Homomorphic Encryption

Louis Goubin
Université de Versailles-St-Quentin-en-Yvelines

Université Paris-Saclay

M1 informatique – Site de Versailles

UE « Calcul Sécurisé »

4ème partie

Introduction

Common notations:

pk – public key

sk – secret key

m – message

c – ciphertext

$c = \mathrm{Encrypt}_{pk}(m)$

$m = \mathrm{Decrypt}_{sk}(c)$

$E_m(pk)$ - encryption algorithm as a circuit

$D_m(sk)$ - decryption algorithm as a circuit

f – is the function or circuit that we want to evaluate on plaintext

F – is the function or circuit that corresponds to f and operates on ciphertext in the cryptosystem

$c_1 = \mathrm{Encrypt}_{pk}(m_1)$

...

$c_n = \mathrm{Encrypt}_{pk}(m_n)$

↓

$c = F(c_1, \ldots, c_n)$

↓

$m = \mathrm{Decrypt}_{sk}(c)$

Partially HE

Multiplicative Partially HE

Unpadded RSA

pk=(n,e)

$c=E_{pk}(m)=m^e \bmod n$

$c_1 * c_2 = m_1^e m_2^e \bmod n = E_{pk}(m_1 * m_2)$

Additive Partially HE

Paillier scheme

pk=(n,g)

$c=E_{pk}(m)=g^m r^n \bmod n^2$

r in {0,…,n-1} – some random value

$c_1 * c_2 = (g^{m1} r1^n) * (g^{m2} r_2^n) \bmod n =$

$g^{m1+m2} (r_1 r_2)^n \bmod n = E_{pk}(m_1+m_2)$

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

Does FHE Ever Exists?

Fully Homomorphic Encryption (FHE). Some Properties.

FHE property (simplified):

- $Decrypt_{sk}(c_1*c_2)=m_1*m_2$
- $Decrypt_{sk}(c_1+c_2)=m_1+m_2$

I.e.:

$$Decrypt_{sk}(F(c_1,\ldots,c_n))=F(m_1,\ldots,m_n)$$

FHE may support another set of operations to support a ring of plaintexts. Examples: AND, XOR

FHE can be:

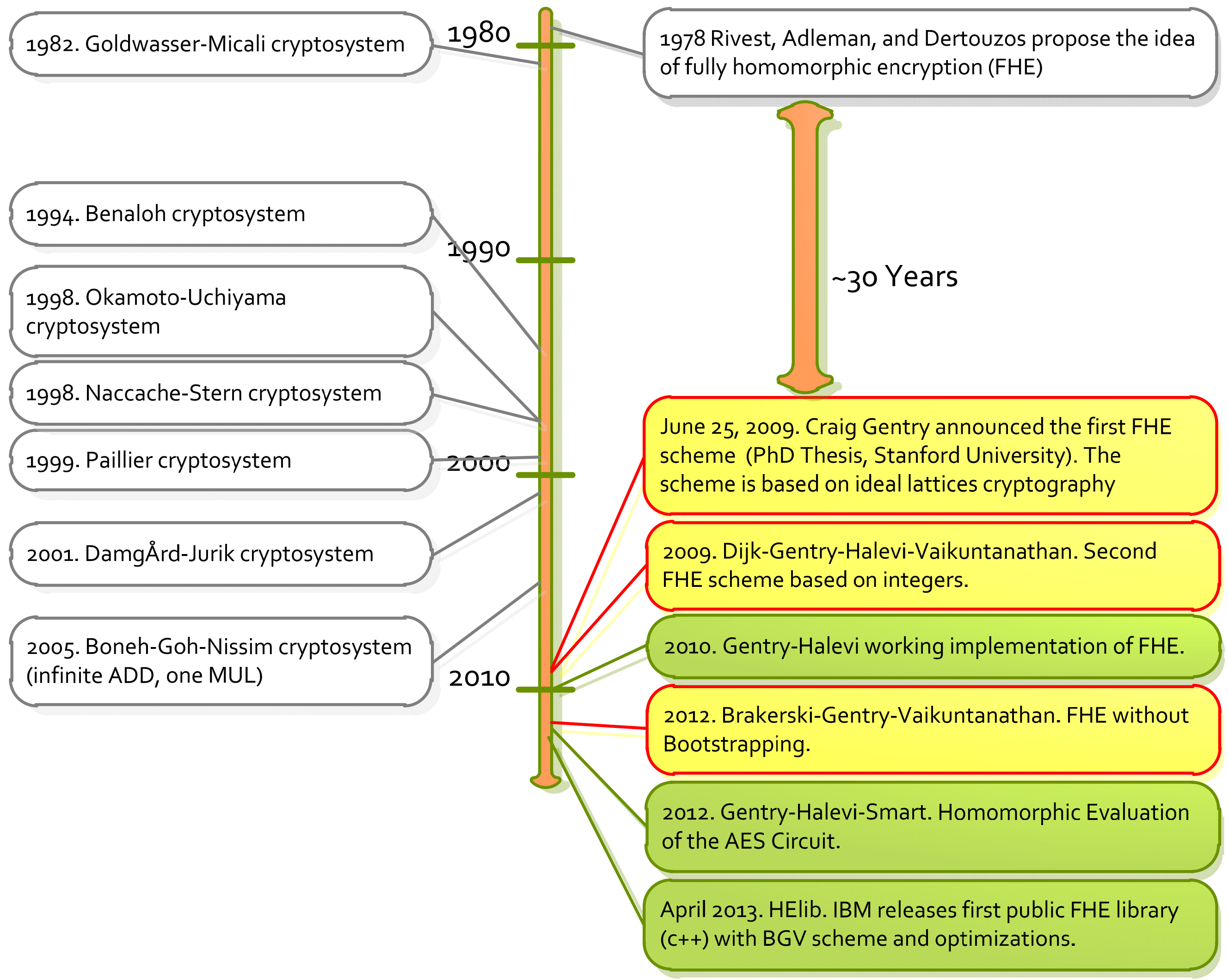
- Public key schemes
- Symmetric key schemes

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

cnrs

“Holy Grail” for 30 years



UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

Types of HE Schemes

Homomorphic Encryption (HE) = type of computation for a set of functions $f(m_1,\ldots,m_n)$ carried on ciphertexts $Enc(m_1)\ldots Enc(m_n)$ with a corresponding function F such that

$$f(m_1,\ldots,m_n) = Dec(F(Enc(m_1),\ldots,Enc(m_n)))$$

Partially HE (PHE) = HE scheme where only one type of operations is possible (multiplication or addition)

Somewhat HE (SHE) = HE scheme that can do a **limited** number of additions and multiplications

Fully HE (FHE) = HE scheme that can perform an **infinite** number of additions and multiplications

SHE over the Integers

SHE over the INTEGERS – SYMMETRIC KEY SCHEME
2009. Dijk-Gentry-Halevi-Vaikuntanathan. Second FHE scheme based on integers.

KeyGen: key is an odd integer p$\in[2^{\eta-1}, 2^{\eta})$.
Encrypt(p, m): to encrypt one bit m$\in[0,1]$

$$c = pq+2r+m$$

q, r – are chosen random, such that |2r|<p/2.
Decrypt(p, c):

$$m = (c \bmod p) \bmod 2$$

Proposed constraint: p~2^{η}, q~$2^{\eta\wedge 3}$, r~$2^{\mathrm{sqrt}(\eta)}$.

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

SHE over the Integers

ADDITION:

Decrypt(c_1+c_2,p) = $((pq_1+2r_1+m_1)+(pq_2+2r_2+m_2) \bmod p) \bmod 2$

$= (p(q_1+q_2)+2(r_1+r_2)+(m_1+m_2) \bmod p) \bmod 2$

$= 2(r_1+r_2)+(m_1+m_2) \bmod 2$

$= m_1 \oplus m_2$

MULTIPLICATION:

Decrypt(c_1*c_2,p) $=((pq_1+2r_1+m_1)*(pq_2+2r_2+m_2) \bmod p) \bmod 2$

$=(p(pq_1q_2+2q_1r_2+q_1m_2+2q_2r_1+q_2m_1)+$

$2(2r_1r_2+r_1m_2+m_2r_1)+m_1m_2) \bmod p) \bmod 2$

$= 2(2r_1r_2+r_1m_2+m_2r_1)+m_1m_2 \bmod 2$

$=$ m1 $\otimes$ m2

- The scheme is both additively and multiplicatively homomorphic for shallow arithmetic circuits.
- The number of ADD and MUL is limited since the noise grows.
- The noise r must be sufficiently smaller than p to allow more ADDs and MULs.

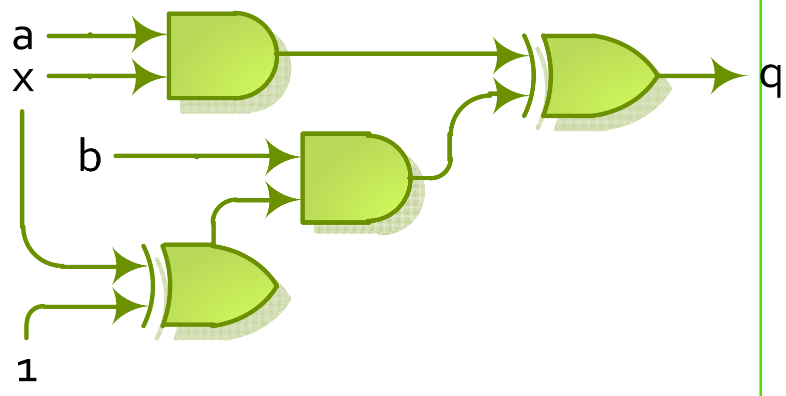
UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

SHE over the Integers

Any computer program can be represented in terms of AND-XOR gates

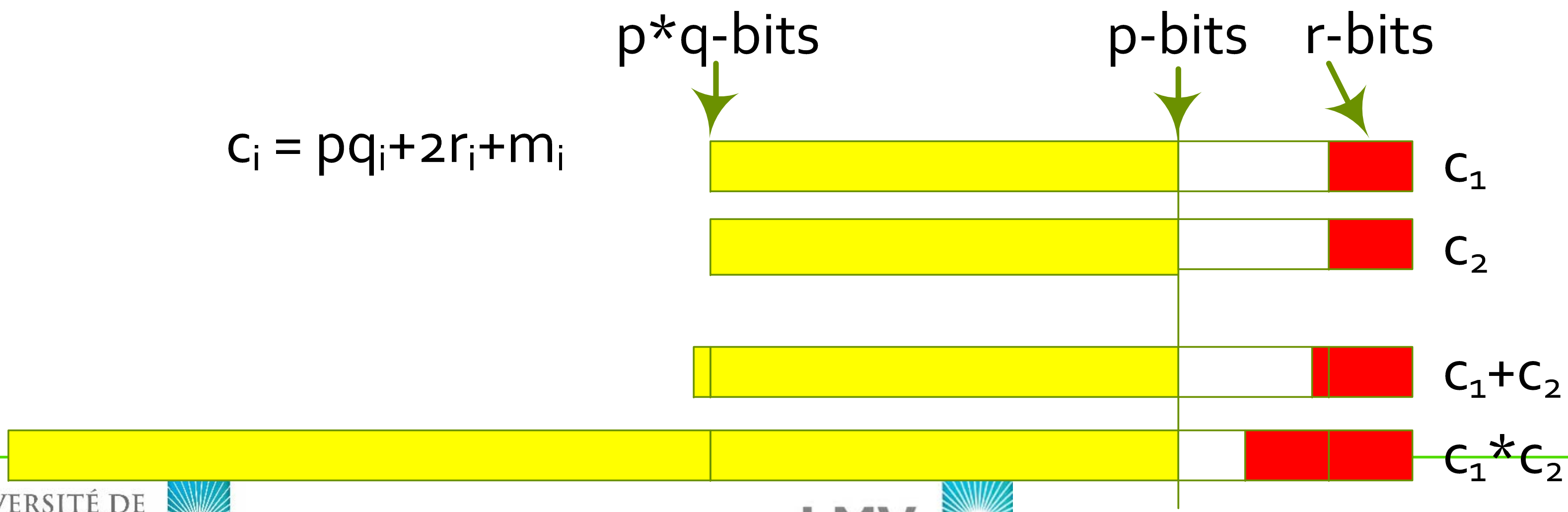
f(x, a, b): if(x==1) q=a; else q=b



$F(c_a, c_x, c_b, c_1)$: $c_q = (c_a * c_x)+(c_x+c_1)*c_b$

Ciphertext and noise size expansion

$c_i = pq_i+2r_i+m_i$



Noise Count

Noise Magnitude

The degree of noise can be calculated by the evaluator internally.
Assume two arguments to an operation

- c_1 – has the noise of degree n_1 bits
- c_2 – has the noise of degree n_2 bits

The result of:

- $ADD(c_1, c_2)$ => noise of degree $\log_2(2^{n1}+2^{n2})$ bits
- $MUL(c_1, c_2)$ => noise of degree (n_1+n_2) bits

Only when the noise would be larger than p-bits then the evaluator must do the “refreshment” step.

SHE over the Integers

SHE over the INTEGERS – PUBLIC KEY SCHEME

KeyGen: pk=$<x_0,\ldots,x_t>$, where:
$x_i = pq_i+r_i$, unless x_0 is the largest, q_0 is odd and r_0 is even
Encrypt(pk, m$\in$\{0,1\}): $c = (m+2r+2\sum_{i\in S}(x_i)) \bmod x_0$
Where S is a random subset of pk, and r is a random noise.
Decrypt(sk=p, c): $m = (c \bmod p) \bmod 2$

Encryption can now be viewed as:
adding m to a random subset sum of "encryptions of zero"

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

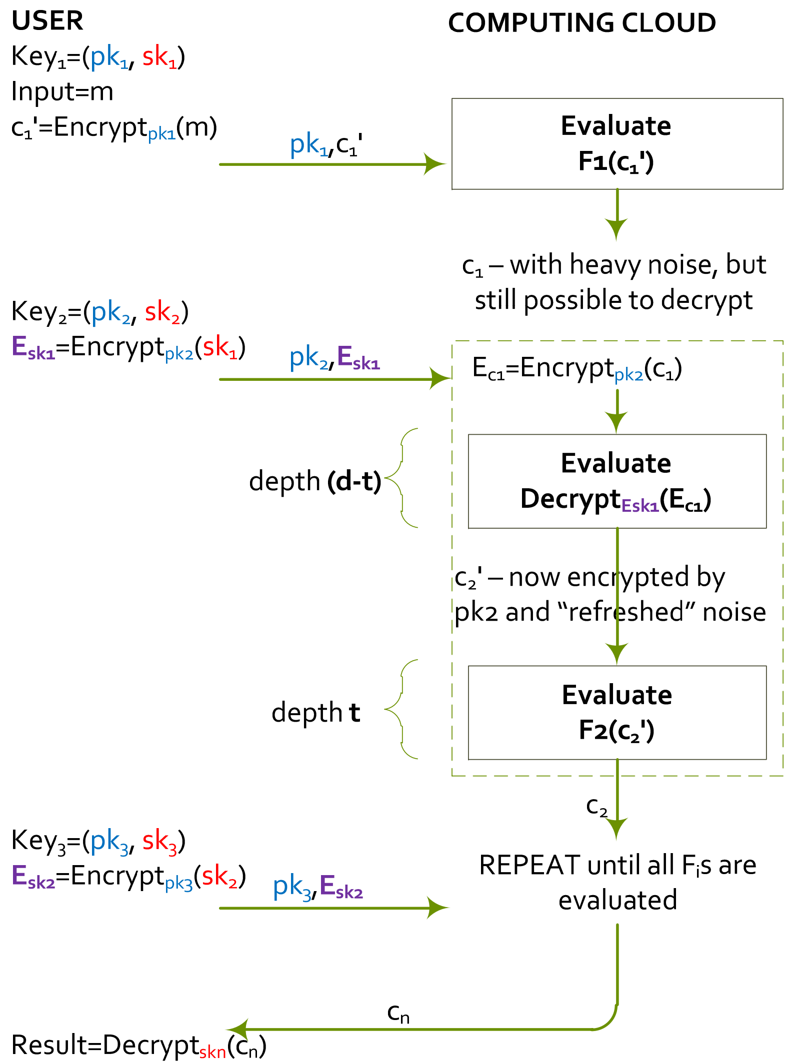
LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

Bootstrapping

Bootstrapping

- Assume SHE can evaluate circuits F up to the **depth d**.
- Assume $F=D_{sk}(m)$ is the decryption algorithm, that can be evaluated by SHE **at most depth (d-1)**. I.e., to decrypt plus "a little more".

Then SHE is called ***bootstrappable*** and can be converted to FHE.



UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100

cnrs

SHE -> FHE over the Integers

FHE over the INTEGERS – Convert SHE to FHE

$F=D_{sk}(m)$ becomes simple if we use the technique called "squashing the decryption circuit".

Idea: add to pk some information about sk so that $D_{sk}(m)$ becomes simpler and SWH becomes bootstrappable.

Price:

- public key contains more information
- ciphertext is larger

New Recent Research

Found different ways to construct FHE without using the squashing step

2011. Craig Gentry and Shai Halevi. **Fully homomorphic encryption without squashing using depth-3 arithmetic circuits.**

2011. Zvika Brakerski and Vinod Vaikuntanathan. **Efficient fully homomorphic encryption from (standard) LWE.**

Way to Evaluate on packed ciphertext

2011. N.P. Smart and F. Vercauteren. **Fully Homomorphic SIMD Operations.**

BGV scheme to construct a FHE of a desired depth D, based on Ring LWE. I.e., no bootstrapping is needed.

2012. Zvika Brakerski, Craig Gentry and Vinod Vaikuntanathan. **Fully homomorphic encryption without bootstrapping.**

2012. Craig Gentry, Shai Halevi, and Nigel P. Smart. **Homomorphic evaluation of the AES circuit.**

2013. **HELib.** IBM.

2013. **CryptDB.** MIT.

2013. Jacob Alperin-Sheriff and Chris Peikert. **Practical bootstrapping in Quasilinear Time.**

Performance

2010. SHE Performance

1 000 000 000 000x

[…] A simple string search using homomorphic encryption is about a **trillion** times slower than without encryption.

Dimension	KeyGen	Enc (amort)	Mult/Dec	Degree
2048 (800,000 bit ints)	1.25 s.	.060 sec	.023 s.	~200
8192 (3,200,000 bit ints)	10 s.	.7 sec	.12 s.	~200
32768 (13,000,000 bit ints)	95 s.	5.3 sec	.6 s.	~200

2010. FHE Performance

Dimension	KeyGen	PK size	ReCrypt
2048	40 s.	70 Mbyte	31 sec
8192	8 min.	285 Mbyte	3 min
32768	2 hrs	2.3 Gb	30 min

Evaluation of AES

2012. Homomorphic Evaluation of the AES Circuit

Gentry-Halevi-Smart
AES-128, 10 rounds
256GB of RAM, D=60.

Variant 1: 36 hours, 54 blocks
(SIMD technique to use more plaintext slots in each ciphertext, so that operations are done in parallel for free)
First round – 7 hours
Last round – 30 min

Amortized speed ~ 40 min/block

Variant 2: 2,5 days, 720 blocks
Amortized speed ~ 5 min/block

HElib

2013. HElib. IBM.

Based on BGV scheme, based on ideal lattices, SIMD operations on packed ciphertext, quasi-linear bootstrapping, and many other improvement technique.
- Does not support bootstrapping (reencrypt) operation.
- GPL licensed

λ =80 bits of security

Modulus	Number of Slots	Time for ADD (ms)	Time for MUL (ms)
257	44	0.7	39
8209	22	0.7	38
65537	2	2.9	177