Secure Genomic Computation

Kristin Lauter

Cryptography Research Group

Microsoft Research

iDASH Secure Genome Analysis Competition

March 16, 2015

iDASH Privacy & security workshop 2015
Secure genome analysis Competition

- Registration: Jan 31 2015
- Submission deadline: Feb 28 2015
- Workshop: March 16, 2015
  UCSD Medical Education and Telemedicine Building ROOM 141/143
- Media coverage in GenomeWeb, Donga Science, Nature

Donga Science, March 13, 2015

○ **MS 연구진 이끌고 DNA 보안 알고리즘 개발**

이 연구원과 같은 연구실에서 한솥밥을 먹고 있는 김미란 연구원(28)은 생체정보 보안 연구 분야에서 떠오르는 샛별이다. 그는 1월 미국 마이크로소프트(MS) 연구소 초청으로 현지에 급파됐다. 작년 내내 MS 연구진을 이끌고 개발한 DNA 보안 기술이 '안전 게놈 분석 경진대회(Secure Genome Analysis Competition)'에 출전했기 때문이다. 이 대회는 샌디에이고 캘리포니아대 의대가 지난해부터 개최하는 첨단 생체정보 보안 대회다.

http://news.donga.com/It/3/all/20150313/70100744/1

GenomeWeb, Nature, …

Why the excitement?

Fundamental Problem: privacy protection

- Burgeoning genome sequencing capability
- Explosion of scientific research possible
- High risk for personal privacy

Fundamental Progress through interaction

- Computer Scientists
- Mathematicians
- Bioinformaticians
- Policy-makers

Data Breaches: Privacy Rights Clearinghouse

- 815,842,526 RECORDS BREACHED
  from 4,495 DATA BREACHES made public since 2005

January 5, 2015	**Morgan Stanley New York, New York**	BSF	INSD	350,000

An employee of Morgan Stanley stole customer information on 350,000 clients including account numbers. Additional information on what other information was captured has not yet been released. Files for as many as 900 clients ended up on a website.

January 6, 2015	**NVIDIA Corporation Santa Clara, CA**	BSO	HACK	Unknown

NVIDIA Corporation suffered a data breach when hackers infiltrated their network and stole employee usernames and passwords.
The company is requesting that those affected change their password and be cautious of "phishing" emails that look like they are coming from a colleague or friend requesting sensitive information.

Data access and sharing requirements

- Allow access to researchers to large data sets
- Secure Genome Wide Association Studies (GWAS)
- Desire for centrally hosted, curated data
- Provide services based on genomic science discoveries

Two scenarios for interactions:

- Single data owner (one patient, one hospital)
- Multiple data owners (mutually distrusting)

Two Challenges!

Challenge 1:

Homomorphic encryption (HME) based secure genomic data analysis

- **Task 1: Secure Outsourcing GWAS**
- **Task 2: Secure comparison between genomic data**

Challenge 2:

Secure multiparty computing (SMC) based secure genomic data analysis (two institutions)

- **Task 1: Secure distributed GWAS**
- **Task 2: Secure comparison between genomic data**

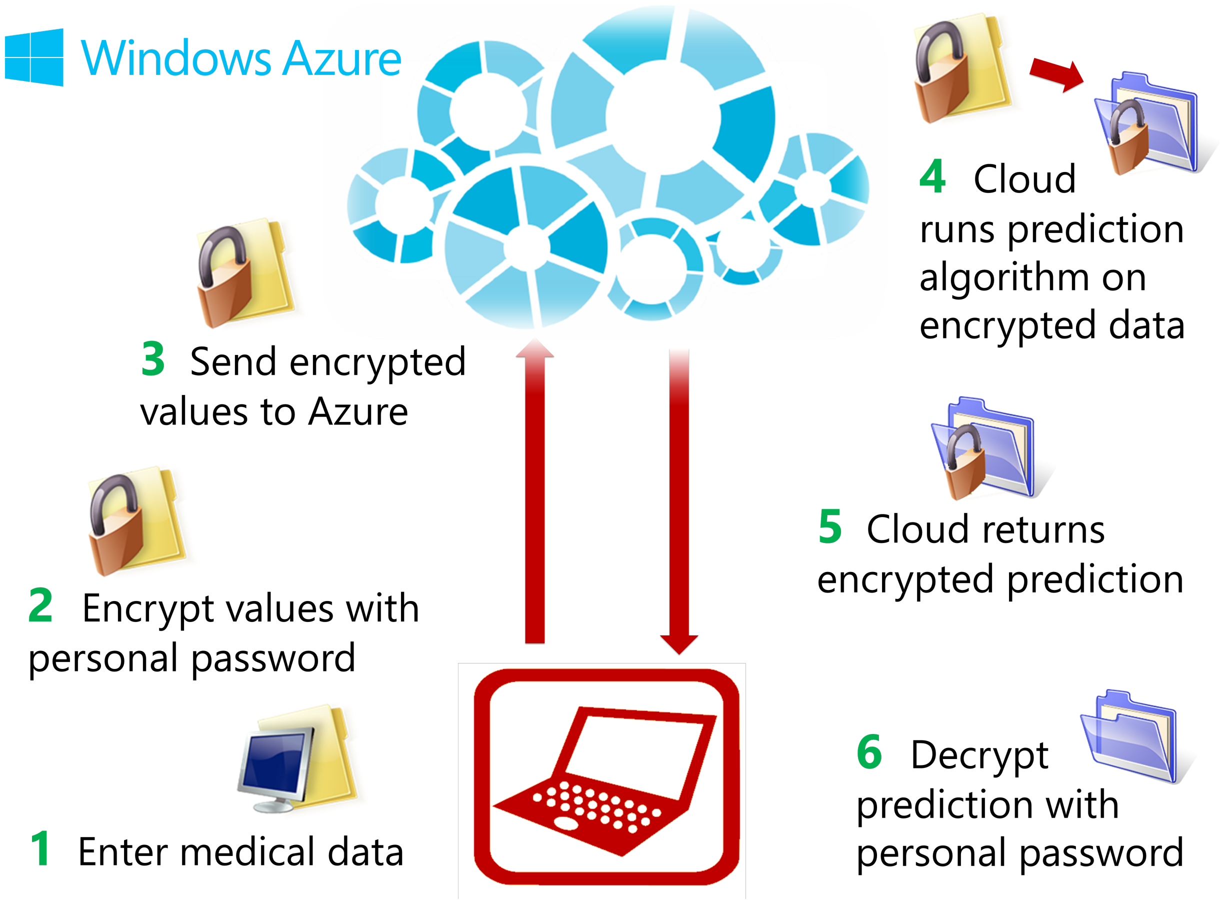
Private cloud services

Preserve privacy through encryption! (clients keep the keys!)

Scenarios:

- Direct-to-patient services
  - Personalized medicine
  - DNA sequence analysis
  - Disease prediction
- Hosted databases for enterprise
  - Hospitals, clinics, companies
  - Allows for third party interaction

Outsourcing computation



Scenario for genomic data

Untrusted cloud service

Stores, computes on encrypted data

Trusted party

hosts data and regulates access

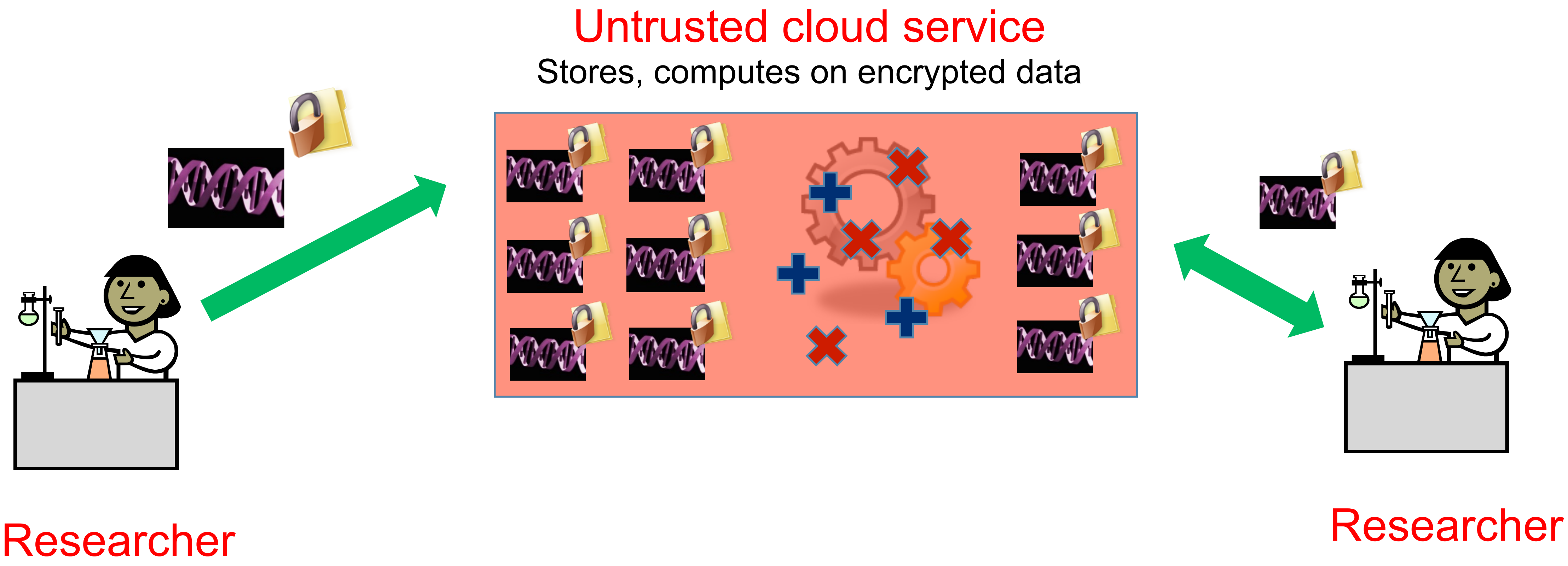
Requests for decryption of results

(requires a policy)

Researcher:

requests encrypted results of specific computations

Multi-party computation for genomic data



Techniques:

- Homomorphic Encryption
  - Paillier encryption (additive operations)
  - Lattice-based encryption (additions and multiplications)

- Multi-party Computation
  - Optimized Garbled Circuits
  - Secret Sharing techniques

What are the Costs? Challenges? Obstacles?

For homomorphic encryption

- Storage costs (large ciphertexts)
- New hard problems (introduced 2010-2015)
- Efficiency at scale (large amounts of data, deep circuits)

For Garbled Circuits

- High interaction costs
- Bandwidth use
- Integrate with storage solutions

What kinds of computation?

- Building predictive models
- Predictive analysis
  - Classification tasks
  - Disease prediction
  - Sequence matching
- Data quality testing
- Basic statistical functions
- Statistical computations on genomic data

Encrypt everything?

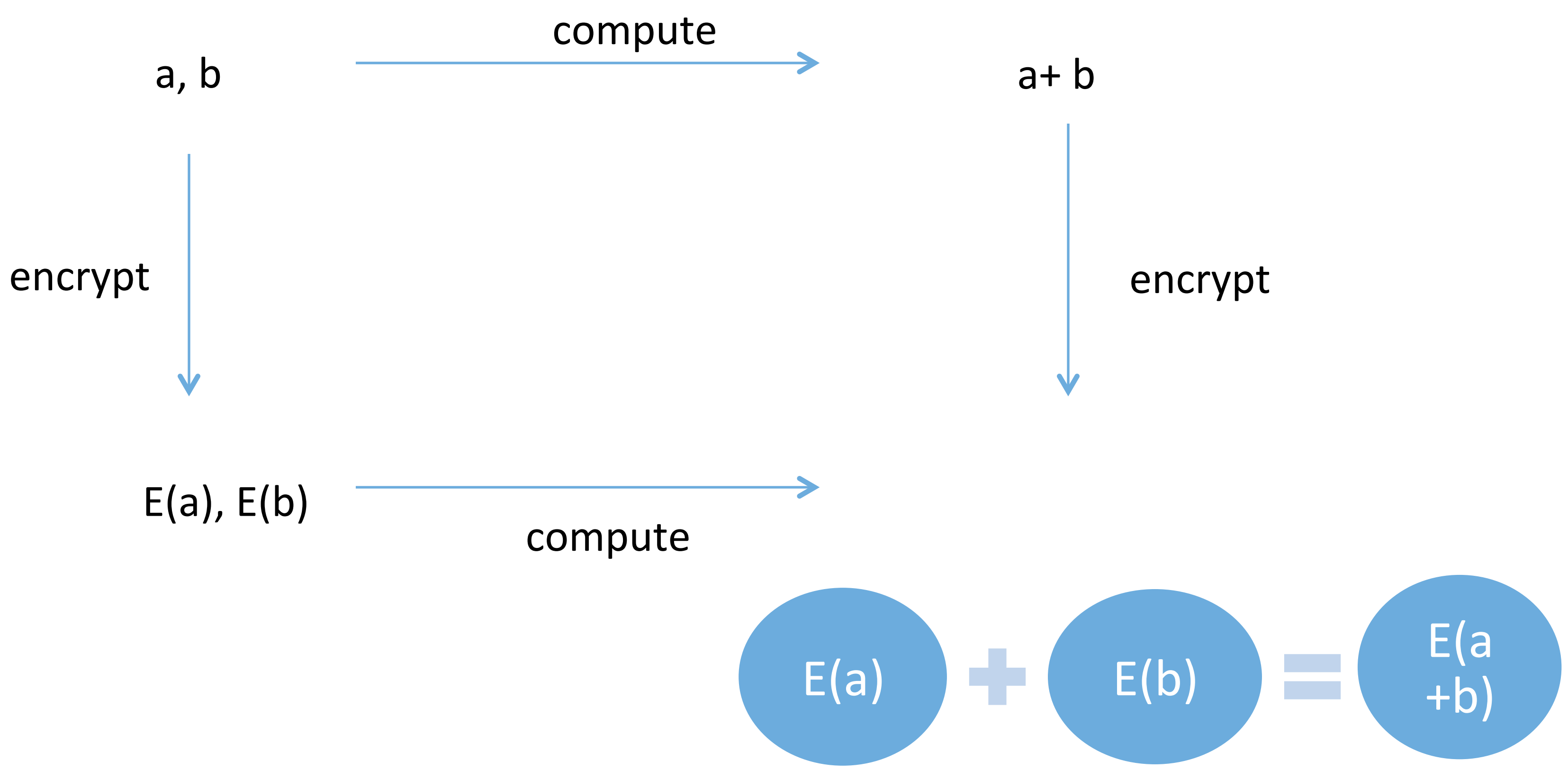
- Protect outsourced data by encrypting everything
- “Conventional” encryption methods do not allow any computation on the encrypted data without using the secret key and decrypting it
- Homomorphic encryption schemes allow specific operations on encrypted data with only public information

Protecting Data via Encryption

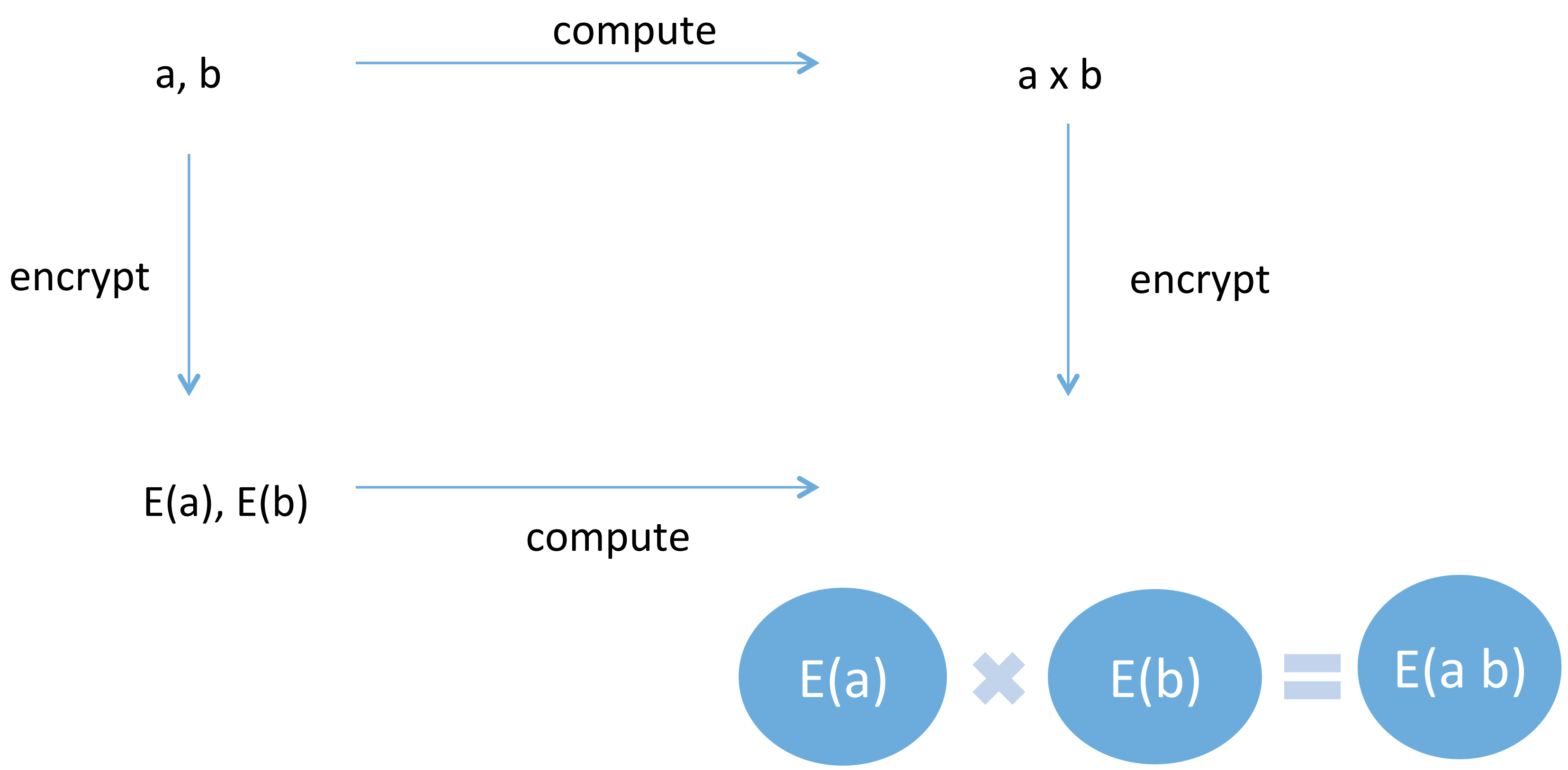
Homomorphic encryption

1. Put your gold in a locked box.
2. Keep the key.
3. Let your jeweler work on it through a glove box.
4. Unlock the box when the jeweler is done!

Homomorphic Encryption: addition

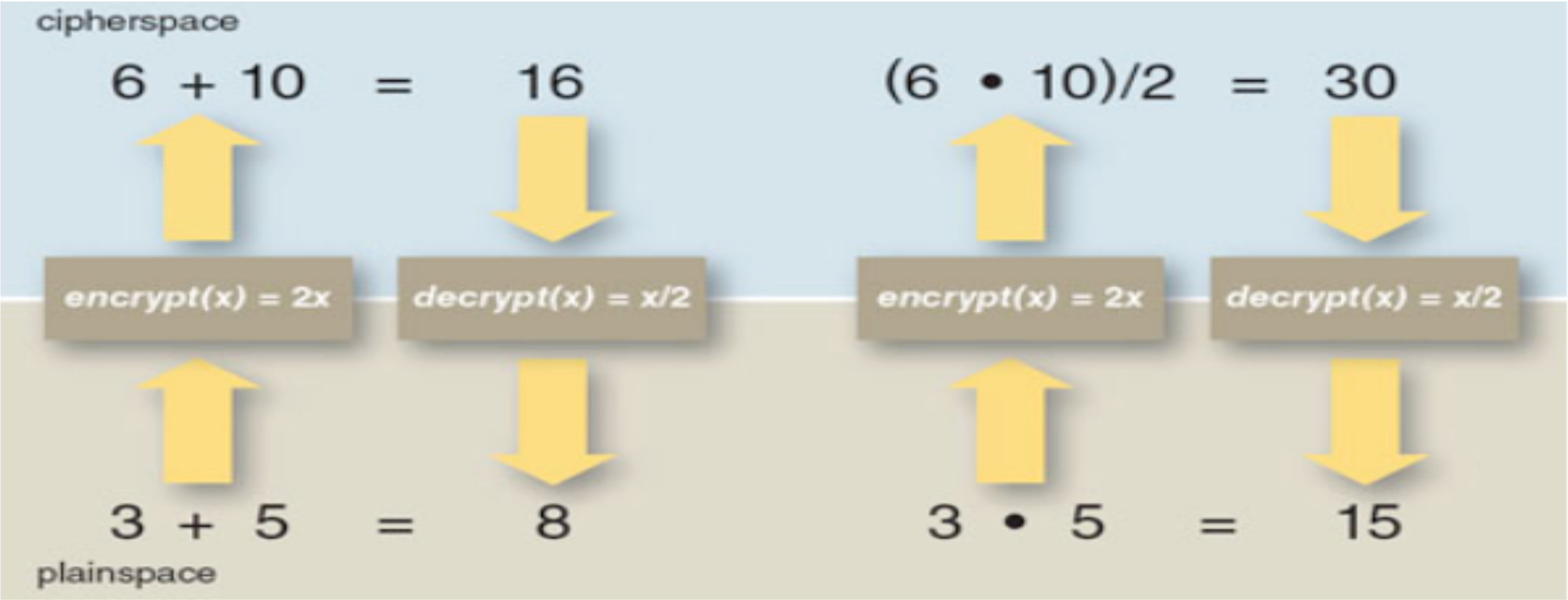


Homomorphic Encryption: multiplication



Operating on encrypted data

"Doubly" homomorphic encryption



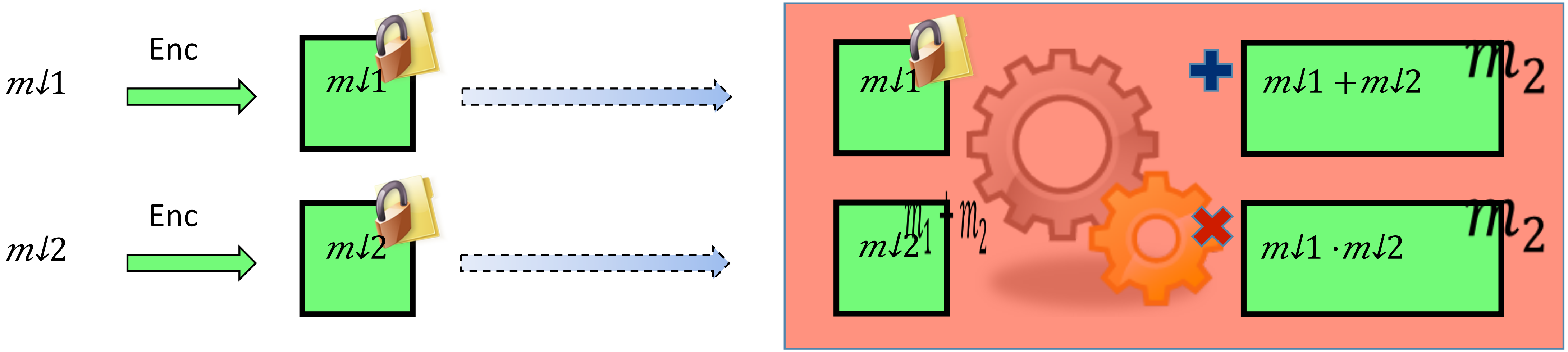
American Scientist, Sept/Oct 2012

Fully Homomorphic Encryption (FHE)

FHE enables unlimited computation on encrypted data

- Public operations on ciphertexts:

$$(Enc(m_1), Enc(m_2)) \rightarrow \mathrm{Enc}(m_1 + m_2)$$
$$(Enc(m_1), Enc(m_2)) \rightarrow \mathrm{Enc}(m_1 \cdot m_2)$$



Fully Homomorphic Encryption (FHE)

FHE enables unlimited computation on encrypted data

- Public operations on ciphertexts:

$$(Enc(m_1), Enc(m_2)) \rightarrow \mathrm{Enc}(m_1 + m_2)$$
$$(Enc(m_1), Enc(m_2)) \rightarrow \mathrm{Enc}(m_1 \cdot m_2)$$

- For data encrypted bitwise ($m_1, m_2 \in \{0,1\}$), operations $m_1 + m_2$ and $m_1 \cdot m_2$ are bitwise (XOR and AND)

- Get arbitrary operations via binary circuits.

Fully Homomorphic Encryption

[BGN05] – unlimited addition + 1 multiplication (pairing-based)

[Gentry09] first scheme with unlimited additions and multiplications

Impractical!

Much progress since then…

FHE Schemes

- **Small Principal Ideal Problem (SPIP)**
  - Gen’09, SV’10, GH’11
- **Approximate GCD**
  - vDGHV’10, CMNT’11, CNT’12, CCKLLTY’13
- **LWE/RLWE**
- BV’11a, BV’11b, BGV’12, GHS’12,LTV’12, Bra’12, FV’12, BLLN’13

Compare to other public key systems:

RSA (1975), ECC (1985), Pairings (2000)

HElib (IBM) publically available implementation

FHE schemes do exist!

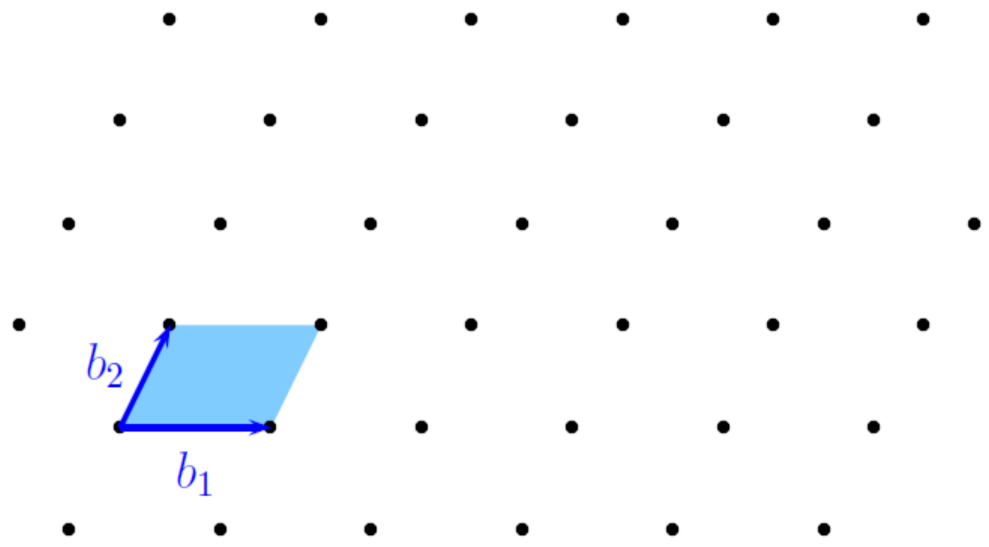
- BUT FHE on binary circuits with bitwise encryption is extremely inefficient:

    - huge ciphertexts,
    - costly noise handling,
    - large overhead in storage space and computation time

Lattice-based Crypto

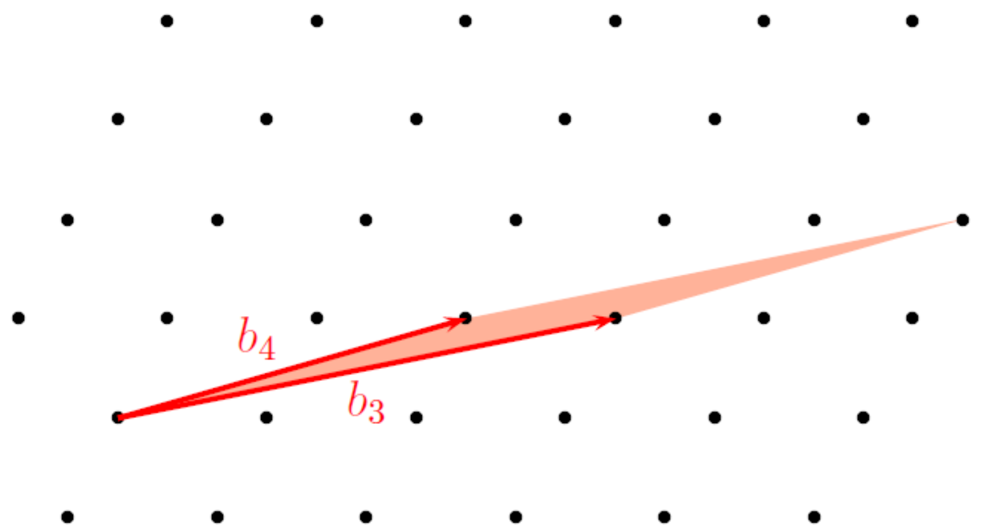
- Alternative Public Key Crypto
  - RSA, Diffie-Hellman, ECC, Pairings, …
- SECURITY:
  - best attacks take exponential time
  - secure against quantum attacks (so far…)
- Hard Problems:
  - approximate SVP (in the worst case) on ideal lattices in R
  - search version of Ring-based Learning With Errors (R-LWE)
  - Further reductions: D-RLWE, PLWE

Lattice with a Good (short) Basis



$L = \mathbb{Z}b_1 + \mathbb{Z}b_2$

Lattice with a Bad Basis



$$L = \mathbb{Z}b_3 + \mathbb{Z}b_4$$

Idea of new schemes

- Lattice vectors → coefficients of polynomials
- Polynomials can be added and multiplied
- Encryption adds noise to a “secret” inner product
- Decryption subtracts the secret and then the noise becomes easy to cancel
- Hard problem is to “decode” noisy vectors
- Uses a discretized version of the problem
- If you have a short basis, it is easy to decompose vectors

Ring-based Learning With Errors (R-LWE)

- Let $q \equiv 1 \bmod 2n$ be a prime, $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. $n=2^k$. Consider the polynomial ring

$$R_q = \mathbb{Z}_q[x]/(x^n + 1).$$

- Given a secret element $s \in R_q$ and a number of pairs

$$(a_i, b_i = a_i s + e_i),$$

- where $a_i \leftarrow R_q$ are chosen uniformly at random, and $e_i \leftarrow D_\sigma(R_q)$ are chosen coefficient wise according to the discrete Gaussian error distribution $D_\sigma(\mathbb{Z}_q)$.

- **R-LWE problem:** Find the secret s (search), or distinguish whether a list of pairs (a_i, b_i) was chosen as described above or whether both $a_i, b_i \leftarrow R_q$ were chosen uniformly at random (decision).

Secret-key Encryption from R-LWE

- Gen(1^n): Sample a "small" ring element $s \leftarrow D_\sigma(R_q)$.
  Secret key: sk$=s$.
- Enc(sk,m): m: encoding of message $m \in \{0,1\}^n$ as a "small" element of R_q, a is uniformly random in R_q,
  e is a "small" ring element $e \leftarrow D_\sigma(R_q)$.
  Encryption: c$=(a,\ as+2e+m)$.
- Dec(sk,(a,b)): Output $(b-as)$ mod 2.

This scheme can be turned into a *fully homomorphic encryption scheme,* that can compute any function on encrypted data.

Homomorphic Encryption

- What are the right parameters for a given security level?
- To estimate security, look at runtime of possible attacks:
  Combine lattice-basis reduction (LLL, BKZ) and bounded-distance decoding/distinguishing attacks
- Parameters with security > 128 bits for somewhat homomorphic PK scheme (strongly depends on number of multiplications)

#mult	n	size(q)	PK size	SK size	CT size
1	2048	58 bits	30 KB	2 KB	≥ 30 KB
10	8192	354 bits	720 KB	8 KB	≥ 720 KB
32	65536	1298 bits	20 MB	66 KB	≥ 20 MB

Homomorphic Encryption

- Reference implementation of somewhat homomorphic PK scheme in computer algebra system Magma
- Experimentation phase, still search for better parameters, more optimizations
- Timing for n = 2048, q has 58 bits, 1 mult

Operation	**x86-64** **Intel Core 2 @ 2.1 GHz**
SH_Keygen	250 ms
SH_Enc	24 ms
SH_Add	1 ms
SH_Mul	41 ms
SH_Dec (2-element ciphertext)	15 ms
SH_Dec (3-element ciphertext)	26 ms

Improvements and optimizations:

- Pack more data into ciphertexts [GHS12]
- Use leveled homomorphic schemes (allows limited levels)
- Use arithmetic circuits and restrict to computations with low multiplicative depth [LNV11]
- Integer encoding improvements [LNV11]

This comes at a cost: restrictions on the type of computations that can be done!

What can we compute with FHE?

Requires bit-wise encoding and encryption:

AES decryption [GHS'13], [CCKLLTY'13]

(GHS'13 uses BGV'12, CCKLLTY'13 uses Approximate GCD)

Comparison circuits

Sequence Matching: Edit distance, Smith-Waterman [CLL14]

Integer and real number encoding via bit-decomposition:

Machine Learning algorithms (real numbers) – [GLN12]

- Uses [BV11] (without relinearization, ie. ciphertexts grow)

Approximate Logistic Regression – [BLN13]

Statistics on Genomic Data –[LLN14]

Homomorphic Encryption from RLWE

- Uses polynomial rings as plaintext and ciphertext spaces

$$R = \mathbf{Z}[X]/(X^n + 1), \quad n = 2^k$$

- Work with polynomials in R modulo some $q \in \mathbf{Z}$

- Homomorphic operations (✚/ ✖) correspond to polynomial operations (add/mult) in R
- ✚ is relatively efficient, ✖ is costly

- Use this structure to encode and work with your data

Encoding real numbers

- LNV’11 Encoding - Integer **a**
  - Bit decomposition: **a** = $\sum_{i=0}^{n-1} a_i 2^i$
  - Define its encoding to be **m** = $\sum_{i=0}^{n-1} a_i x^i \in R$
  - After decryption, evaluate **m** at x=2
- GLN, BLN - Real number **b** up to precision **s**
  - Encode **10^sb** as above
  - E.g. encode **π** with precision **s=2** as

Encode(314) = $x^8 + x^5 + x^4 + x^3 + 2$

  - Need to scale computation accordingly…

“Practical Homomorphic Encryption”

- do not need *fully* homomorphic encryption
- “somewhat” does not mean *partially*
- encode integer information as “integers”
- not bit-by-bit
- several orders of magnitude speed-up
- do not need deep circuits to do a single multiplication
- do not need boot-strapping
- need to keep track of parameters to ensure correctness and security

HE Performance

80-bit security

- Parameter set I: $n=4096,\ q\approx 2^{192}$, ciphertext $\approx 100\text{KB}$
- Parameter set II: $n=8192,\ q\approx 2^{384}$, ciphertext $\approx 400\text{KB}$

Operation	KeyGen	Encrypt	Add	Mult	Decrypt
Parameters I	3.6s	0.3s	0.001s	0.05s	0.04s
Parameters II	18.1s	0.8s	0.003s	0.24s	0.26s

Proof-of-concept implementation: computer algebra system Magma,
Intel Core i7 @ 3.1GHz, 64-bit Windows 8.1

Machine Learning for Predictive Modeling

Supervised Learning

Goal: derive a function from labeled training data

Outcome: use the "learned" function to give a prediction (label) on new data

Training data represented as vectors.

Linear Means Classifier (binary)

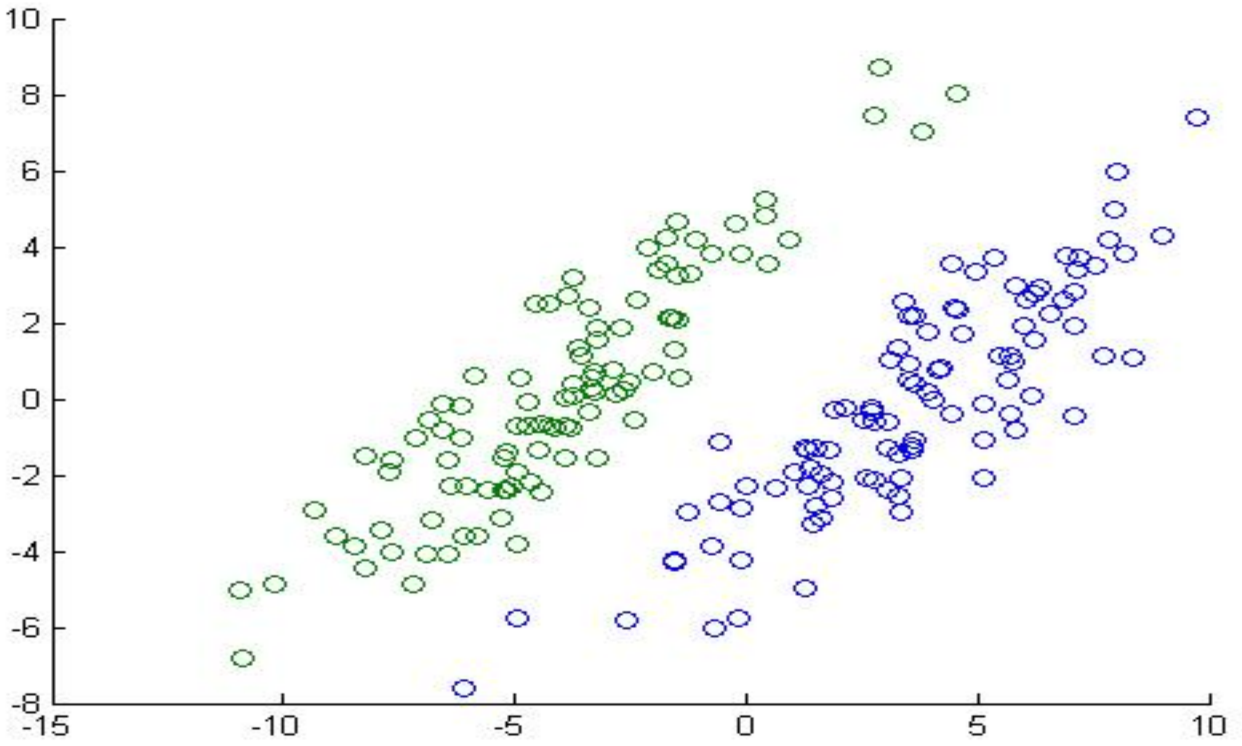
- Divide training data into (two) classes according to their label
- Compute mean vectors for each class
- Compute difference between means
- Compute the midpoint
- Define a hyperplane between the means, separating the two classes

Predictions on Medical data

Mean Compactness	Mean Concavity	Mean Concave Points	Mean Symmetry	Mean Fractal Dim.	Radius SE	Texture SE	Perimeter SE	Area SE	Smoothness SE	Compactness SE	Concavity SE	Concave Points SE	Symmetry SE	Fractal Dim. SE	Worst Radius	Worst Texture	Worst Perimeter	Worst Area	Worst Smoothness	Worst Compactness	Worst Concavity	Worst Concave Points	Worst Symmetry	Worst Fractal Dim.	Predicted Label	Truth
.05	0.00	0.00	0.17	0.06	0.54	2.93	3.62	29.11	0.01	0.01	0.00	0.00	0.03	0.00	10.49	34.24	66.50	330.60	0.11	0.07	0.00	0.00	0.25	0.07	Benign	Benign
.13	0.10	0.04	0.15	0.06	0.23	1.11	2.22	19.54	0.00	0.05	0.07	0.02	0.02	0.00	15.48	27.27	105.90	733.50	0.10	0.32	0.37	0.11	0.23	0.08	Benign	Benign
.10	0.11	0.04	0.14	0.07	0.24	2.90	1.94	16.97	0.01	0.03	0.06	0.01	0.01	0.00	12.48	37.16	82.28	474.20	0.13	0.25	0.36	0.10	0.21	0.09	Benign	Benign
.11	0.04	0.04	0.15	0.06	0.36	1.49	2.89	29.84	0.01	0.03	0.02	0.02	0.02	0.01	15.30	33.17	100.20	706.70	0.12	0.23	0.13	0.10	0.23	0.08	Benign	Benign
.04	0.00	0.00	0.11	0.06	0.31	3.90	2.04	22.81	0.01	0.01	0.00	0.00	0.02	0.00	11.92	38.30	75.19	439.60	0.09	0.05	0.00	0.00	0.16	0.06	Benign	Benign
.21	0.26	0.09	0.21	0.07	0.26	1.21	2.36	22.65	0.00	0.05	0.07	0.02	0.02	0.01	17.52	42.79	128.70	915.00	0.14	0.79	1.17	0.24	0.41	0.14	Malignant	Malignant
.22	0.32	0.15	0.21	0.07	0.96	1.03	8.76	118.80	0.01	0.04	0.08	0.03	0.02	0.01	24.29	29.41	179.10	1819.00	0.14	0.42	0.66	0.25	0.29	0.10	Malignant	Malignant
.12	0.24	0.14	0.17	0.06	1.18	1.26	7.67	158.70	0.01	0.03	0.05	0.02	0.01	0.00	25.45	26.40	166.10	2027.00	0.14	0.21	0.41	0.22	0.21	0.07	Malignant	Malignant
.10	0.14	0.10	0.18	0.06	0.77	2.46	5.20	99.04	0.01	0.02	0.04	0.02	0.02	0.00	23.69	38.25	155.00	1731.00	0.12	0.19	0.32	0.16	0.26	0.07	Malignant	Malignant
.10	0.09	0.05	0.16	0.06	0.46	1.08	3.43	48.55	0.01	0.04	0.05	0.02	0.01	0.00	18.98	34.12	126.70	1124.00	0.11	0.31	0.34	0.14	0.22	0.08	Malignant	Malignant
.28	0.35	0.15	0.24	0.07	0.73	1.60	5.77	86.22	0.01	0.06	0.07	0.02	0.02	0.01	25.74	39.42	184.60	1821.00	0.17	0.87	0.94	0.27	0.41	0.12	Malignant	Malignant
.04	0.00	0.00	0.16	0.06	0.39	1.43	2.55	19.15	0.01	0.00	0.00	0.00	0.03	0.00	9.46	30.37	59.16	268.60	0.09	0.06	0.00	0.00	0.29	0.07	Benign	Benign

Binary classification example

- FDA data set



Machine Learning on Encrypted Data

- Implements Polynomial Machine Learning Algorithms
- Integer Algorithms
- Division-Free Linear Means Classifier
- Fisher’s Linear Discriminant Classifier

DFI-LM experiments

$(P_1)\ q = 2^{128},\ t = 2^{15},\ \sigma = 16,\ d = 4096$

SH.Keygen	SH.Enc	SH.Dec(2)	SH.Dec(3)	SH.Add	SH.Mult
156	379	29	52	1	106

Timing in ms in Magma on a single core of an Intel Core i5 CPU650 @ 3.2 GHz. 128-bit security with distinguishing advantage 2^{-64}.

data	# features	algorithm	train	classify
surrogate	2	linear means	230	235
Iris	4	linear means	510	496

Statistics on Genomic Data

- **Pearson Goodness-Of-Fit Test**
  - checks data for bias (Hardy-Weinberg equilibrium)
- **Cochran-Armitage Test for Trend**
  - Determine **correlation** between genome and traits
- **Linkage Disequilibrium Statistic**
  - Estimates correlations between genes
  - **Estimation Maximization (EM) algorithm for haplotyping**

Hardy-Weinberg Equilibrium (HWE)

- Need to determine if data set is unbiased
  - Check that allele frequencies are **independent**

    $p_{AA} = p_A^2$ $\quad$ $p_{Aa} = 2p_A p_a$ $\quad$ $p_{aa} = p_a^2$

  - Observed counts: N_{AA} , N_{Aa} , N_{aa}

    $$p_A = \frac{2N_{AA} + N_{aa}}{N} \qquad p_a = 1 - p_A$$

  - Expected counts: E_{AA} , E_{Aa} , E_{aa}

    $E_{AA} = Np_A^2$ $\quad$ $E_{Aa} = 2Np_A p_a$ $\quad$ $N_{aa} = Np_a^2$

Pearson Test

$$X^2 = \frac{(E_{AA} - N_{AA})^2}{E_{AA}} + \frac{(E_{Aa} - N_{Aa})^2}{E_{Aa}} + \frac{(E_{aa} - N_{aa})^2}{E_{aa}}$$

deg 4 in N_{AA} , N_{Aa} , N_{aa}

Data

Encrypt!!

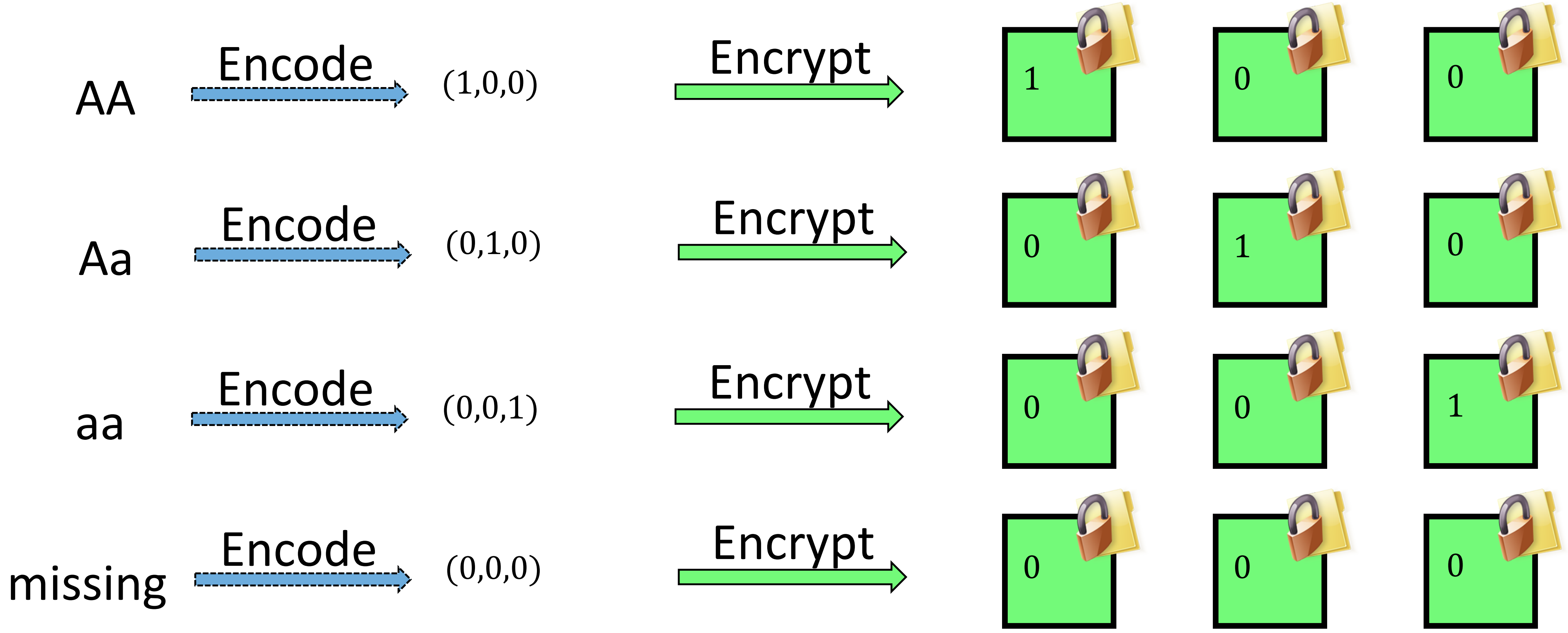
Input Data: genotypes

	Person 1	Person 2	Person 3	Person 4	…
Chr 1, location 345	AA	Aa	AA	aa	…
Chr 19, location 147	Bb	bb	bb	BB	…
…	…	…	…	…	

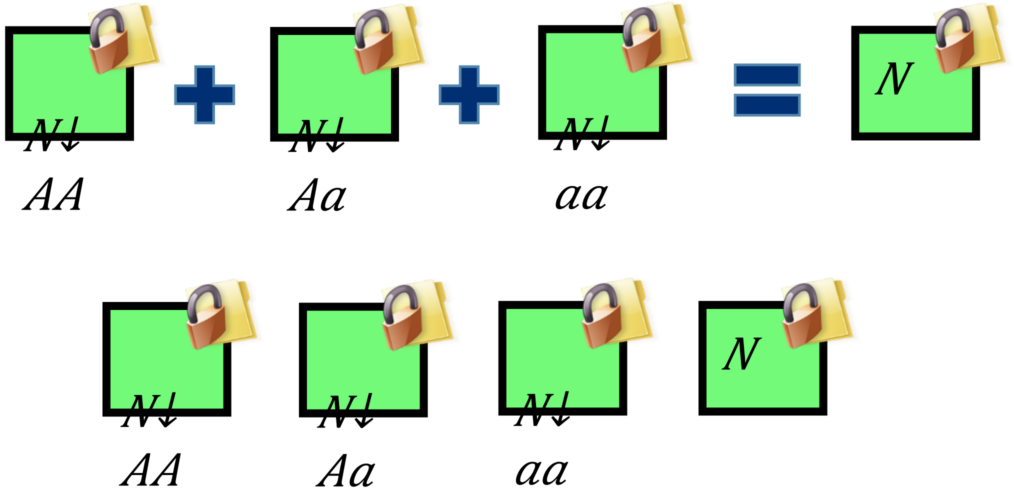
2 questions:

- How to **encode** genotypes (AA,Aa,aa)
- How to **obtain observed counts** from encrypted genotypes?

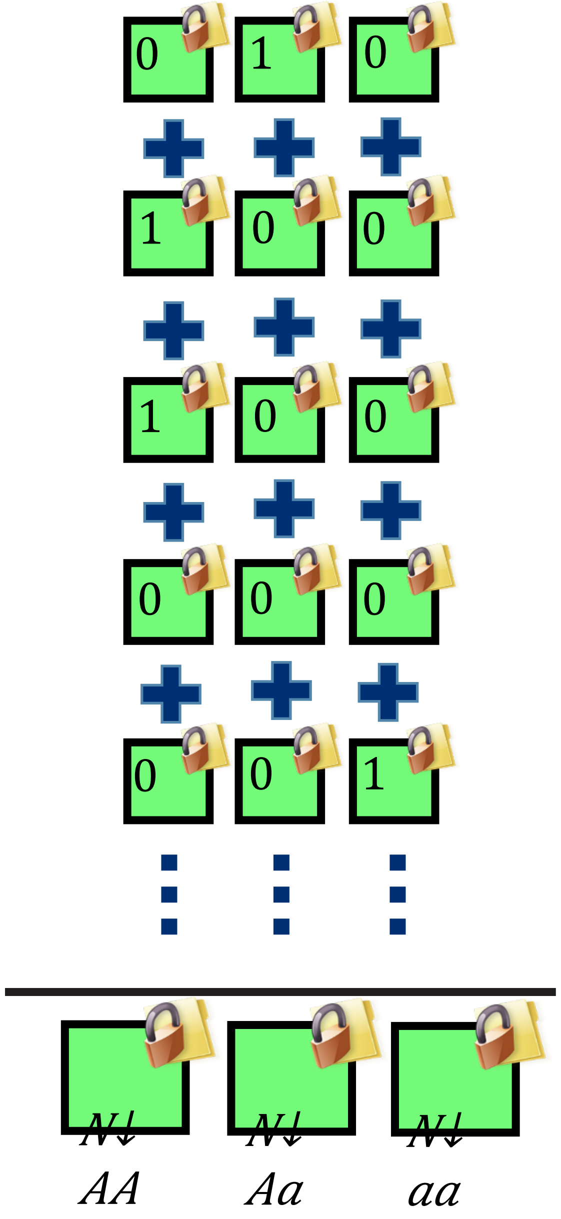
Encoding and encrypting of genotype data



Computing genotype counts



- Only homomorphic additions
- Cost linear in size of data sample



Pearson goodness-of-fit test

Tests for Hardy-Weinberg Equilibrium, i.e. whether allele frequencies are statistically independent

$$p_{AA} = p_A^2 \, , \; p_{Aa} = 2 p_A p_a \, , \; p_{aa} = p_a^2$$

- $p_{AA} = N_{AA}/N$, $p_{Aa} = N_{Aa}/N$, $p_{aa} = N_{aa}/N$
- Observed counts: N_{AA}, N_{Aa}, N_{aa},

$$p_A = 2N_{AA} + N_{Aa}/2N \, , \; p_a = 1 - p_A$$

- Expected counts: $E_{AA} = N p_A^2$, $E_{Aa} = 2N p_A p_a$, $E_{aa} = N p_a^2$

Pearson goodness-of-fit test

- Compute the X^2 test statistic

$$X^2 = (N_{AA} - E_{AA})^2 / E_{AA} + (N_{Aa} - E_{Aa})^2 / E_{Aa} + (N_{aa} - E_{aa})^2 / E_{aa}$$

- Problem: Arithmetic circuits over R do not allow divisions
- Rewrite the formula to avoid divisions

Modified algorithm

It turns out that

$$X^2 = \frac{\alpha}{2N}\left(\frac{1}{\beta_1} + \frac{1}{\beta_2} + \frac{1}{\beta_3}\right),$$

where

$$\beta_2 = (2N_{AA} + N_{Aa})(2N_{aa} + N_{Aa}), \qquad \beta_3 = 2(2N_{aa} + N_{Aa})^2$$

- Return encryptions of values $\alpha, \beta_1, \beta_2, \beta_3, N$
- X^2 is computed after decryption

Genetic algorithm performance

80-bit security

- Parameter set I: $n=4096,\ q\approx 2^{192}$, ciphertext $\approx 100\text{KB}$
- Parameter set II: $n=8192,\ q\approx 2^{384}$, ciphertext $\approx 400\text{KB}$

Algorithm	Pearson	EM (iterations)			LD	CATT
		1	2	3		
Parameters I	0.3s	0.6s	1.1s	-	0.2s	1.0s
Parameters II	1.4s	2.3s	4.5s	6.9s	0.7s	3.6s

Proof-of-concept implementation: computer algebra system Magma,
Intel Core i7 @ 3.1GHz, 64-bit Windows 8.1

Performance

- Data quality (Pearson Goodness-of-Fit)

~ 0.3 seconds, 1,000 patients

- Predicting Heart Attack (Logistic Regression)

~ 0.2 seconds

- Building models (Linear Means Classifier)

~0.9 secs train, classify: 30 features, 100 training samples

- Sequence matching (Edit distance)

~27 seconds amortized, length 8

Core i7 3.4GHz

80-bit security

Joint work with:

- Can Homomorphic Encryption be Practical?

Kristin Lauter, Michael Naehrig, Vinod Vaikuntanathan, CCSW 2011

- ML Confidential: Machine Learning on Encrypted Data

* Thore Graepel, Kristin Lauter, Michael Naehrig, ICISC 2012

- Predictive Analysis on Encrypted Medical Data

Joppe W. Bos, Kristin Lauter, and Michael Naehrig, Journal of Biomedical Informatics, 2014.

- Private Computation on Encrypted Genomic Data

Kristin Lauter, * Adriana Lopez-Alt, * Michael Naehrig, GenoPri2014, LatinCrypt2014.

- Homomorphic Computation of Edit Distance

Jung Hee Cheon, Miran Kim, Kristin Lauter, in submission.

Challenges for the future:

- Public Databases: multiple patients under different keys
- More efficient encryption at scale
- Integrate with other crypto solutions
- Expand functionality
- Attack underlying hard problems