7-7 Long Term Cryptography and Applications to Privacy-Preserving Data Mining

Le Trieu PHONG, Yoshinori AONO, Takuya HAYASHI, and Lihua WANG

We develop a technique called Security-updatable Public-key Homomorphic Encryption with Rich Encodings (SPHERE), enabling both homomorphic operations and security-updatability over encrypted data. By SPHERE, it is possible to protect private data for the long term, while enabling the utility of encrypted data. We demonstrate that SPHERE can be useful in privacy-preserving data mining, especially when the data come from medical sources containing genomic information.

1 Introduction

With the development of data mining techniques, various new businesses based on big data analysis have appeared. This includes data containing private information, and the data must be analyzed while preserving privacy. One way to do this is by using "homomorphic encryption." In homomorphic encryption, various calculations can be performed on the encrypted data itself, so it is possible to do statistical analysis on the encrypted data, and only a person holding the decryption key can obtain those statistical results, making it possible to build a statistical analysis system that considers privacy.

In data analysis handling medical data or genomic information typified by genome-wide association studies, a leak of information could affect not just that person but also his/her relatives and children, so it is necessary to maintain security for a longer time than for usual data. On the other hand, with the evolution of computers and developments in cryptanalysis technology, cryptanalysis abilities are improving gradually, so the security of encrypted data declines over time. Security of cryptosystems is estimated from predicted advancements in computers, but it is difficult to predict their futures, so security of cryptosystems can be guaranteed for only a few decades at most. Therefore, it is necessary to ensure secure use over a long period by, for example, decrypting encrypted data first and then encrypting it by more advanced cryptosystems. In such a case, the data is exposed temporarily to a high risk of leak.

Against this backdrop, we have developed a new homomorphic encryption technology called SPHERE (Security-updatable Public-key Homomorphic Encryption with Rich Encodings) which not only uses homomorphic encryption for statistical analysis etc., but also enables updating the security level (key length) without increasing leak risks. Therefore, it enables secure use of encrypted data over a long period of time.

Further, as an application, we conducted a variant of logistic regression analysis over data encrypted by SPHERE. Logistic regression analysis includes operations such as exponentiations and logarithms that are unsuitable for computation over encrypted data, so high speed calculation is difficult to achieve with it as-is. This problem was resolved by improving polynomial approximations of exponential functions and logarithmic functions, as well as data preprocessing done on the data provider's side. The experiment was done with a hundred million records of simulated data, and it was confirmed that the analysis can be done in about 30 minutes.

2 New homomorphic encryption scheme (SPHERE)

The SPHERE scheme we developed consists of ℓ dimension vector of $\mathbb{Z}_p$ (where $\mathbb{Z}_p = \{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\}$) in plaintext, and $n+\ell$ dimension vector of $\mathbb{Z}_q$ in ciphertext. The homomorphic operation involves addition multiple times, and multiplication once (tensor product of vector), and also supports a security level update function. Figure 1 shows its cryptographic algorithms. In the figure, ParamGen( ) is parameter generation, KeyGen( ) is key generation, Enc( ) is encryption, and Dec( ) is decryption. *pk* is the public key, and *sk* the secret key. The security of ciphertext and keys is determined by the Learning with Error problem. For security proofs and other details, see [1].

PKE part

ParamGen(1^λ):
Fix $q = q(\lambda) \in \mathbb{Z}^+$
Fix $l \in \mathbb{Z}^+$
Fix $p \in \mathbb{Z}^+$, $\gcd(p, q) = 1$
Return $pp = (q, l, p)$

KeyGen($1^\lambda, pp$):
Take $s = s(\lambda, pp) \in \mathbb{R}^+$
Take $n = n(\lambda, pp) \in \mathbb{Z}^+$
$R, S \xleftarrow{g} \mathbb{Z}_s^{n \times l}$, $A \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$
$P = pR - AS \in \mathbb{Z}_q^{n \times l}$
Return $pk = (A, P, n, s)$, $sk = S$

Enc($pk, m \in \mathbb{Z}_p^{1 \times l}$):
$e_1, e_2 \xleftarrow{g} \mathbb{Z}_s^{1 \times n}$, $e_3 \xleftarrow{g} \mathbb{Z}_s^{1 \times l}$
$c_1 = e_1 A + pe_2 \in \mathbb{Z}_q^{1 \times n}$
$c_2 = e_1 P + pe_3 + m \in \mathbb{Z}_q^{1 \times l}$
Return $c = (c_1, c_2)$

Dec($S, c = (c_1, c_2)$):
$\overline{m} = c_1 S + c_2 \in \mathbb{Z}_q^{1 \times l}$
$m = \overline{m} \bmod p$
Return m

Homomorphic part

Add(c, c'):
Return $c + c' \bmod q$

AddM($c_{\mathrm{mul}}, c'_{\mathrm{mul}}$)
Return $c_{\mathrm{mul}} + c'_{\mathrm{mul}} \bmod q$

DecM(sk, c_{mul}):
Let $sk = S \in \mathbb{Z}_q^{n \times l}$
$\overline{m} = \begin{bmatrix} S \\ I_l \end{bmatrix}^{\mathsf{T}} (c_{\mathrm{mul}}) \begin{bmatrix} S \\ I_l \end{bmatrix} \in \mathbb{Z}_q^{l \times l}$ (I_l: the identity matrix)
Return $\overline{m} \in \mathbb{Z}_p^{l \times l}$

Mul(pp, pk, c, c'):
Return $c_{\mathrm{mul}} = c^{\mathsf{T}} c' \in \mathbb{Z}_q^{(n+l) \times (n+l)}$

DecA(sk, c_{add}):
identical to Dec

Fig. 1 Cryptographic algorithms of SPHERE

2.1 Learning with Error problem

When Matrix A sampled uniformly at random from integers modulo q and integer vector $b = (Ax + e \bmod q)$ are given, the problem for solving integer vector x is said to be a Learning with Error (LWE) problem. Here, e is the noise vector extracted from discrete Gaussian distribution of variance s^2. For the shortest vector problem and the assessment of its complexity, see [2]. Complexity of an LWE problem is given by the exponential time of the dimension of vector x, so the greater the dimension of x, the more complex the calculation. Further, the LWE problem is different from integer factorization and discrete logarithm problems in the sense that it can possibly resist against even large-scale quantum computers. Therefore, the hardness of the LWE problem is a guarantee for post-quantum cryptography.

In an example of decrypting plaintext from ciphertext, we see a relation between SPHERE and LWE problems. The ciphertext of SPHERE, $c = (c_1, c_2)$, is given as $c_1 = e_1 A + pe_2 \bmod q, c_2 = e_1 P + pe_3 + m \bmod q$. A and P are the public keys, e_1, e_2, e_3 are noise vectors unknown to the attacker, and m is plaintext. Plaintext $m = c_2 - e_1 P + pe_3 \bmod q$ is the vector of $\mathbb{Z}_p$, so if e_1 is found, the plaintext can be obtained from $m = (c_2 - e_1 P) \bmod p$. However, to get e_1, $c_1 = (e_1 A + pe_2) \bmod q$ has to be solved first. This is equivalent to an LWE problem, so if the problem is sufficiently difficult, decrypting the plaintext from the ciphertext is also likely to be difficult.

2.2 Security level update

The security of SPHERE is determined by the LWE problem, and the difficulty of the LWE problem lies in the exponential time of the dimension of the secret vector (denoted by parameter n in SPHERE). Therefore, if n is changed to new $n'(> n)$ without disturbing the information or structure of ciphertext, the security level can be updated. To achieve this, we applied a technique called dimension switching [3]. Figure 2 shows the security level update process. In the figure, the keys before and after update are $(pk_i, sk_i) = \{(A_i, P_i, n_i), S_i\}$, and $i = 1,2$ indicates before and after update, respectively. Bits( ) is the bit conversion function to reduce each component to a size as small as noise, while keeping the information of the ciphertext intact. Power2( ) is the function amounting to Bits (c_1) Power2 $(S_1) = c_1 S_1$. UKGen( ) is update key generation, and Update( ) is the ciphertext security level update process.

Key rotation and security update

UKGen($pp, pk_1, sk_1, pk_2, sk_2$):
Let $pk_i = (A_i, P_i, n_i, s_i)$. Let $sk_i = S_i$ $(i = 1, 2)$
Let $\kappa = \lceil \log_2 q \rceil$. Take $X \xleftarrow{\$} \mathbb{Z}_q^{n_1 \kappa \times n_2}$, $E \xleftarrow{g} \mathbb{Z}_{s_2}^{n_1 \kappa \times l}$
$Y = -XS_2 + pE + \mathsf{Power2}(S_1) \in \mathbb{Z}_q^{n_1 \kappa \times l}$
Return $uk_{n_1 \to n_2} = (X, Y)$

Update($pp, c, uk_{n_1 \to n_2}, pk_2$):
Let $c = (c_1, c_2) \in \mathbb{Z}_q^{1 \times n_1} \times \mathbb{Z}_q^{1 \times l}$
Let $pk_2 = (A_2, P_2, n_2, s_2)$
Let $uk_{n_1 \to n_2} = (X, Y)$, and $f_1, f_2 \xleftarrow{g} \mathbb{Z}_{s_2}^{1 \times n_2}$, $f_3 \xleftarrow{g} \mathbb{Z}_{s_2}^{1 \times l}$
$E_0 = f_1[A_2 | P_2] + p[f_2 | f_3] \in \mathbb{Z}_q^{1 \times (n_2 + l)}$
$F = [\mathsf{Bits}(c_1)X | \mathsf{Bits}(c_1)Y + c_2] \in \mathbb{Z}_q^{1 \times (n_2 + l)}$
Return $c' = E_0 + F \in \mathbb{Z}_q^{1 \times (n_2 + l)}$

Fig. 2 Security level update process details

2.3 Fixed point number encoding

As described at the start of this section, each element in the plaintext space of SPHERE is an ℓ dimension vector of $\mathbb{Z}_p$. However, real values (with precision) must be handled in the actual processing system which needs to be constructed with homomorphic encryption, such as a statisti-

cal analysis system. A technique of encoding $\mathbb{Z}_p$ coefficient polynomial, and then to integers and fixed point numbers is described here.

For encoding of $\mathbb{Z}_p$ coefficient polynomials, one can consider each coefficient of polynomial $A(x) = \sum_{i=0}^{\ell-1} a_i x^i$ to be vector $A = (a_0 \dots a_{\ell-1})$ which is contained in each element. In this expression, addition/subtraction of vectors $(A \pm B)$ will obviously correspond to addition/subtraction of the corresponding polynomial $(A(x) \pm B(x))$. In polynomial multiplication, each element of tensor product $A^{\mathsf{T}}B$ of vector is the product $a_i b_j$ of the elements of vector A, B, so if the elements of $A^{\mathsf{T}}B$ are appropriately summed, it is possible to calculate each coefficient of $A(x) \times B(x)$.

The signed binary expression when integers and fixed point numbers are encoded will be as follows.

$$N = \sum_{i=-L}^{\ell-L-1} n_i 2^i \ \left(n_i \in \begin{cases} \{0,1\} \text{ if } N \geq 0 \\ \{-1,0\} \text{ if } N < 0 \end{cases}\right)$$

It is defined by corresponding each n_i to the coefficients of the polynomial. When $L = 0$, it is an integer, but when $L > 0$, it is a fixed point number of L digits precision below the decimal point. The operations of $\mathbb{Z}_p$ coefficient polynomial and SPHERE plaintext are corresponding, so operation of plaintext of SPHERE enables the operation of integers and fixed point numbers. If each coefficient in the result of the operation with vectors is within the range of $\mathbb{Z}_p = \{-\frac{p-1}{2}, \dots, \frac{p-1}{2}\}$, then the corresponding polynomials $N(x)$ *and* N correspond on a one-to-one basis, and if the operation result is not within the $\mathbb{Z}_p$ range, it is not possible to revert to integers and fixed point numbers correctly, so it is necessary to control the number of operations and size of p appropriately.

2.4 Implementation results

The SPHERE scheme is implemented and the processing time is measured in Table 1 and Table 2. In the table, bit-sec means bit security, Key rotation indicates key update of the same security level, and Security update indicates the security level update process. The experiment was carried out using a Xeon E5-2660v3 (2.60 GHz), and all calculations were done using a single thread.

3 Application to privacy-preserving data mining – logistic regression analysis

Logistic regression analysis is one type of supervised machine learning, and it has a wide range of applications. An example can be a service that diagnoses whether a patient has an illness or not. In such a case, the data for training and predicting is information that involves privacy, so of course it must be handled carefully. We applied homomorphic encryption technology to carry out logistic regression analysis while handling data securely. In the system we developed, the training data and the data to be classified are encrypted, so they are completely protected.

Figure 3 shows the data processing model of the system we developed. The data given by the data providers (patients themselves, medical institutions, etc.) is encrypted and saved in the central server. The server carries out the process related to logistic regression analysis using the homomorphisms of encryption, and sends the processed result still in encrypted form to the data analyst. By decrypting that, the data analyst can extract the learned logistic regression coefficient.

3.1 Logistic regression analysis

The data set to be studied is expressed as record $\{x^{(i)}, y^{(i)}\}_{1 \leq i \leq N_{data}}$ of N_{data} items, and each record shall be a pair of $d+1$ dimensional real number vector $x^{(i)} = \left(1, x_1^{(i)}, \dots, x_d^{(i)}\right)$ and label $y^{(i)} \in \{0,1\}$. The cost function J where the regression coefficient $\theta = (\theta_0, \dots, \theta_d)$ is considered as the variable, is defined as

Table 1 Measured timings of cryptographic algorithms (Milliseconds)

bit-sec	KeyGen	Enc	Dec	Add	Mul	AddM	DecM
80	1428	63.2	0.92	0.003	35.1	29.3	1313
128	2513	94.7	1.22	0.004	60.8	50.1	2296
256	7249	313	2.05	0.010	164	136	6643

Table 2 Measured timings of security update process (Seconds)

	bit-sec → bit-sec	UKGen	Update
Key rotation	80 → 80	165.6	1.1
	128 → 128	291.4	1.9
	256 → 256	846.6	5.3
Security update	80 → 128	238.2	1.5
	128 → 256	519.8	3.3

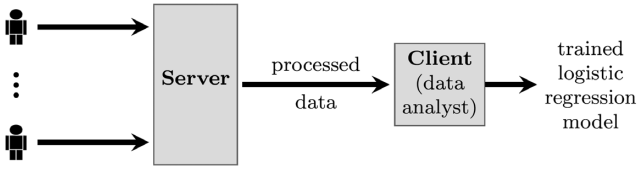


Fig. 3 Data process model

$$J(\theta) = \frac{\lambda}{2N_{data}} \sum_{j=1}^{d} \theta_j^2 + J^*(\theta).$$

However, it shall be

$$J^*(\theta) = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \left[-y^{(i)} \log\left(h_\theta(x^{(i)})\right) - (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right) \right].$$

Here, $h_\theta(x) = h_\theta(x_0, \ldots, x_d)$ is the sigmoid function, defined as $h_\theta(x) = \frac{1}{1 + \exp(-\theta^\top x)} = \frac{1}{1 + \exp(-\sum_{j=0}^{d} \theta_j x_j)}$. Always, $x_0 = 1$.

In the training phase of logistic regression analysis, $\theta^* = \mathrm{argmin}_\theta J(\theta)$ is calculated for minimizing cost for the assigned data set.

In the predicting phase, corresponding to newly entered $x^{new} = (1, x_1^{new}, \ldots, x_d^{new})$ data to be judged, the label y^{new} belongs to either of $\{0,1\}$, which is estimated by the following formula.

$$y^{new} = \begin{cases} 1 \text{ if } h_{\theta^*}(x^{new}) \geq \text{thres} \\ 0 \text{ if } h_{\theta^*}(x^{new}) < \text{thres} \end{cases}$$

Here, thres $\in (0,1)$ is the threshold value, and in most cases, thres=1/2 is used.

3.2 Improvement 1 for homomorphic encryption: Deletion of exponential/logarithmic function by polynomial approximation

The definition of function $J^*(\theta)$ includes an exponential/logarithmic function, so if one tries to implement these functions precisely by homomorphic encryption, then it leads to an extremely complicated form. To solve this problem, a learning algorithm is built by approximating these functions using appropriate polynomials, specifically quadratic polynomials.

Using quadratic approximation $\sum_{j=0}^{2} a_j (u^j)$ of function $\log(\frac{1}{1 + \exp(u)})$, $\log\left(h_\theta(x)\right)$ and $\log\left(1 - h_\theta(x)\right)$ are approximated to

$$\log\left(1 - h_\theta(x)\right) \approx \sum_{j=0}^{2} a_j (\theta^\top x)^j, \log\left(h_\theta(x)\right) \approx \sum_{j=0}^{2} (-1)^j a_j (\theta^\top x)^j.$$

At this time, the approximated $J^*_{approx}(\theta)$ of $J^*(\theta)$ where the above formula was used is

$$J^*_{approx}(\theta) = \frac{1}{N_{data}} \sum_{j=1}^{2} \sum_{r_1, r_2}^{d} a_j \left(\theta_{r_1}, \ldots, \theta_{r_j}\right) A_{j,r_1,r_j} - a_0.$$

However, it will be

$$A_{1,r_1,r_1} = \sum_{i=1}^{N_{data}} \left(2y^{(i)} - 1\right) \left(x_{r_1}^{(i)}\right) \quad (1)$$

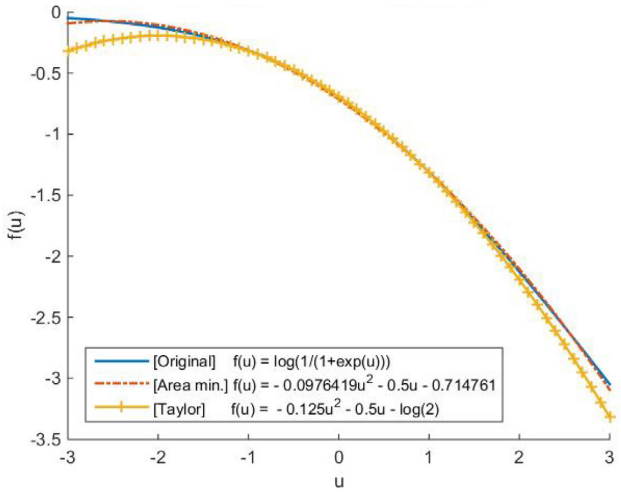


Fig. 4 Original function and the approximated quadratic polynomials

$$A_{2,r_1,r_2} = \sum_{i=1}^{N_{data}} -(x_{r_1}^{(i)} x_{r_2}^{(i)}) \quad (2).$$

(Taylor) $a_0 = -\log 2, a_1 = -0.5, a_2 = -0.125$ by Taylor expansion is well known as an approximation coefficient. The error is comparatively large in two-dimensional development, so we used (LSM) $a_0 = -0.714761, a_1 = -0.5, a_2 = -0.0976419$ in which the error was minimized by the least squares method in interval $x \in [-3,3]$. Figure 4 shows the original function, and the graphs approximated by the respective quadratic polynomials.

3.3 Improvement 2 for homomorphic encryption: Removing multiplications by the data provider performing calculations beforehand

Logistic regression analysis is possible if the homomorphic encryption method supports quadratic form computation, by polynomial approximation of the exponential function and logarithmic function. However, as seen in Table 1, SPHERE's multiplication Mul( ) and its decryption DecM( ) are relatively slow. Therefore, to analyze on a large-scale data set, such as 100 million records, it is necessary to make the system more efficient.

From the formulas (1) and (2) shown in Subsection **3.2**, we see that

each item $a_{1,r_1}^{(i)} = \left(2y^{(i)} - 1\right)\left(x_{r_1}^{(i)}\right), a_{2,r_1,r_2}^{(i)} - \left(x_{r_1}^{(i)} x_{r_2}^{(i)}\right)$ is comprised of records $\{x^{(i)}, y^{(i)}\}$. This means that for the computation of $a_{1,r_1}^{(i)}, a_{2,r_1,r_2}^{(i)}$, the data provider can compute before encryption. Using this, logistic regression analysis using homomorphic encryption is carried out as follows.

The data provider that has records $\{x^{(i)}, y^{(i)}\}$ calculates $a_{1,r_1}^{(i)}, a_{2,r_1,r_2}^{(i)} (0 \leq r_1, r_2 \leq d)$, and encrypts each, and sends those results to the server. The server can obtain $\mathrm{Enc}(A_{1,r_1,r_1}), \mathrm{Enc}(A_{2,r_1,r_2})$ by homomorphic addition of

the received ciphertexts $\mathrm{Enc}(a_{1,r_1}^{(i)}), \mathrm{Enc}(a_{2,r_1,r_2}^{(i)})$. Later, the client will decrypt, and $J_{approx}^{*}(\theta)$ is obtained from $A_{1,r_1,r_1}, A_{2,r_1,r_2}$, and using that, the learned logistic regression coefficient θ^{*} is calculated.

In this method, while the data provider must encrypt each $a_{1,r_1}^{(i)}, a_{2,r_1,r_2}^{(i)}$, it is sufficient for the server to conduct only homomorphic addition, so more efficient calculations are possible in a model where there are many more data providers than servers, as in the model we have assumed. Also, since this improvement requires only homomorphic additions, additive homomorphic cryptosystems, e.g., Paillier encryption [4], which only supports additions on encrypted data, can be employed.

3.4 Experiment results

Regarding the improvements described above, (1) the computation time of the server for a large-scale simulated data set was measured, and (2) the statistical index of an actual data set was evaluated. The latter was done to evaluate the effect on practicality by quadratic polynomial approximation.

Figure 5 shows the computation time of the server for a large-scale simulated data set. The number of records are 100 million and 200 million, and the horizontal axis is the data's dimension d. The computation experiment was conducted using parameters that achieve 128-bit security, and using a server equipped with two Xeon E5 -2660v3 (2.60 GHz), the computations were done in 20 threads. Even if $d \leq 40$ (approximately), we found that the analysis is completed within 30 minutes, even for a 100 million data set.

Next, Table 3 shows the evaluation of statistical index of the actual data set. From among the public data repositories of University of California, Irvine [5], the data set related to diabetes among the Pima Indians of North America (Pima), and the data set of relations between results of single-photon emission computed tomography of a heart and the presence or absence of heart diseases (SPECTF) are used for evaluation. The accuracy, F-score and AUC were treated as evaluation indices. For the definition of each index, see [6]. It is clear that the accuracy, F-score and AUC do not deviate greatly, and the effect of polynomial approximation is minor. Refer to [7] [8] for details on other data sets.

4 Summary

This paper describes SPHERE, a long-term usable homomorphic cryptography technology developed by the Security Fundamentals Laboratory, and the building of a privacy-preserving logistic regression analysis system as its application. Although omitted in this paper, besides logistic regression analysis, the technology was used for linear regression analysis and biometric authentication [1].

While data collection/analysis by IoT, etc. has become easy, protecting personal information from being leaked from the data is a pressing issue. We hope that the cryptographic technology and the applied technology that we developed will serve as one of the solutions for such issues.

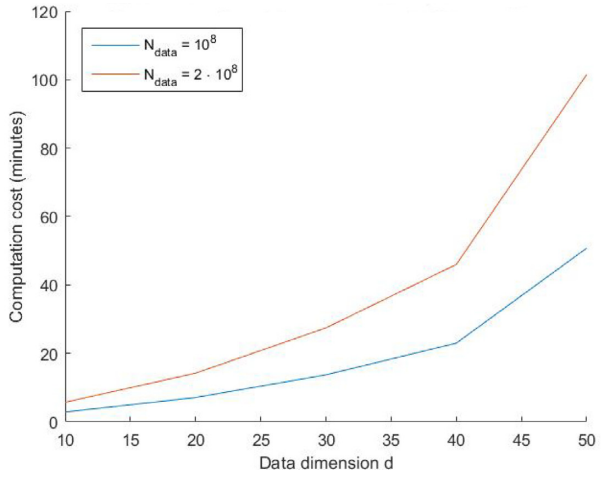


Fig. 5 Computation time of server for large-scale simulated data set

Table 3 Comparison of accuracy, F-score and AUC, between original function and quadratic polynomial approximation

Dataset	Logistic Regression over	Cost Function	Accuracy	F-score	AUC
Pima	Unencrypted data	Original	80.2%	0.688525	0.873653
	Encrypted data	Approximate	80.7%	0.694215	0.876347
SPECTF	Unencrypted data	Original	79.1%	0.876972	0.783333
	Encrypted data	Approximate	75.4%	0.851613	0.761628

References

1 Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Fast and Secure Linear Regression and Biometric Authentication with Security Update," IACR Cryptology ePrint Archive 2015, 692, 2015.

2 N. Shinohara, Y. Aono, and T. Hayashi, "Recent Developments in Security Evaluation Technologies of Cryptosystems," Special issue of this NICT Journal, 7–1, 2016.

3 Z. Brakerski, and V. Vaikuntanathan, "Efficient Fully Homomorphic Encryption from (Standard) LWE," IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, pp.97–106, 2011.

4 P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," Advances in Cryptology– International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT '99, pp.223–238, 1999.

5 UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/

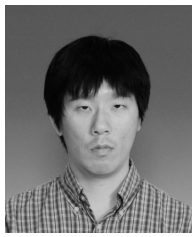
6 H. Nakagawa, "Machine Learning SCHOOL OF ENGINEERING THE UNIVERSITY OF TOKYO," SCHOOL OF ENGINEERING THE UNIVERSITY OF TOKYO, 2015. (in Japanese)

7 Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Scalable and Secure Logistic Regression via Homomorphic Encryption," 6th ACM on Conference on Data and Application Security and Privacy, CODASPY 2016, pp.142–144, 2016.

8 Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Privacy-Preserving Logistic Regression with Distributed Data Sources via Homomorphic Encryption," accepted to IEICE Transactions on Information and Systems, vol.E99-D, no.8, 2016.

Le Trieu PHONG, Ph.D.

Senior Researcher, Security Fundamentals Laboratory, Cybersecurity Research Institute
Design and Analysis of Cryptographic Protocols



Yoshinori AONO, Ph.D.

Researcher, Security Fundamentals Laboratory, Cybersecurity Research Institute
Cryptanalysis, Lattice Theory

Takuya HAYASHI, Ph.D.

Researcher, Security Fundamentals Laboratory, Cybersecurity Research Institute
Cryptanalysis, Efficient Implementation

Lihua WANG, Ph.D.

Senior Researcher, Security Fundamentals Laboratory, Cybersecurity Research Institute
Design and Analysis of Cryptographic Protocols