes masquées sur base de données chiffrée
- algorithmes ou fonctions privés et données confidentielles
- calcul multipartite
  - Smartgrids

État de l'art

- 1978 : Rivest et al., *Privacy Homomorphisms*

État de l'art

- 1978 : Rivest et al., *Privacy Homomorphisms*
- 1978-2009 : cryptosystèmes homomorphes uniquement pour l'addition ou la multiplication (Paillier, Goldwasser-Micali, ...)

État de l'art

- 1978 : Rivest et al., *Privacy Homomorphisms*
- 1978-2009 : cryptosystèmes homomorphes uniquement pour l'addition ou la multiplication (Paillier, Goldwasser-Micali, ...)
- 2009 : percée théorique de Gentry (STOC'09) avec la construction du premier système complètement homomorphe non cassé.
  Concrètement :

$$p_{\oplus,\otimes}(m_1, ..., m_n) = Dec(p_{add,mul}(Enc(m_1), ..., Enc(m_n)))$$

  - cryptographie basée sur les réseaux
  - inefficace en pratique : le surcoût des calculs est trop important (overhead polynomial)

État de l'art

- 1978 : Rivest et al., *Privacy Homomorphisms*
- 1978-2009 : cryptosystèmes homomorphes uniquement pour l'addition ou la multiplication (Paillier, Goldwasser-Micali, ...)
- 2009 : percée théorique de Gentry (STOC'09) avec la construction du premier système complètement homomorphe non cassé.
  Concrètement :

$$p_{\oplus,\otimes}(m_1, ..., m_n) = Dec(p_{add,mul}(Enc(m_1), ..., Enc(m_n)))$$

  - cryptographie basée sur les réseaux
  - inefficace en pratique : le surcoût des calculs est trop important (overhead polynomial)
- Depuis : beaucoup d'autres systèmes, mais très peu d'implémentations. (HCRYPT par Brenner et al., schéma de Smart-Vercauteren)

État de l'art

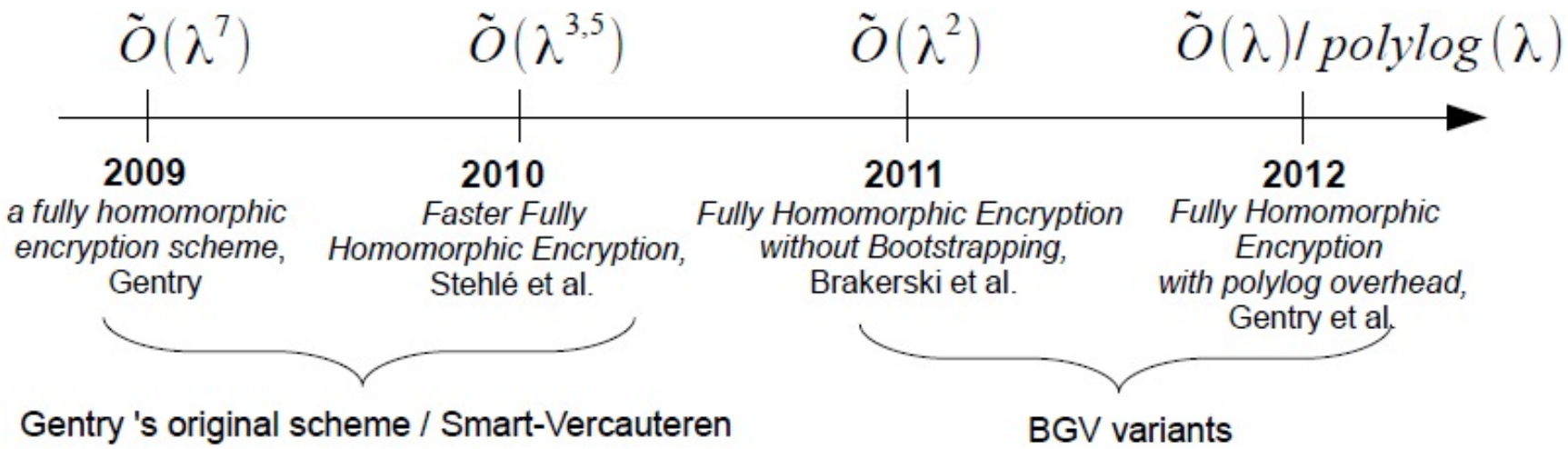
- 1978 : Rivest et al., *Privacy Homomorphisms*
- 1978-2009 : cryptosystèmes homomorphes uniquement pour l'addition ou la multiplication (Paillier, Goldwasser-Micali, ...)
- 2009 : percée théorique de Gentry (STOC'09) avec la construction du premier système complètement homomorphe non cassé.
  Concrètement :

$$p_{\oplus,\otimes}(m_1, ..., m_n) = Dec(p_{add,mul}(Enc(m_1), ..., Enc(m_n)))$$

  - cryptographie basée sur les réseaux
  - inefficace en pratique : le surcoût des calculs est trop important (overhead polynomial)
- Depuis : beaucoup d'autres systèmes, mais très peu d'implémentations. (HCRYPT par Brenner et al., schéma de Smart-Vercauteren)

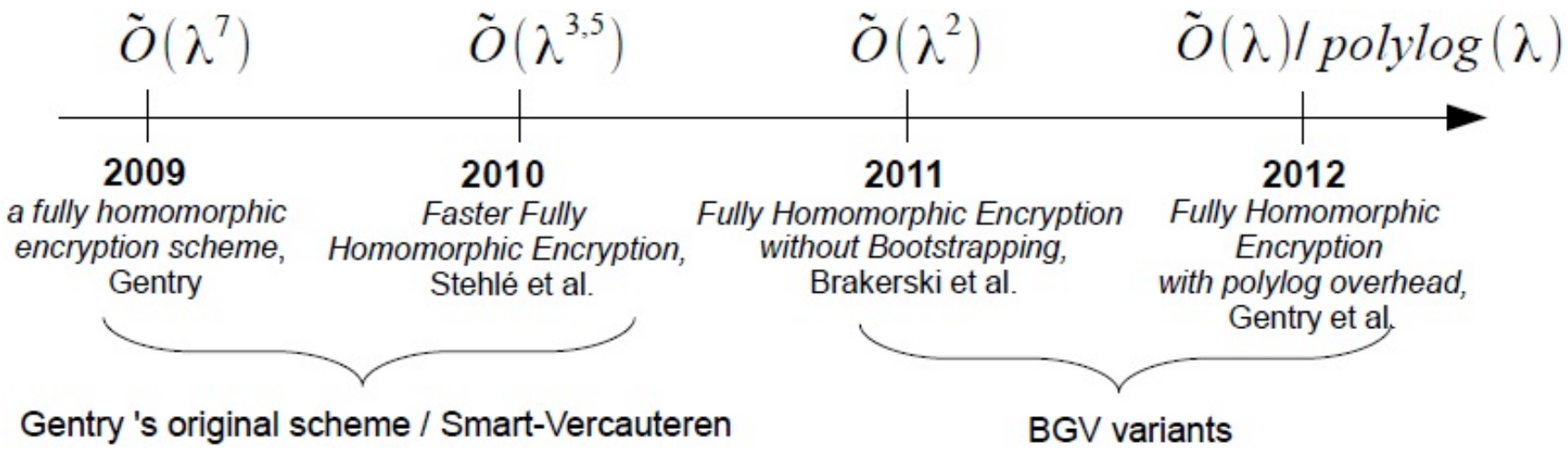
Notre approche

- Evolution de l'overhead par bit à chiffrer en fonction du paramètre de sécurité λ (meilleures attaques connues en 2^{λ}) :



Notre approche

- Evolution de l'overhead par bit à chiffrer en fonction du paramètre de sécurité λ (meilleures attaques connues en 2^{λ}) :



- Objectifs de notre démarche :
  - avoir des repères sur les performances concrètes du chiffrement homomorphe
  - modularité de l'implémentation
  - facilité d'utilisation pour des programmeurs

Sommaire

Le principe de BGV

- Clé privée : s
- Clé publique : $(As + 2e, -A)$, où $e \in \chi$
- Chiffré de $m \in \{0, 1\}$:
  $(c_1 = m + (As + 2e)r, c_2 = Ar) \in \mathbb{A}_q^2$
- Déchiffrement :

$$c_1 - c_2 s \bmod q = m + 2er$$

- Addition de deux chiffrés :

$$c_1 + c_1' = m + m' + As(r + r') + 2(er + er')$$

$$c_2 + c_2' = A(r + r')$$

Condition nécessaire pour déchiffrer : $2(er + er') + m + m' < q$

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).
On génère L modules $q_1 < ... < q_L$.

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).
On génère L modules $q_1 < ... < q_L$.
Un chiffré (chiffrant $m \in \{0, 1\}$ par rapport à une clé privée $s \in \mathbb{A}_{q_1}$) est un couple de polynômes de $\mathbb{A}_{q_i} = \mathbb{A}/q_i\mathbb{A}$:

$c_1 = m + (\mathbf{A}s + 2\mathbf{e}).\mathbf{r}$

$c_2 = \mathbf{A}.\mathbf{r}$, où $\mathbf{r} \in \mathbb{A}_2^N$, $\mathbf{A} \in \mathbb{A}_{q_i}^N$ et $\mathbf{e} \leftarrow \chi^N$

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).
On génère L modules $q_1 < ... < q_L$.
Un chiffré (chiffrant $m \in \{0, 1\}$ par rapport à une clé privée $s \in \mathbb{A}_{q_1}$) est un couple de polynômes de $\mathbb{A}_{q_i} = \mathbb{A}/q_i\mathbb{A}$:

$c_1 = m + (\mathbf{A}s + 2\mathbf{e}).\mathbf{r}$

$c_2 = \mathbf{A}.\mathbf{r}$, où $\mathbf{r} \in \mathbb{A}_2^N$, $\mathbf{A} \in \mathbb{A}_{q_i}^N$ et $\mathbf{e} \leftarrow \chi^N$

- Addition : $(c_1, c_2), (c_1', c_2') \in \mathbb{A}_{q_i}^2$, $(c_1 + c_1', c_2 + c_2') \in \mathbb{A}_{q_i}$.

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).
On génère L modules $q_1 < ... < q_L$.
Un chiffré (chiffrant $m \in \{0, 1\}$ par rapport à une clé privée $s \in \mathbb{A}_{q_1}$) est un couple de polynômes de $\mathbb{A}_{q_i} = \mathbb{A}/q_i\mathbb{A}$:

$c_1 = m + (\mathbf{A}s + 2\mathbf{e}).\mathbf{r}$

$c_2 = \mathbf{A}.\mathbf{r}$, où $\mathbf{r} \in \mathbb{A}_2^N$, $\mathbf{A} \in \mathbb{A}_{q_i}^N$ et $\mathbf{e} \leftarrow \chi^N$

- Addition : $(c_1, c_2), (c_1', c_2') \in \mathbb{A}_{q_i}^2$, $(c_1 + c_1', c_2 + c_2') \in \mathbb{A}_{q_i}$.
- Multiplication : $(c_1, c_2), (c_1', c_2') \in \mathbb{A}_{q_i}^2$, on fait le produit tensoriel : $(c_1, c_2) \otimes (c_1', c_2') = (c_1.c_1', c_1.c_2' + c_2.c_1', c_2.c_2')$

Le cryptosystème BGV

On considère l'anneau $\mathbb{A} = \mathbb{Z}^n[X]/\Phi(X)$, où $\Phi(X)$ est un polynôme cyclotomique, par exemple $X^d + 1$ ($n = 1$ et $d = 2^k$ dans la version polynomiale, $d = 1$ dans la version vectorielle).
On génère L modules $q_1 < ... < q_L$.
Un chiffré (chiffrant $m \in \{0, 1\}$ par rapport à une clé privée $s \in \mathbb{A}_{q_1}$) est un couple de polynômes de $\mathbb{A}_{q_i} = \mathbb{A}/q_i\mathbb{A}$:

$c_1 = m + (\mathbf{A}s + 2\mathbf{e}).\mathbf{r}$

$c_2 = \mathbf{A}.\mathbf{r}$, où $\mathbf{r} \in \mathbb{A}_2^N$, $\mathbf{A} \in \mathbb{A}_{q_i}^N$ et $\mathbf{e} \leftarrow \chi^N$

- Addition : $(c_1, c_2), (c_1', c_2') \in \mathbb{A}_{q_i}^2$, $(c_1 + c_1', c_2 + c_2') \in \mathbb{A}_{q_i}$.
- Multiplication : $(c_1, c_2), (c_1', c_2') \in \mathbb{A}_{q_i}^2$, on fait le produit tensoriel : $(c_1, c_2) \otimes (c_1', c_2') = (c_1.c_1', c_1.c_2' + c_2.c_1', c_2.c_2')$
- Mise à niveau : $(c_1, c_2) \in \mathbb{A}_{q_i}^2 \rightarrow (c_1', c_2') \in \mathbb{A}_{q_j}^2$, avec $j < i$
  - opération nécessaire après chaque multiplication (coûteux)
  - intérêt : réduction du bruit

Sommaire

Implémentation de BGV

Cette implémentation recouvre plusieurs aspects :

Implémentation de BGV

Cette implémentation recouvre plusieurs aspects :

- génie logiciel : expression haut niveau d'opérateurs et d'algorithmes
  - addition, multiplication (n-bits), soustraction (complément à 2), opérateurs «, », !, tri à bulles, FFT
  - class C++ template<typename bit, int size>

Implémentation de BGV

Cette implémentation recouvre plusieurs aspects :

- génie logiciel : expression haut niveau d'opérateurs et d'algorithmes
  - addition, multiplication (n-bits), soustraction (complément à 2), opérateurs «, », !, tri à bulles, FFT
  - class C++ template<typename bit, int size>
- algorithmique : adaptation des primitives algorithmiques de base aux contraintes de l'homomorphe
  - avoir recours le moins possible à des multiplications dans les algorithmes

Implémentation de BGV

Cette implémentation recouvre plusieurs aspects :

- génie logiciel : expression haut niveau d'opérateurs et d'algorithmes
  - addition, multiplication (n-bits), soustraction (complément à 2), opérateurs «, », !, tri à bulles, FFT
  - class C++ template<typename bit, int size>
- algorithmique : adaptation des primitives algorithmiques de base aux contraintes de l'homomorphe
  - avoir recours le moins possible à des multiplications dans les algorithmes
  - Caractérisation de quelques algorithmes de base :

	$\sum_{i=1}^{10} t[i]$ (16 bits)	$b^2 - 4ac$ (16 bits)	tri à bulles (10x8 bits)	FFT (256x32 bits)
# add	423	1188	3240	7291592
# mul	279	1126	2790	5296128
× depth	16	32	136	166
Av. //	14.62	27.88	17.23	18676.10

Autres aspects de notre implémentation

Autres aspects de notre implémentation

Prototype d'infrastructure de compilation d'algorithmes de haut niveau vers un support d'exécution homomorphe permettant :

Autres aspects de notre implémentation

Prototype d'infrastructure de compilation d'algorithmes de haut niveau vers un support d'exécution homomorphe permettant :

- Support pour le contrôle dépendant des données (affectation conditionnelle, déréférencement)

Autres aspects de notre implémentation

Prototype d'infrastructure de compilation d'algorithmes de haut niveau vers un support d'exécution homomorphe permettant :

- Support pour le contrôle dépendant des données (affectation conditionnelle, déréférencement)
- Construction de circuits booléens

Autres aspects de notre implémentation

Prototype d'infrastructure de compilation d'algorithmes de haut niveau vers un support d'exécution homomorphe permettant :

- Support pour le contrôle dépendant des données (affectation conditionnelle, déréférencement)
- Construction de circuits booléens
- Exécution parallèle littérale ou niveau circuits

Premiers résultats expérimentaux

Exécution parallèle littérale (version vectorielle de BGV) sur un Intel dual core à 2GHz, comparé à HCRYPT, pour une sécurité de $\lambda \sim 15$:

Premiers résultats expérimentaux

Exécution parallèle littérale (version vectorielle de BGV) sur un Intel dual core à 2GHz, comparé à HCRYPT, pour une sécurité de $\lambda \sim 15$:

	$b^2 - 4ac$ (8 bits)	$b^2 - 4ac$ (16 bits)
BGV (CPU)	0.406 s	4.124 s
BGV (PK size)	1.1 MB	7.8 MB
HCRYPT	58.9 s	3 m 39 s
	$\sum_{i=1}^{10} t[i]$ (8 bits)	$\sum_{i=1}^{10} t[i]$ (16 bits)
BGV (CPU)	0.125 s	0.562 s
BGV (PK size)	196 kB	1.1 MB
HCRYPT	27.2 s	55.4 s
	b. sort (10×4 bits)	b. sort (10×8 bits)
BGV (CPU)	5.219 s	18.110 s
BGV (PK size)	68.5 MB	525 MB
HCRYPT	5 m 5 s	9 m 41 s

Premiers résultats expérimentaux

Version polynomiale de BGV (sans parallélisme).

	$\sum_{i=1}^{10} t[i]$ (8 bits)
BGV poly. ($\lambda \sim 40$)	57.3 s

Rappel :

	$\sum_{i=1}^{10} t[i]$ (8 bits)
HCRYPT ($\lambda \sim 15$)	27.2 s
BGV vect. ($\lambda \sim 15$)	0.125 s

Expérimentations en cours ...

Sommaire

Conclusion et perspectives

- Bilan des objectifs

Conclusion et perspectives

- Bilan des objectifs
  - Modularité de l'implémentation : état de l'art instable (beaucoup de publications de nouveaux cryptosystèmes ou de nouvelles versions)
  - Performances concrètes de chiffrement homomorphe :
    - par exemple opérateurs non linéaires simples en traitement du signal (par ex. seuil)
    - les algorithmes plus volumineux sont hors de portée pour le moment
  - Exploitation du parallélisme que permet de faire l'implémentation

Conclusion et perspectives

- Bilan des objectifs
  - Modularité de l'implémentation : état de l'art instable (beaucoup de publications de nouveaux cryptosystèmes ou de nouvelles versions)
  - Performances concrètes de chiffrement homomorphe :
    - par exemple opérateurs non linéaires simples en traitement du signal (par ex. seuil)
    - les algorithmes plus volumineux sont hors de portée pour le moment
  - Exploitation du parallélisme que permet de faire l'implémentation
- Perspectives :
  - rendre les algorithmes plus « homomorphic-friendly »
  - utilisation conjointe avec des algorithmes de chiffrement symétrique (par ex. AES)
  - pousser le niveau de sécurité

leti

LABORATOIRE D'ÉLECTRONIQUE
ET DE TECHNOLOGIES
DE L'INFORMATION

list

LABORATOIRE D'INTÉGRATION
DES SYSTÈMES
ET DES TECHNOLOGIES

Merci pour votre attention !

digiteo

INSTITUT CARNOT CEA LIST

cea

MINATEC

INSTITUT CARNOT CEA LETI