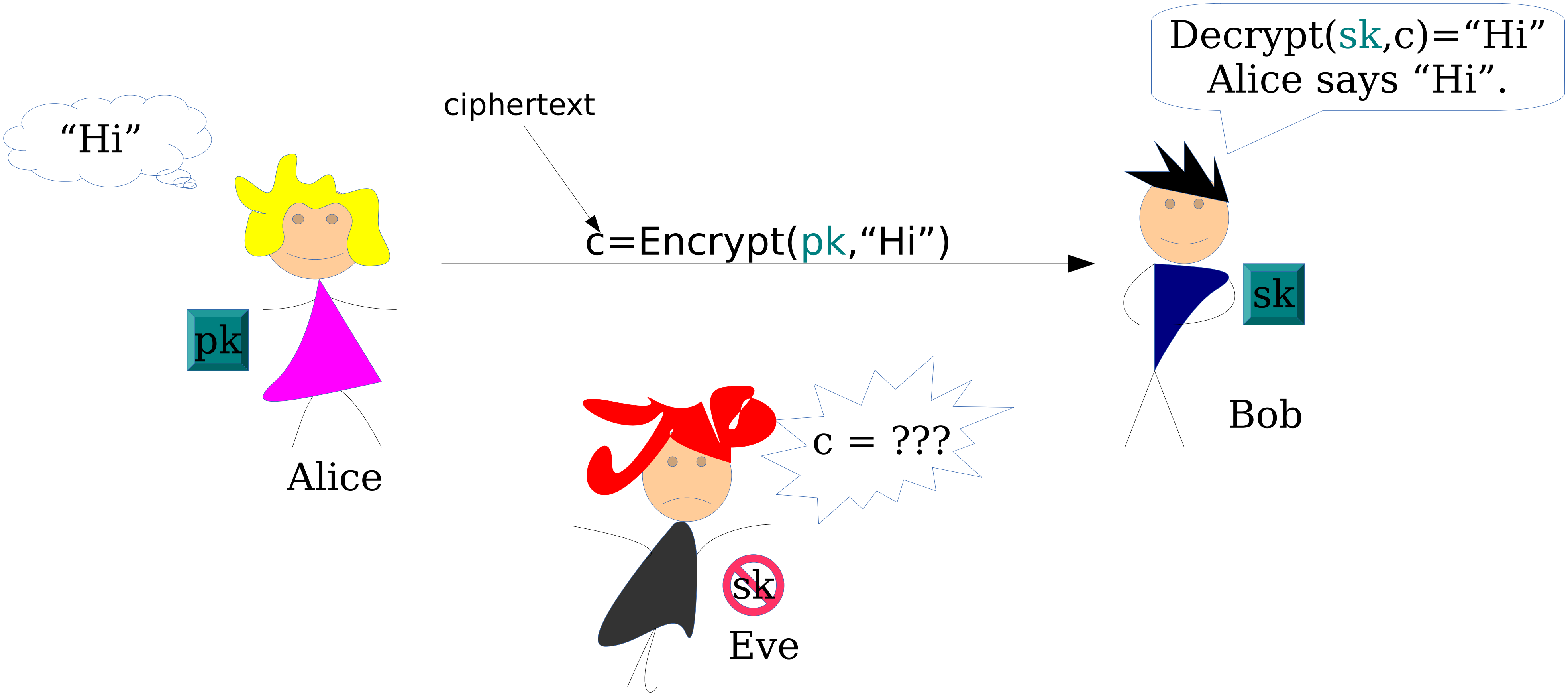
Lattices, Post-Quantum Security, and Fully Homomorphic Encryption

Daniele Micciancio
(UC San Diego)

April 2020

Encryption

- Secure communication over insecure channel



Modern Cryptography

- Hard mathematical Problem:
  - Factoring: Given pq, find p and q
- Cryptographic Construction:
  - Encryption scheme
- Proof of security:

If you can break encryption → then you can factor numbers

the encryption is secure ← If factoring is hard

Factoring and Quantum (In)Security

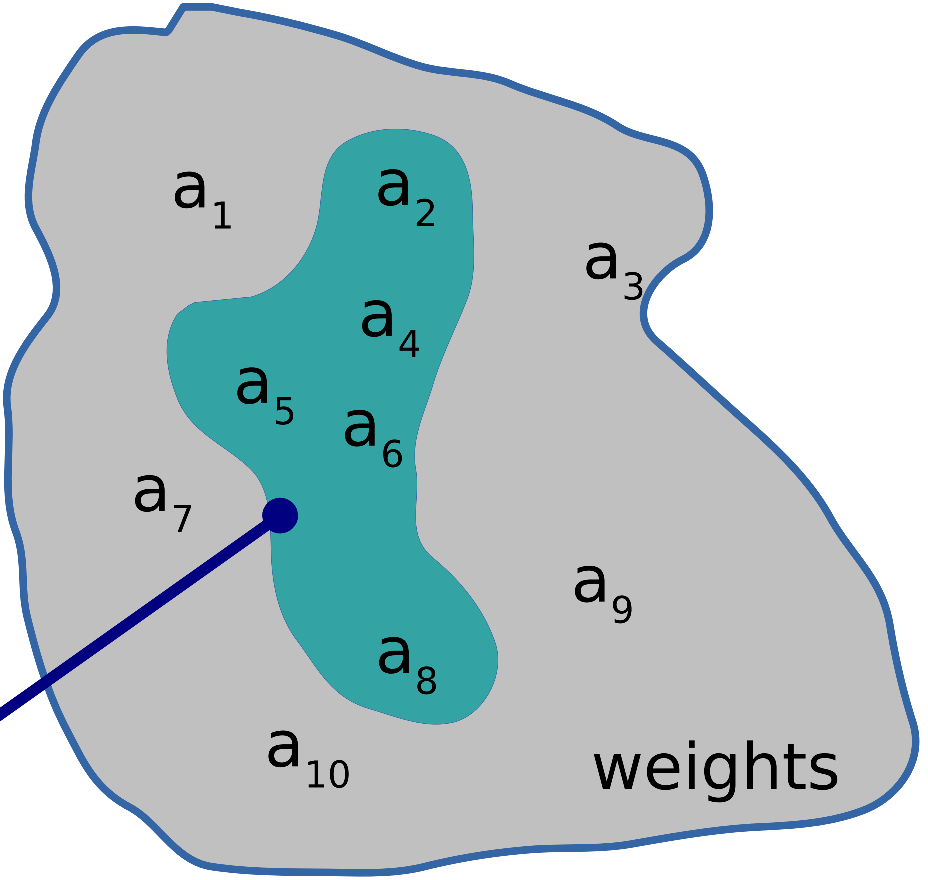
- Shor (1994):
  - efficient quantum algorithm to factor numbers
- Assumption that factoring is hard does not hold in a “post-quantum” world
- Same holds for most other mathematical problems currently in use:
  - discrete logarithm, elliptic curve, etc.
- Need for new mathematical problems that are not solvable by quantum algorithms

Subset-Sum Problem

- Given n integer numbers
  - $a_1,\ldots,a_n$

  and a target value
  - b
- Goal:
  - Find a subset that adds up to b

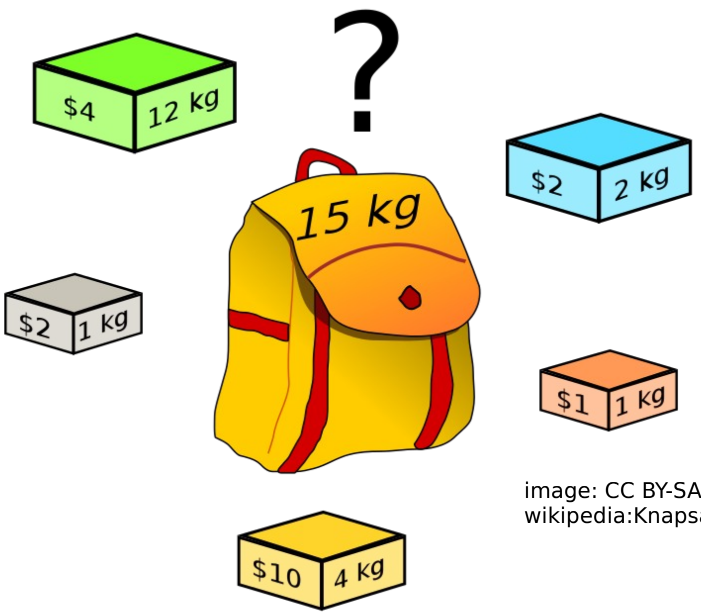
    $\Sigma \{a_i \mid i \in S\} = b$



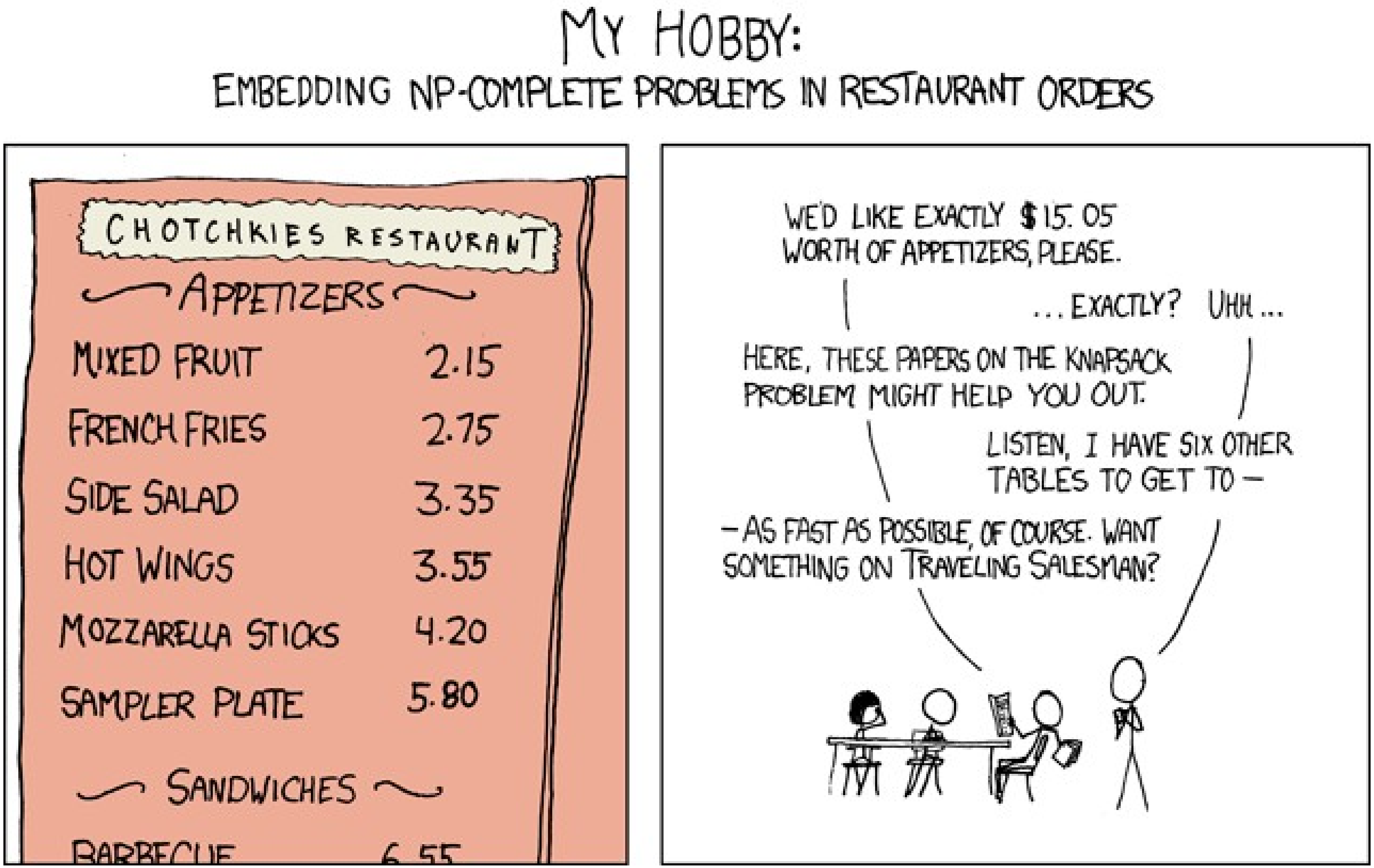
$$b=a_2+a_4+a_5+a_6+a_8$$

Subset-Sum / Knapsack

- Also known as the “Knapsack” problem
  - Fill a knapsack of capacity b
  - using a selection of items of size $a_1,...,a_n$
  - items can be used multiple times



Try it out!



https://xkcd.com/licence.html

Hardness of Subset-Sum

- NP-complete: no efficient algorithm unless P=NP (or NP ⊂ BQP)
- One of Karp 21 NP-complete problems
  - [Karp 1972]

NP-hard problems:

1. Set packing
2. Vertex Cover

...

18. Knapsack

...

21. Max Cut

REDUCIBILITY AMONG COMBINATORIAL PROBLEMS†

Richard M. Karp

University of California at Berkeley

Abstract: A large class of computational problems involve the determination of properties of graphs, digraphs, integers, arrays of integers, finite families of finite sets, boolean formulas and elements of other countable domains. Through simple encodings from such domains into the set of words over a finite alphabet these problems can be converted into language recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily solved when an algorithm for its solution is found which terminates within a number of steps bounded by a polynomial in the length of the input. We show that a large number of classic unsolved problems of covering, matching, packing, routing, assignment and sequencing are equivalent, in the sense that either each of them possesses a polynomial-bounded algorithm or none of them does.

1. INTRODUCTION

All the general methods presently known for computing the chromatic number of a graph, deciding whether a graph has a Hamilton circuit, or solving a system of linear inequalities in which the variables are constrained to be 0 or 1, require a combinatorial search for which the worst case time requirement grows exponentially with the length of the input. In this paper we give theorems which strongly suggest, but do not imply, that these problems, as well as many others, will remain intractable perpetually.

†This research was partially supported by National Science Foundation Grant GJ-474.

Lattice/Knapsack Cryptgraphy: abridged (pre-)history

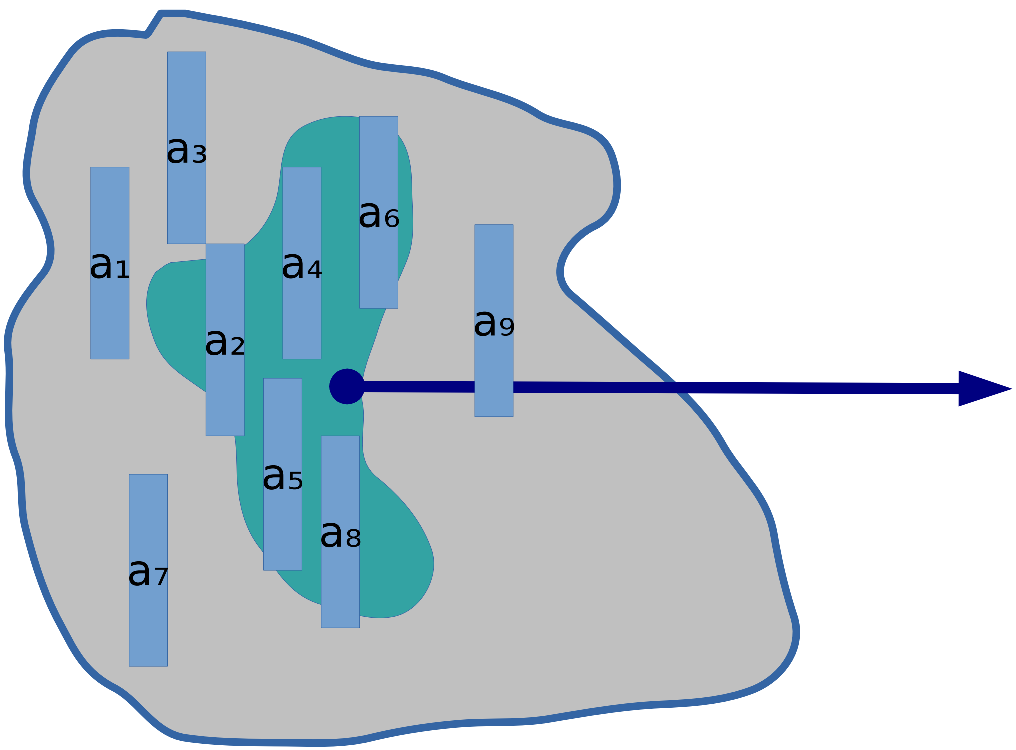
- Knapsack public key cryptosystem
  - [Merkle, Hellman 1978]
- Cryptanalysis
  - [Shamir 1984],[Lagarias,Odlyzko 1985]
- Several variants kept being suggested for almost two decades, but invariably broken
  - “The Rise and Fall of Knapsack Cryptosystems” [Odlyzko 1990]
- Turning point [Ajtai 1996]
  - worst-case/average-case connection
  - the “right” way to use knapsack/lattices for cryptography

Subset-Sum vs Lattice Problems

- Subset-sum over vectors $a_i = \begin{bmatrix} 1 \\ 4 \\ \vdots \\ 5 \end{bmatrix} \in Z^n$

- Essentially the same as the knapsack problem, just more convenient in cryptography applications

$$\begin{bmatrix} 4 \\ 1 \\ 6 \\ 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 8 \\ 1 \\ 7 \\ 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 12 \\ 2 \\ 13 \\ 5 \\ 6 \end{bmatrix}$$



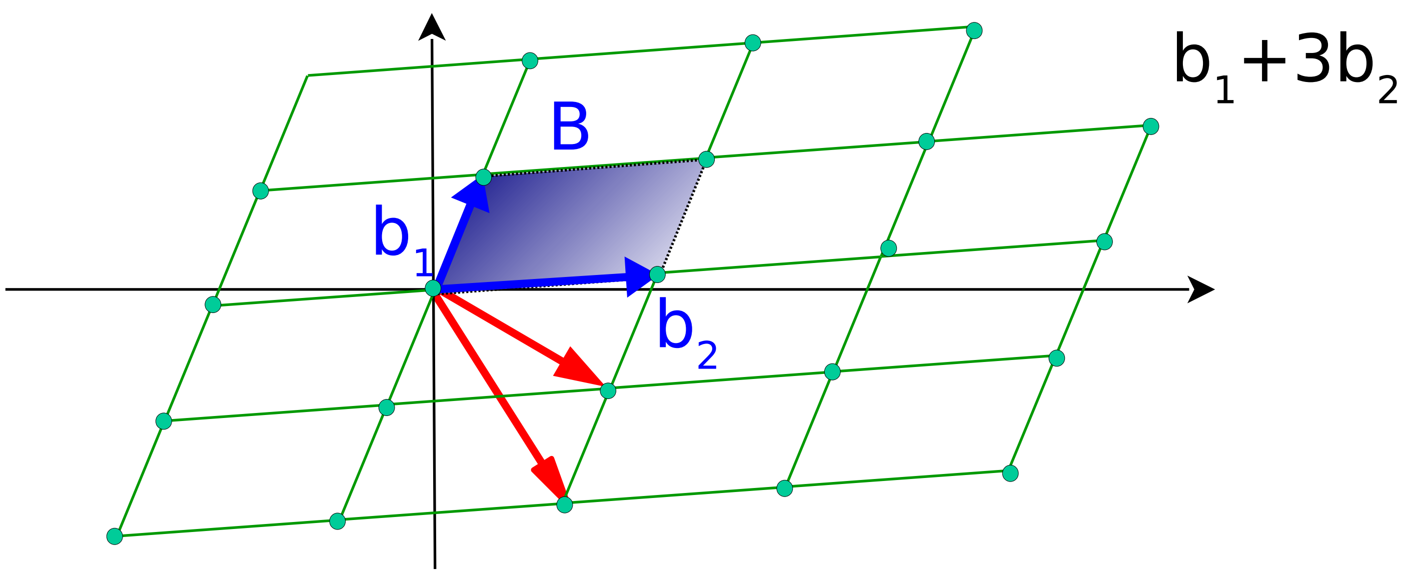
$$b = a_2 + a_4 + a_5 + a_6 + a_8$$

linear combination with small coefficients

Geometry of Lattices

Set of all integer linear combinations of basis vectors $B = [b_1,\dots,b_n] \subset R^n$

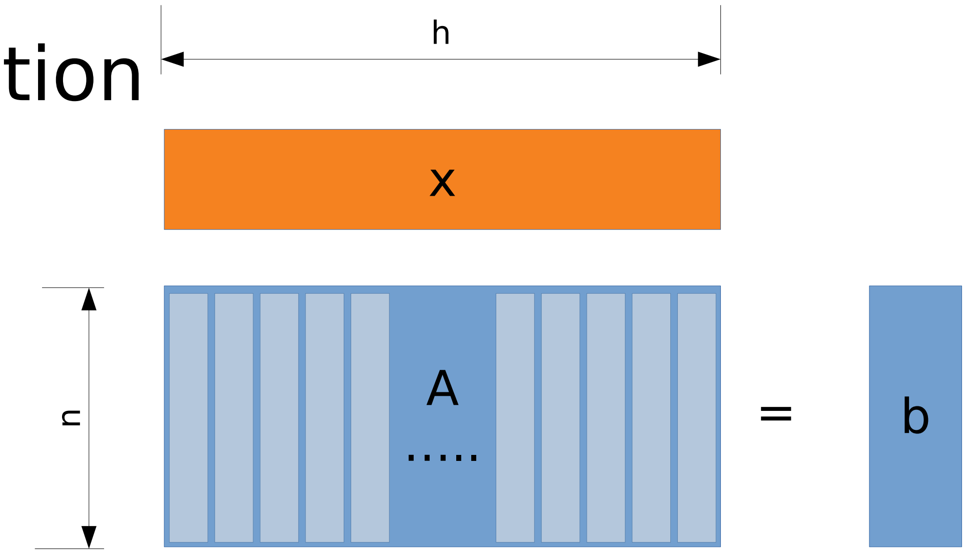
$L(B)=\{Bx: x \in Z^n\} \subset \text{span}(B)=\{Bx: x \in R^n\}$

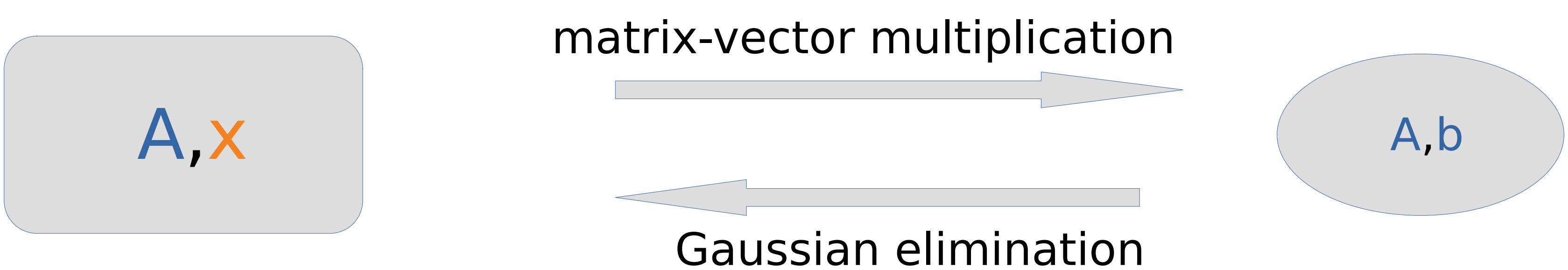


Linear functions

Matrix-Vector multiplication

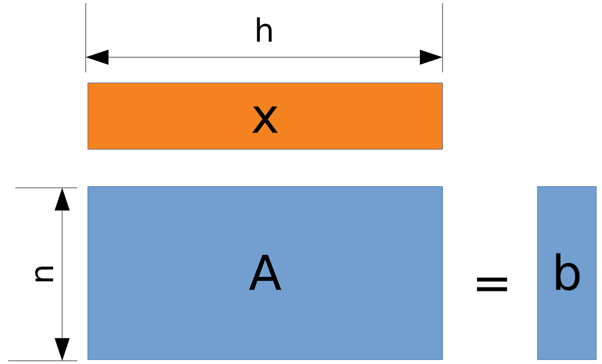
- $A \in Z_q^{n\text{x}h}$, $x \in Z_q^{h}$, $b \in Z_q^{n}$
- $f_A(x) = Ax$
- $f_A(x+y) = f_A(x) + f_A(y)$
- Easy to compute and invert



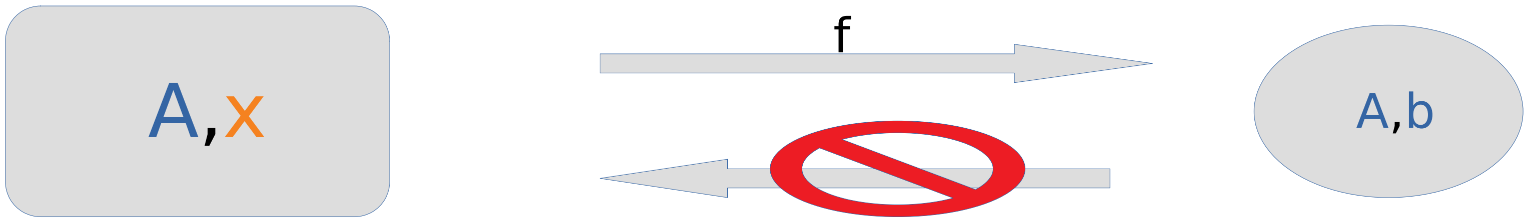


Short Integer Solution (SIS)

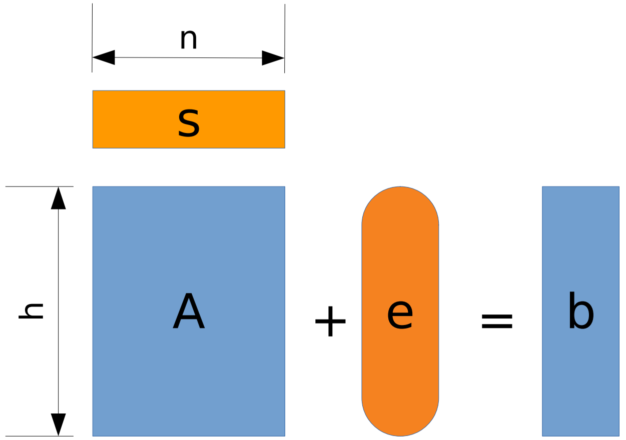
- [Ajtai 1996] One-Way Function:
  - $f_A(x) = Ax \pmod q$
  - $A \in Z_q^{n\mathrm{x}h}$, $x \in \{0,1\}^h$, $b \in Z_q^n$



- Short Integer Solution Problem:
  - Given [A,b] find a **<u>small</u>** x such that Ax=b
  - More generally, $||x|| < \beta$



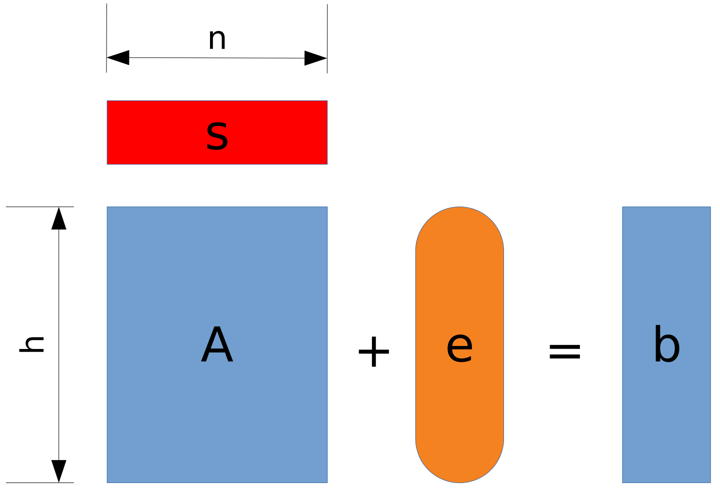
Learning With Errors (LWE)



- LWE function family:
  - Key: $A \in Z_q[n\text{x}h]$
  - $\text{LWE}_A(s,e) = As + e \pmod q$
  - Small $|e|_{max} < \beta = O(\sqrt{n})$
  - q,m=poly(n)
  - Injective version of Ajtai's SIS function
- [Regev 2005] assuming quantum hard lattice problems
  - LWE_A is one-way: Hard to recover (s,e) from [A,b]
  - $b=\text{LWE}_A(s,e)$ is **pseudorandom** ($\approx$ uniform over $Z_q[h]$)
  - [Peikert 2009], [BLPRS13] hard under *classical* reductions

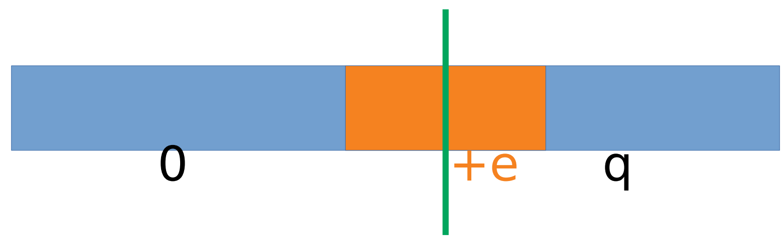
Encrypting with LWE

- Idea: Use b=LWE_A(s,e) as a one-time pad
- Private key encryption scheme:
  - secret key: $s \in Z_q^n$,
  - message: $m \in Z$
  - encryption randomness: [A,e]
  - Enc_s(m; [A,e]) = [A,b+m]
- [BFKL93],[GRS08]
  - Learning Parity with Noise (LPN): q=2
  - If LWE_A is one-way, then b=As+e is pseudo-random
- Regev LWE: q → poly(n)

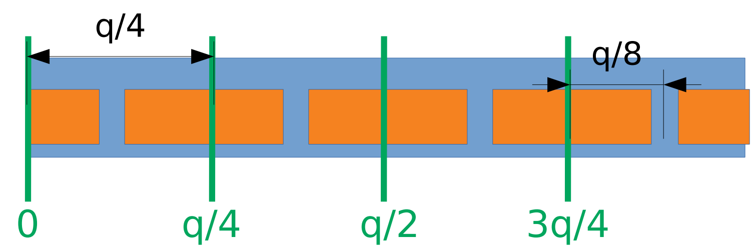


Decryption

- $\mathrm{Enc}_s(m;[A,e]) = [A, b+m]$ where $b = As+e$
- Decryption:
  - $\mathrm{Dec}_s([A, b+m]) = (b+m) - As = m+e \bmod q$



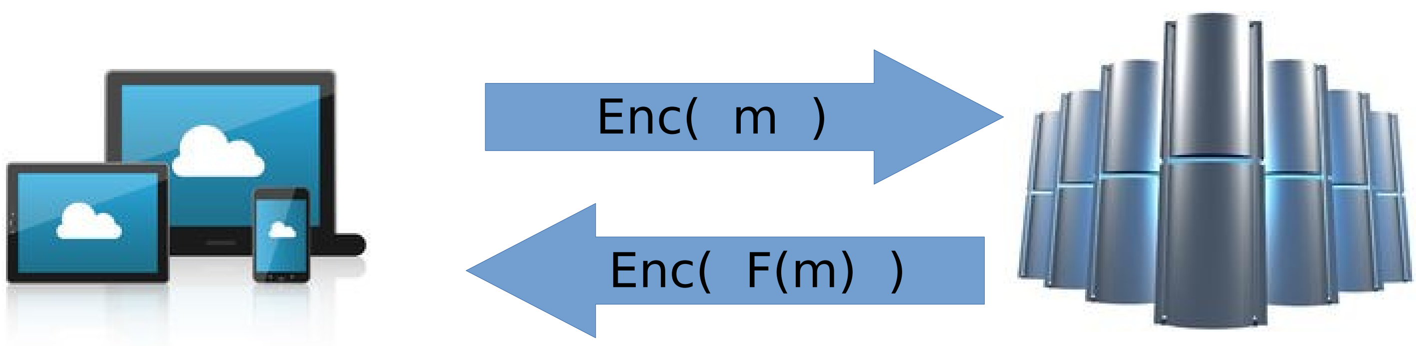
  - Low order bits of m are corrupted by e
- Fix: scale m, and round:



(Fully) Homomorphic Encryption

- Encryption: used to protect data at rest or in transit

- Fully Homomorphic Encryption: supports arbitrary computations on encrypted data



FHE Timeline

- Concept originally proposed by
  [Rivest, Adleman, Dertouzos 1978]
- [Gentry 2009]
  - First candidate solution
  - Bootstrapping technique
- Much subsequent work (2010-2020 ...)
  - Basing security on standard (lattice) assumptions
    [BV11,B12,AP13,GSW13,BV14,...]
  - Efficiency improvements
    [GHS12,BGH13,AP13/14,DM15,CP16,CGGI16/17,CKKS17,BDF18,MS18,...]
  - Implementations:
    HElib, SEAL, PALISADE, FHEW, TFHE, HEAAN, Λολ, NFLlib, ...

Homomorphic Addition

$$\mathrm{Enc}_s(m_1) \quad + \quad \mathrm{Enc}_s(m_2)$$

$$= [A_1, A_1 s + e_1 + m_1] + [A_2, A_2 s + e_2 + m_2]$$

$$= [(A_1 + A_2), (A_1 + A_2) s + (e_1 + e_2) + (m_1 + m_2)]$$

$\mathrm{Enc}_s(m;\beta)$: encryption of m with error $|e| < \beta$

- $\mathrm{Enc}_s(m_1;\beta_1) + \mathrm{Enc}_s(m_2;\beta_2) \subset \mathrm{Enc}_s(m_1 + m_2;\beta_1 + \beta_2)$
- $c * \mathrm{Enc}_s(m_1;\beta_1) \subset \mathrm{Enc}_s(c * m_1; c * \beta_1)$

Can take any linear comination of ciphertexts with small coefficients

Multiplication by any constant

- Enc'[m] = (Enc[m],Enc[2m],Enc[4m],…,Enc[$2^{\log(q)}$m])
- Multiplication by $c \in Z_q$:
  - Write $c = \Sigma_i\, c_i\, 2^i$, where $c_i \in \{0,1\}$
  - Compute $\Sigma_i\, c_i$ Enc[2^i m] = Enc[$\Sigma_i\, c_i\, 2^i$ m] = Enc[cm]
- c*Enc'[m] = Enc[cm]
- We can also compute Enc'[cm]:

  c*Enc'[m]=(cEnc'[m], (2c)Enc'[m], .., ($2^{\log q}$c)Enc'[m])

  = (Enc[cm], Enc[(2c)m], .., Enc[($2^{\log q}$c)m])

  = Enc'[cm]

Public Key Encryption

- Public Key:

  $[A_1,b_1]$ = $\mathrm{Enc}_s(0)$, …, $[A_n,b_n]$=$\mathrm{Enc}_s(0)$

- Encrypt(m): $(\Sigma_i\ r_i * [A_i,b_i]) + (O,m)$
  - $\mathrm{Enc}_s(0)+\ldots+\mathrm{Enc}_s(0)+\mathrm{Enc}_s(m;0) = \mathrm{Enc}_s(m)$
- Decrypt normally using secret key
- [Regev05] LWE Public Key Encryption
- [Rothblum11]: any linear homomorphic encryption implies public key encryption

Homomorphic Multiplication?

- Is it possible to multiply two ciphertexts?
  - $Enc_s(m_1;\beta_1) * Enc_s(m_2;\beta_2) \subset Enc_s(m_1 * m_2; B(\beta_1, \beta_2))$
- Any computation can be expressed in terms of addition and multiplication
  - 0: False, 1: True
  - $1\text{-}x = Not(x)$
  - $x * y = x \wedge y$
  - $x + y - x * y = x \vee y$

How to multiply two ciphertexts

- Linearity allows to multiply ciphertexts!
- Several multiplication methods:
  1) Encryption Nesting [2008 …]
  2) Ciphertext Tensoring [2011 …]
  3) Homomorphic Decryption [2013 …]
  4) Gate Bootstrapping [2015 …]
- Notes:
  - Main difference between FHE schemes
  - Only allows a bounded number of multiplication
  - Basic Multiplication + Bootstrapping = FHE

(1) Homomorphic Multiplication by Encryption Nesting

Multiplication by Encryption Nesting

- $C_0 = Enc_{S0}(m_0)$, $C_1 = Enc_{S1}(m_1)$
- Multiply C_0 homomorphically by C_1
  - $Enc_{S0}(m_0)*C_1 = Enc_{S0}(m_0*C_1)$
  - But $m_0*C_1 = m_0*Enc_{S1}(m_1) = Enc_{S1}(m_0*m_1)$
  - So, end result is $Enc_{S0}(Enc_{S1}(m_0*m_1))$
- Decrypt by applying Dec_{S0} and then Dec_{S1}
- $(Enc_{S0} \,.\, Enc_{S1})$ is still linearly homomorphic
  - Nested encryptions still support homomorphic addition
- Extends to more multiplications $(Enc_{S0}(Enc_{S1}(Enc_{S2}(m_0*m_1*m_2))))$, etc.

Multiplication by Encryption Nesting

- Omitted several important details:
  - Can only multiply by “small” constant C_1
  - Can only left-multply by constants
  - Ciphertexts get bigger, requiring |S0|>|S1|
- [Aguilar Melchor, Gaborit, Herranz, 2010]
  - Can only multiply ciphertexts in sequence
  - Limited (sublinear) number of multiplications
  - Not enough to support bootstrapping

(2) Multiplication by

Ciphertext Tensoring

Trivial (Symbolic) Multiplication

- Symbolic homomorphic product
  - $Enc(m_0)*Enc(m_1) = (\text{“}*\text{”}, Enc(m_0), Enc(m_1))$
- Decryption(“*”,C_0,C_1)
  - Decrypt $C_0 \rightarrow m_0$
  - Decrypt $C_1 \rightarrow m_1$
  - Compute $m_0 * m_1$
- Applies to arbitary operations
- Trivial, uninteresting
  - Ciphertext and Decryption grow with computation
  - Compactness: decryption of $f(Enc(m))$ should be sublinear in $|f|$

Trivial (Symbolic) Multiplication

- Symbolic homomorphic product
  - $Enc(m_0)*Enc(m_1) = (“*”, Enc(m_0), Enc(m_1))$
- $C = (“*”, C_0, C_1)$ allows to compute **<u>any function</u>** of C_0 and C_1
- This seems unnecessary
  - all we want to do is to decrypt C
  - enough to compute **<u>decryption function</u>** on C
  - what does the decryption function look like?

Decryption is linear

- $\mathrm{Dec}_s(A,b) = b - As = m+e$
- Decryption is linear a linear function of the ciphertext $C=(A,b)$
- Remark:
  - Only approx. decryption is linear
  - Exact decryption involves non-linear rounding
- $\underline{\mathrm{Dec}_s(C_0)*\mathrm{Dec}_s(C_1)}$ is bilinear in C_0, C_1

Multiplication by Tensoring

- Tensor product of C_0, C_1:
  - $\{C_0[i]*C_1[j] : i,j = 1..n\}$
  - allows to compute any bilinar function of C_0 and C_1
  - still an additive group, so tensor ciphertexts can be added homomorphically
- Several optimizations are possible:
  - No need to compute arbitrary bilinear functions
  - Only bilinear functions of the form

    $(C_0,C_1) \to \mathrm{Dec}(C_0) * \mathrm{Dec}(C_1)$
  - Can use a low rank subspace of tensor product

Multiplication by Tensoring

- $C_0 = \mathrm{Enc}_{S0}(m_0)$, $C_1 = \mathrm{Enc}_{S1}(m_1)$
- Product $C = C_0 * C_1 = C_0 \times C_1$ (tensor product)
- [Brakerski, Vaikuntanathan 2011]
  - C is larger than C_0, C_1
  - Only limited number of multiplications
  - Also introduces a "key switching" technique that allows to reduce the size of ciphertext
  - Support "bootstrapping", leading to a FHE

Tensoring and Key Switching

- Decryption: $\mathrm{Dec}_s(A,b) = b - As \approx m$
  - Linear *<u>in the secret key</u>* $s' = (-s, 1)$
  - $\mathrm{Dec}_{s'}(A,b) = [A,b]s' \approx m$
- Given two ciphertexts $c_1\ c_2$:
  - $\langle c_1 , s' \rangle \approx m_1$
  - $\langle c_2 , s' \rangle \approx m_2$
  - $\langle c_1 \times c_2, s' \times s' \rangle = \langle c_1, s' \rangle \langle c_2, s' \rangle \approx m_1\ m_2$
- $c_1 \times c_2$ is an encryption of $m_1 m_2$ w.r.t. $s' \times s'$

Key Switching

- Key Switching: $c = \mathrm{Enc}_s(m) \rightarrow c' = \mathrm{Enc}_t(m)$
- Linear decryption: $\mathrm{Dec}_s(c) = \langle c, s \rangle \approx m$
- Linear Homomorphism:
  - $\langle c, \mathrm{Enc}_t(s) \rangle = \mathrm{Enc}_t(\langle c, s \rangle) \approx \mathrm{Enc}_t(m)$
- $\mathrm{Enc}_t(s)$ allows to switch key: $\mathrm{Enc}_s(m) \rightarrow \mathrm{Enc}_t(m)$
- Multiplication by tensoring:
  - $t = s$ (requires circular security assumption)
  - $\langle c_1 \times c_2, s \times s \rangle \approx m_1 m_2$
  - $\langle c_1 \times c_2, \mathrm{Enc}_s(s \times s) \rangle = \mathrm{Enc}_s(m_1 m_2)$

(3) Multiplication by

Homomorphic Decryption

Decryption is linear

- $\mathrm{Dec}_s(A,b) = b - As = m+e$
- Linear in the ciphertext (A,b)
- Linear *in the secret key* $s'=(-s,1)$
  - $\mathrm{Dec}_{s'}(A,b) = [A,b]s' = m+e$
  - $\mathrm{Dec}_{cs'}(A,b) = [A,b](cs') = cm+ce$
- Remark:
  - Only approx. decryption is linear
  - Exact decryption involves non-linear rounding

Multiplication via Homomorphic Decryption

- Idea:
  - Encryption $Enc(m) = (A, As+e+m)$ is linearly homomorphic
  - Decryption $Dec(A,b) = b – As = m+e$ is linear in $s'=(-s,1)$
  - We can decrypt homomorphically using an encryption of s'
- Details
  - Given: $Enc(m)=(a,b)$ and $Enc(s')=(Enc(-s),Enc(1))$
  - Compute $Enc(m)*Enc(s') = a*Enc(-s)+b*Enc(1)=Enc(m)$
- More interesting:
  - Given $Enc(m)$ and $Enc(cs')$
  - Compute $Enc(m)*Enc(cs') = Enc(cm)$

Homomorphic "decrypt and multiply"

- Enc''(c) = Enc'(cs') = Enc'("E(m)→c*m")
- Enc''(c) = {Enc($\alpha_i c$)}$_i$ for some α_i(s)
- Homomorphic Properties:
  - Enc''(m_1) + Enc''(m_2) = Enc''(m_1+m_2)
  - Enc''(m_1)*Enc''(m_2)

    ={Enc($\alpha_i m_1$)*Enc''(m_2)}$_i$

    ={Enc($\alpha_i m_1 * m_2$)}

    = Enc''($m_1 * m_2$)

Relation to GSW encryption

- [Gentry,Sahai,Waters'13]
  - FHE based on "approximate eigenvectors" intuition
  - $C_1 = \mathrm{Enc}_s(m_1)$, $C_2 = \mathrm{Enc}_s(m_2)$
  - $C_1 * s \approx m_1 * s$, $C_2 * s \approx m_2 * s$
  - $(C_1 * C_2) * s \approx C_1 * (C_2 * s)$

    $\approx C_1 * (m_2 * s) \approx m_2 * (C_1 * s) \approx (m_1 m_2) * s$
  - $C_1 * C_2 \approx \mathrm{Enc}_s(m_1 m_2)$
- GSW vs Enc''(m)
  - conceptually different
  - technically equivalent:

    perform essentially the same operations

(4) Homomorphic Multiplication by

Gate Bootstrapping

Bootstrapping and FHE

- Encryption scheme supporting
  - $Enc(m_0)+Enc(m_1) = Enc(m_0+m_1)$
  - $Enc(m_0)*Enc(m_1) = Enc(m_0*m_1+e)$
- Not quite a FHE yet:
  - Enc can evaluate any arithmetic circuit
  - But noise grows with computation
- Effectively:
  - can only evaluate small circuits / branching programs
- Bootstrapping: technique to redude e by homomorphic decryption
  - [Gentry 2009] FHE(Dec) → FHE(PTIME)

Bootstrapping

- Refresh: Enc(s,m;$q/8$) → Enc(s,m;$q/16$)
- Consider the function $f_c(s)$=Dec(s,c)
- Compute f_c homomorphically on [s]=Enc(s,s ; **e**)
  - c = Enc(s,m;$q/8$), [s] = Enc(s,s ; **e**)
  - f_c([s])= [f_c(s)] = [Dec(s,c)] = [m] =Enc(s,m)
- [m]=Enc(s,m;**e'**) where **e'** depends only on **e** and f_c.
- Setting **e'**<$q/16$:

  Enc(m_1;$q/16$) + Enc(m_2;$q/16$) = Enc(m_1+m_2;$q/8$)
  → Enc(m_1+m_2;$q/16$)
- Can perform any number of additions!

FHEW: gate bootstrapping

- Bootstrapping:

  Enc(s,m;q/8) → Enc(s,m;q/16)
- [Ducas, Micciancio, 2015]
  - Use arithmetics modulo 4
  - Bootstrapping + Compute:

    Enc(s,m;q/8) → Enc(s,floor(m/2);q/16)
- Enough to compute arbitrary circuits:
  - $m_1, m_2 \in \{0,1\} \subset Z_4 = \{0,1,2,3\}$
  - MUL(m_1,m_2) = floor((m_1+m_2)/2)
  - NOT(m) = 1-m
- Cannot do this working directly mod 2
  - All unary gates mod 2 (0,1,id,not) are linear!

m_1	m_2	m_1+m_2	sum/2
0	0	0	0
0	1	1	0
1	0	1	0
1	1	2	1

Many other FHE variants

- Optimizations: [GHS12],[BGV12],[B/FV12] ...
- TFHE,HEAAN [CGGI16,17], [CKKS17]
- Bootstrapping algorithms: [AP13,BV14,AP14,GINX16,...]
- Libraries: HElib, SEAL, PALISADE, LoL, ...
- All share similar ideas, building blocks, techniques
- Complexity of bootstrapping still main efficiency bottleneck

Summary

- Lattice Based cryptography
  - Post-quantum security
  - Homomorphic addition
- Can also multiply ciphertexts
  - FHE: arbitrary computations on encrypted data
- **Active research area**
  - Efficiency
  - Circular security:
    - can $Enc_s(sxs)$ be safely revealed?

Additional References

- [BFKL93] Blum,Furst,Kearns,Lipton
- [GRS08] Gilbert,Robshaw,Seurin
- [BV11,14] Brakerski, Vaikuntanathan
- [GHS12] Gentry, Halevi, Smart
- [BGV12] Brakerski,Gentry,Vaikuntanathan
- [B/FV12] Brakerski / Fan,Vercauteren
- [BLPRS13] Brakerski,Langlois,Peikert,Regev,Stehle
- [AP13,14] Alperin-Sherif, Peikert
- [GINX16] Gama, Izabachene, Nguyen, Xie
- [CGGI16/17] Chilotti,Gama, Georgieva, Izabachene
- [CKKS17] Cheon,Kim,Kim,Song

Thank You!

Questions?