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 한솔박을 먹고 있는 김미란 연구원(28)은 생체정보 보안 연구 분야에서 떠오르는 셋별이다. 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

1. Enter medical data
2. Encrypt values with personal password
3. Send encrypted values to Azure
4. Cloud runs prediction algorithm on encrypted data
5. Cloud returns encrypted prediction
6. Decrypt prediction with personal password
Scenario for genomic data

Trusted party
hosts data and
regulates access

Untrusted cloud service
Stores, computes on encrypted data

Requests for decryption of results
(requires a policy)

Researcher:
requests encrypted results of specific computations
Multi-party computation for genomic data

Untrusted cloud service
Stores, computes on encrypted data

Researcher

Researcher
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

\[ a, b \rightarrow \text{compute} \rightarrow a + b \rightarrow \text{encrypt} \]

\[ E(a), E(b) \rightarrow \text{compute} \]

\[ E(a) + E(b) = E(a + b) \]
Homomorphic Encryption: multiplication

\[
\begin{align*}
& a, b \\
\rightarrow & \text{compute} \\
& a \times b \\
\downarrow & \text{encrypt} \\
& E(a), E(b) \\
\rightarrow & \text{compute} \\
& E(a), E(b) \\
\downarrow & \text{encrypt} \\
& E(a) \times E(b) = E(a \times b)
\end{align*}
\]
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:

\[
(\text{Enc}(m\downarrow_1), \text{Enc}(m\downarrow_2)) \rightarrow \text{Enc}(m\downarrow_1 + m\downarrow_2)

(\text{Enc}(m\downarrow_1), \text{Enc}(m\downarrow_2)) \rightarrow \text{Enc}(m\downarrow_1 \cdot m\downarrow_2)
\]
Fully Homomorphic Encryption (FHE)

FHE enables **unlimited** computation on **encrypted data**

- **Public operations on ciphertexts:**

  \[ \begin{align*}
  (\text{Enc}(m_{\downarrow 1}), \text{Enc}(m_{\downarrow 2})) & \rightarrow \text{Enc}(m_{\downarrow 1} + m_{\downarrow 2}) \\
  (\text{Enc}(m_{\downarrow 1}), \text{Enc}(m_{\downarrow 2})) & \rightarrow \text{Enc}(m_{\downarrow 1} \cdot m_{\downarrow 2})
  \end{align*} \]

- For data encrypted bitwise \((m_{\downarrow 1}, m_{\downarrow 2} \in \{0,1\})\), operations \(m_{\downarrow 1} + m_{\downarrow 2}\) and \(m_{\downarrow 1} \cdot m_{\downarrow 2}\) 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

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:

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 $\rightarrow$ 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 \( q \equiv 1 \mod 2^n \) 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 $s \leftarrow D\downarrow \sigma (R\downarrow q)$. Secret key: sk = s.

• Enc(sk, m): m: encoding of message $m \in \{0,1\}^n$ as a “small” element of $R\downarrow q$, $a$ is uniformly random in $R\downarrow q$, $e$ is a “small” ring element $e \leftarrow D\downarrow \sigma (R\downarrow q)$. 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)

<table>
<thead>
<tr>
<th>#mult</th>
<th>n</th>
<th>size(q)</th>
<th>PK size</th>
<th>SK size</th>
<th>CT size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2048</td>
<td>58 bits</td>
<td>30 KB</td>
<td>2 KB</td>
<td>≥ 30 KB</td>
</tr>
<tr>
<td>10</td>
<td>8192</td>
<td>354 bits</td>
<td>720 KB</td>
<td>8 KB</td>
<td>≥ 720 KB</td>
</tr>
<tr>
<td>32</td>
<td>65536</td>
<td>1298 bits</td>
<td>20 MB</td>
<td>66 KB</td>
<td>≥ 20 MB</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Operation</th>
<th>x86-64</th>
</tr>
</thead>
<tbody>
<tr>
<td>SH_Keygen</td>
<td>250 ms</td>
</tr>
<tr>
<td>SH_Enc</td>
<td>24 ms</td>
</tr>
<tr>
<td>SH_Add</td>
<td>1 ms</td>
</tr>
<tr>
<td>SH_Mul</td>
<td>41 ms</td>
</tr>
<tr>
<td>SH_Dec (2-element ciphertext)</td>
<td>15 ms</td>
</tr>
<tr>
<td>SH_Dec (3-element ciphertext)</td>
<td>26 ms</td>
</tr>
</tbody>
</table>
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
  \[ R = \mathbb{Z}[X]/(X^n + 1), \quad n = 2^k \]

- Work with polynomials in \( R \) modulo some \( q \in \mathbb{Z} \)

- Homomorphic operations (\( + / \times \)) correspond to polynomial operations (add/mult) in \( R \)
  - \( + \) is relatively efficient, \( \times \) 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 \( m = \sum_{i=0}^{n-1} a_i x^i \) \( \in \mathbb{R} \)
  - After decryption, evaluate \( m \) at \( x=2 \)

○ GLN, BLN - Real number \( b \) up to precision \( s \)
  - Encode \( 10^s b \) as above
  - E.g. encode \( \pi \) with precision \( s=2 \) as
    \[
    \text{Encode}(314) = x^8 + x^5 + x^4 + x^3 + 2
    \]
  - 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: \( n=4096, q\approx2^{192} \), ciphertext \( \approx100\text{KB} \)
- Parameter set II: \( n=8192, q\approx2^{384} \), ciphertext \( \approx400\text{KB} \)

<table>
<thead>
<tr>
<th>Operation</th>
<th>KeyGen</th>
<th>Encrypt</th>
<th>Add</th>
<th>Mult</th>
<th>Decrypt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters I</td>
<td>3.6s</td>
<td>0.3s</td>
<td>0.001s</td>
<td>0.05s</td>
<td>0.04s</td>
</tr>
<tr>
<td>Parameters II</td>
<td>18.1s</td>
<td>0.8s</td>
<td>0.003s</td>
<td>0.24s</td>
<td>0.26s</td>
</tr>
</tbody>
</table>

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
<table>
<thead>
<tr>
<th>Predictions on Medical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td>0.05</td>
</tr>
<tr>
<td>0.13</td>
</tr>
<tr>
<td>0.10</td>
</tr>
<tr>
<td>0.11</td>
</tr>
<tr>
<td>0.04</td>
</tr>
<tr>
<td>0.21</td>
</tr>
<tr>
<td>0.02</td>
</tr>
<tr>
<td>0.12</td>
</tr>
<tr>
<td>0.10</td>
</tr>
<tr>
<td>0.10</td>
</tr>
<tr>
<td>0.28</td>
</tr>
<tr>
<td>0.04</td>
</tr>
<tr>
<td>0.00</td>
</tr>
</tbody>
</table>
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

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

<table>
<thead>
<tr>
<th>SH.Keygen</th>
<th>SH.Enc</th>
<th>SH.Dec(^2)</th>
<th>SH.Dec(^3)</th>
<th>SH.Add</th>
<th>SH.Mult</th>
</tr>
</thead>
<tbody>
<tr>
<td>156</td>
<td>379</td>
<td>29</td>
<td>52</td>
<td>1</td>
<td>106</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>data</th>
<th># features</th>
<th>algorithm</th>
<th>train</th>
<th>classify</th>
</tr>
</thead>
<tbody>
<tr>
<td>surrogate</td>
<td>2</td>
<td>linear means</td>
<td>230</td>
<td>235</td>
</tr>
<tr>
<td>Iris</td>
<td>4</td>
<td>linear means</td>
<td>510</td>
<td>496</td>
</tr>
</tbody>
</table>
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
    \[ p_{AA} = p_A^2 \quad p_{Aa} = 2p_Ap_a \quad p_{aa} = p_a^2 \]
  - Observed counts: \( N_{AA}, N_{Aa}, N_{aa} \)
    \[ p_A = \frac{2N_{AA} + N_{aa}}{N} \quad p_a = 1 - p_A \]
  - Expected counts: \( E_{AA}, E_{Aa}, E_{aa} \)
    \[ E_{AA} = Np_A^2 \quad E_{Aa} = 2Np_Ap_a \quad N_{aa} = Np_a^2 \]

Pearson Test
\[
\chi^2 = \frac{(E_{AA} - N_{AA})^2}{E_{AA}} + \frac{(E_{Aa} - N_{Aa})^2}{E_{Aa}} + \frac{(E_{aa} - N_{aa})^2}{E_{aa}}
\]
\[ \text{deg 4 in } N_{AA}, N_{Aa}, N_{aa} \]
### Data

**Input Data:** genotypes

<table>
<thead>
<tr>
<th>Chr 1, location 345</th>
<th>Person 1</th>
<th>Person 2</th>
<th>Person 3</th>
<th>Person 4</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>Aa</td>
<td>AA</td>
<td>aa</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Chr 19, location 147</td>
<td>Bb</td>
<td>bb</td>
<td>bb</td>
<td>BB</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**2 questions:**

- How to **encode** genotypes (AA,Aa,aa)
- How to **obtain observed counts** from encrypted genotypes?
Encoding and encrypting of genotype data

- **AA**: Encode $\rightarrow (1,0,0)$ Encrypt $\rightarrow 1$
- **Aa**: Encode $\rightarrow (0,1,0)$ Encrypt $\rightarrow 0$
- **aa**: Encode $\rightarrow (0,0,1)$ Encrypt $\rightarrow 0$
- **missing**: Encode $\rightarrow (0,0,0)$ Encrypt $\rightarrow 0$
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

\[ p\downarrow AA = p\downarrow A \downarrow A \cdot 2, \ p\downarrow Aa = 2p\downarrow A \ p\downarrow a, \ p\downarrow aa = p\downarrow a \downarrow a \]

- \( p\downarrow AA = N\downarrow AA / N \), \( p\downarrow Aa = N\downarrow Aa / N \), \( p\downarrow aa = N\downarrow aa / N \)

- **Observed counts**: \( N\downarrow AA, N\downarrow Aa, N\downarrow aa \),

\[ p\downarrow A = 2N\downarrow AA + N\downarrow Aa / 2N, \ p\downarrow a = 1 - p\downarrow A \]

- **Expected counts**: \( E\downarrow AA = Np\downarrow A \downarrow A \cdot 2, \ E\downarrow Aa = 2Np\downarrow A \ p\downarrow a \), \( E\downarrow aa = Np\downarrow a \downarrow a \)
Pearson goodness-of-fit test

• Compute the $\chi^2$ test statistic

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

• Problem: Arithmetic circuits over $R$ do not allow divisions
• Rewrite the formula to avoid divisions
Modified algorithm

It turns out that

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

where

\[ \beta_2 = (2N_A + N_a) (2N_a + N_A a) \quad \beta_3 = 2 (2N_a + N_A a)^2 \]

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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Pearson</th>
<th>EM (iterations)</th>
<th>LD</th>
<th>CATT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Parameters I</td>
<td>0.3s</td>
<td>0.6s</td>
<td>1.1s</td>
<td>-</td>
</tr>
<tr>
<td>Parameters II</td>
<td>1.4s</td>
<td>2.3s</td>
<td>4.5s</td>
<td>6.9s</td>
</tr>
</tbody>
</table>

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

- Private Computation on Encrypted Genomic Data

- 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