Cloud Computing Algebraic Homomorphic Encryption Scheme

Reem Alattas
Department of Computer Science & Engineering,
University of Bridgeport,
Bridgeport, CT, USA

ABSTRACT: Although cloud computing is growing rapidly, a key challenge is to build confidence that the cloud can handle data securely. Data is migrated to the cloud after encryption. However, this data must be decrypted before carrying out any calculations; which can be considered as a security breach. Homomorphic encryption solved this problem by allowing different operations to be conducted on encrypted data and the result will come out encrypted as well. In this paper, we propose the application of Algebraic Homomorphic Encryption Scheme based on Fermat's Little Theorem on cloud computing for better security.

KEYWORDS: Cloud computing, homomorphic encryption, security, cryptography, algebraic homomorphism, Fermat's Little Theorem.

1 INTRODUCTION

Cloud computing opens up a new world of opportunities, but mixed in with these opportunities are numerous security challenges that need to be considered and addressed. Among these challenges are availability, third party control, and data security. Data in the cloud is usually globally distributed which raises concerns about jurisdiction, data exposure, and privacy. If all data stored in the cloud was encrypted, that would effectively solve many issues. However, a user would be unable to leverage the power of the cloud to carry out computation on data without first decrypting it, or shipping it entirely back to the user for computation. The cloud provider thus has to decrypt the data first, perform the computation then send the result to the user.

Homomorphic encryption schemes allow the transformation of cipher-text $C(m)$ of message $m$, to cipher-text $C(f(m))$ of a computation/function of message $m$, without disclosing the message. Therefore, the user could carry out any arbitrary computation on the hosted data without the cloud provider intervention.

In this paper, we propose applying Algebra Homomorphic Encryption Scheme Based on Fermat's Little Theorem (AHEF) on cloud computing to solve the data security and third party control issues. AHEF is based on the concept of fully homomorphism and Fermat's little theorem.

This paper structure is as follows: related work and approaches are discussed in section 2. Then, section 3 gives a brief overview of homomorphic encryption and section 4 introduces the application of AHEF on cloud computing. The scheme of the new methodology is described in Section 5. Finally, we give a short summary of our contributions in section 6.

2 RELATED WORK

In 1978, Ronald Rivest, Leonard Adleman and Michael Dertouzos introduced for the first time the concept of Homomorphic encryption. Since then, little progress has been made for almost 30 years. The encryption system of Shafi Goldwasser and Silvio Micali, that was proposed in 1982, was an additive Homomorphic encryption, but it could encrypt only a single bit. In the same notion, Pascal Paillier proposed a provable security encryption system in 1999 that was also an
Cloud Computing Algebraic Homomorphic Encryption Scheme

additive Homomorphic encryption. Few years later, in 2005, Dan Boneh, Eu-Jin Goh and Kobi Nissim invented a security system that can perform an unlimited number of additions but only one multiplication.

Most recently, Craig Gentry proposed the first fully homomorphic encryption scheme in 2009. That system evaluates an arbitrary number of additions and multiplications; and thus computes a function of any type on the encrypted data.

The application of fully homomorphic encryption is an important brick in cloud computing security. Generally, we could outsource the calculations on confidential data to the cloud, while keeping the secret key to decrypt the result of calculation.

3 HOMOMORPHIC ENCRYPTION

The proposed algebraic homomorphic encryption scheme is based on the concept of fully homomorphism, and uses a subset of it. It is also based on Fermat’s little theorem and Fraction Module.

Fermat’s little theorem is one of the four number theorems. It states that if \( p \) is a prime number, then for any integer \( a \), the number \( a^p - a \) is an integer multiple of \( p \). In the notation of modular arithmetic, this is expressed as:

\[
 a^{p-1} \equiv a \pmod{p} \tag{1}
\]

If \( a \) is not divisible by \( p \), Fermat’s little theorem is equivalent to the statement that \( (a^{p-1}) - 1 \) is an integer multiple of \( p \):

\[
 a^{p-1} \equiv 1 \pmod{p} \tag{2}
\]

Fraction Module is simply a new operation. When discussing homomorphic encryption in this paper, we call this operation similar module operation, and use the symbol \( \text{smod} \) to present it.

4 ALGEBRA HOMOMORPHIC ENCRYPTION SCHEME BASED ON FERMAT’S LITTLE THEOREM

Xiang and Cui came up with the Algebraic Homomorphism Encryption Scheme based on Fermat’s Little Theorem (AHEF), which can be described as follows:

1. Select two large secure prime numbers \( p \) and \( q \). Let \( N = pq \), such that \( p \) and \( q \) are secret, and \( N \) is public.
2. A rational number \( x \) can be expressed as the fraction form \( x = \frac{x_a}{x_b} \) such that the numerator \( x_a \) is an integer, and the denominator \( x_b \) is a positive integer.
3. Select a random integer \( r \). The encryption algorithm is \( E(x) \), and the encrypted cipher text is:

\[
 c = E(x) = f \text{mod} \left( \frac{x_a}{x_b} r^{(p-1)+1}, N \right) \tag{3}
\]

4. Decryption algorithm \( D() \), such that

\[
 x = D(c) = f \text{mod}(c, p) \tag{4}
\]

A fully homomorphic encryption scheme, such as AHEF, must respect both addition and multiplication operations as shown below.

4.1 MULTIPlicative HOMOMORPHISM

Let \( x \) and \( y \) be rational numbers, then AHEF meets the multiplicative homomorphism, i.e.

\[
 E(xy) = f \text{mod}(E(x)E(y), N) \tag{5}
\]

Or

\[
 xy = D(E(x)E(y)) = f \text{mod}(E(x)E(y), p) \tag{6}
\]

4.2 ADDITIVE HOMOMORPHISM

Let \( x \) and \( y \) be rational numbers, then AHEF meets the multiplicative homomorphism, i.e.

\[
 E(x + y) = f \text{mod}(E(x) + E(y), N) \tag{7}
\]

Or

\[
 x + y = D(E(x) + E(y)) = f \text{mod}(E(x) + E(y), p) \tag{8}
\]
4.3 **Mathematical Example**

A simple example to verify the nature of algebraic homomorphism of AHEF is given below.

Selecting $p = 173$, $q = 199$, then $N = pq = 34427$.

Let $x = 2.4$ and $y = -1.75$. Now, we will express $x$ and $y$ as fractions: $x = \frac{12}{5}$, $y = -\frac{7}{4}$.

Then, we will randomly select $r_x = 17$, $r_y = 26$. AHEF can be used to encrypt $x$ and $y$:

$$E(x) = f_{mod}\left(\frac{12}{5}, N\right) = \frac{28730}{18170}$$

$$E(y) = f_{mod}\left(-\frac{7}{4}, N\right) = \frac{-28379}{13671}$$

4.3.1 **Multiplicative Homomorphism**

$$D(E(x)E(y)) = f_{mod}(E(x)E(y), p)$$

$$= f_{mod}\left(\frac{28730}{18170} \times -\frac{28379}{13671}, p\right)$$

$$= f_{mod}\left(-\frac{815328670}{248402070}, p\right)$$

$$= smod(-815328670, 173)$$

$$= smod(248402070, 173)$$

$$= -\frac{84}{20} = xy$$

4.3.2 **Additive Homomorphism**

$$D(E(x) + E(y)) = f_{mod}(E(x) + E(y), p)$$

$$= f_{mod}\left(\frac{28730}{18170} + -\frac{28379}{13671}, p\right)$$

$$= smod(smod(28730 \times 13671, p) + smod(18170, p), p)$$

$$= smod(392767830, 173) + smod(-515646430, 173, 173)$$

$$= smod(248402070, 173)$$

$$= smod(48 + (-35), 173)$$

$$= \frac{13}{20} = x + y$$

The security of AHEF algorithm is based on the difficulty of dividing by a large integer. Due to the random number being used in the encryption process, for the same plaintext $x$, the two encrypted results are not the same, i.e. $E1(x) \neq E2(x)$, but $D(E1(x)) = D(E2(x))$. This feature guarantees that users cannot infer the original data through statistical laws. More security properties can be found in [1].
5 AHEF SCHEME

As shown in Fig. 1, the process will start by sending the encrypted data to the cloud provider. The user can access the encrypted data on the cloud. Moreover, she can do calculations on that encrypted data, get the encrypted result. Then, decrypt the result on premise for better security.

6 SUMMARY

In this paper, AHEF algorithm was applied to cloud computing in order to carry out different calculations on encrypted data without decryption. The obtained result is encrypted as well and can be decrypted securely on premise.

REFERENCES


