International Journal of Computer Applications (0975 - 8887)
Volume 131 - No.17, December 2015

Review on HEROIC Framework: Homomorphically EncRypted One Instruction Computer

Sreelekshmi S.
PG Scholar
Dept. of Computer Science
College of Engineering Perumon

Devi Dath
Assistant Professor
Dept. of Computer Science
College of Engineering Perumon

ABSTRACT

Outsourcing to cloud brings a new face to computation. In outsourcing, maintenance cost as well as upgrade cost is low to zero. Outsourcing's features are held back by data privacy concerns. In cloud, privacy of data can be assured by encryption schemes. Common encryption schemes will decrypt the data before processing and re encrypt data after processing, key sharing required. The sensitive data inside the processor are vulnerable to eavesdropping and other attacks.HEROIC Framework (Homomorphically EncRypted One Instruction Computer),a secure architecture for processing data in encrypted form is introduced as a promising solution to the security and privacy concerns. It is single instruction architecture which rules out the need for storing private key inside the processor. In this framework a variant of Paillier encryption scheme is used for homomorphically encrypting both data and instruction.

General Terms

Homomorphic encryption,Paillier scheme

Keywords

Outsourcing,Cloud computing, Single instruction architecture

1. INTRODUCTION

Outsourcing computations[1] are becoming an effective option due to decreased cost of cloud services. Cloud can be defined as anything that renders a service through internet. Outsourcing enables a user to offload the computational work to untrusted third parties. The user's data given to third party in the cloud are evaluated with a function provided by the user and the computed result is returned with a proof of correctness. Outsourcing can be reliable only if the user data are secure in the cloud and results given are trustworthy. The privacy and security concerns about data also increased with the rise of new technology.

The concerns led the data owner to choose different cryptography schemes so as to secure the data.Cryptography is the most commonly used security measure for securing data from different attacks. For general encryption schemes the server's system should contain the private key and the encrypted data before processing need to be decrypted. This makes the data inside the processor vulnerable to many kinds of attacks like eavesdrop. The processor is vulnerable to large number of attacks like using hardware Trojans to save or transmit sensitive data from processor. To make the sensitive data secure it should be encrypted in such a way that it can be processed in the encrypted form. Homomorphic encryption is the best way to manipulate data in encrypted form.The use of homomorphic scheme in outsourced computation is shown in Fig. 1. In homomorphic encryption[2] the encrypted data is used for various computations and the result when decrypted gives meaningful results.Different homomorphic encryption can be used based on the which functionality it support. The different homomorphic encryption scheme include partially homomorphic and fully homommorphic. This framework uses partially homomorphic scheme

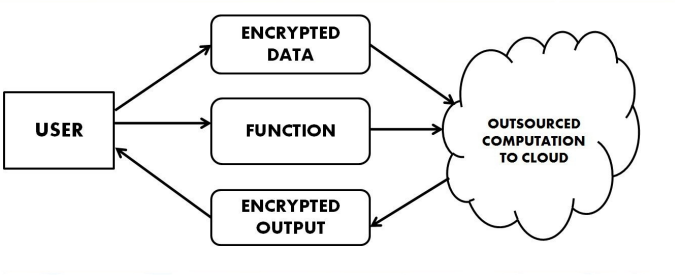

Fig. 1. Homomorphic encryption in outsourced computation

In homomorphic encryption scheme both data and code can be encrypted form. The processing of data in encrypted form is not supported by the general processor architectures like CISC(Complex Instruction Set Computer) and RISC(Reduced Instruction Set Computer). for this purpose a single instruction architecture s used. This architecture is defined to be Turing complete. This makes it the most appropriate one for this framework.This use of Homomorphic encryption and single instruction makes the HEROIC Framework[3] secure and flexible.

2. REVIEW ON HEROIC FRAMEWORK

2.1 Single Instruction Architecture

In HEROIC Framework the instructions are in the encrypted form. The existing architectures like RISC and CISC does not support the processing of encrypted instructions. These architectures were designed for performance and efficiency, security was never a de-

sign option. These architectures bring down the security of sensitive data as they may lead to eavesdrop. One of such Trojan attack is the use of hardware Trojans[4] inside the processor. This type of Trojan was developed by modification of the material's physical composition. Like these virtually undetectable Trojans large number of attacks are possible, which are of great threat to the sensitive data.

The need for different op codes are ruled out by the use of single instruction architecture. The OISC(One Instruction Set Computer) supports computations in the encrypted domain. Single instruction architectures should be Turing Complete[5] so as to use it for computational application.The idea of Turing Complete is that it should be capable of recognizing any algorithm. The rules[6] that define Turing Completeness are sequence rule, selection rule and rule. The sequence rule defines the flow of the control through instructions. The selection rule brings in the idea of decision making, whether control should move to next instruction or jump to another set of instructions.Repetition rule helps in flow of control without a decision making stage.

The four basic computer instructions used to follow the above mentioned rules are LOAD, STORE, INCrement and GOTO. One instruction set computer supports only one instruction. Micro operations of a single instruction can be performed using the four instructions. These properties make Single Instruction Architecture and alternative of RISC. A wide varieties of OISC variants are available. Addleq, subleq, pleg are some commonly used variants. HEROIC framework support both subleq and addleq.

The Subleq[7] micro operation is a set of instruction given as

Mem[B]=Mem[B]−Mem[A]
if Mem[B]<= 0then goto C
else goto next instruction

The memory value of B is subtracted from A and based on the output the next instruction to be executed is decided. If value is less or equal to zero the next instruction stored in C is to be executed otherwise the succeeding instruction is to be executed. The flow of control in subleq instruction set is shown in Fig.2.

2.2 Paillier Scheme

Homomorphic encryption scheme provide the data owner an extensive range of security. The sensitive data need not to be disclosed to third party at any stage of outsourcing computation. Computations are performed in the encrypted form itself. This scheme is widely applied in various areas like e-voting, private information retrieval. Based on the functionality supported it is classified into two, partially homomorphic and fully homomorphic. Partially homomorphic support less number of operations, either modular addition or modular multiplication or polynomials to a degree. Fully Homomorphic being less restricted is more powerful and flexible. Partially homomorphic schemes are commonly used than fully homomorphic due to the less over head and availability of well defined schemes like Paillier scheme, Goh's scheme etc. Lately, new fully homomorphic encryption schemes are introduced like Gentry scheme but the criticism of high overhead is holding it back from practical use.

The most commonly used partially homomorphic encryption are RSA scheme[8], Paillier scheme[9], Goh's scheme[10] etc. The main difference between Paillier scheme and Goh's scheme is that Paillier only support modular addition whereas Goh's support arbi-

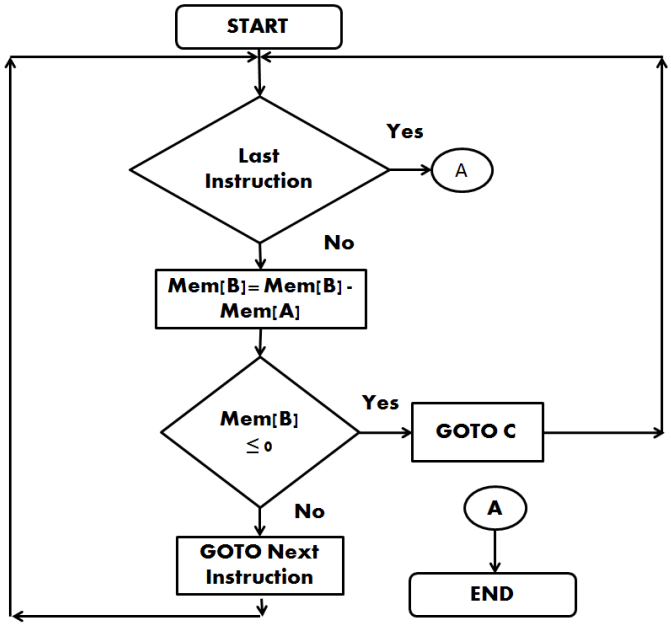

Fig. 2. Flowchart of subleq instruction set

trary number of modular addition and one modular multiplication. The Paillier Cryptosystem found and named after Pascal Paillier who is a French researcher. The Paillier scheme being additive homomorphic can be defined as

$$Dec[Enc(n_1) \triangle Enc(n_2)] = (n_1 \bullet n_2)$$

where $\triangle$ gives modular multiplication and $\bullet$ is the modular addition.It is a three stage scheme.

(1) Key Generation
  (a) Choose two prime numbers p and q such that $gcd(p, q, (p-1)(q-1)) = 1$
  (b) Compute n and λ
  $n = pq$
  $\lambda = \frac{(p-1)(q-1)}{gcd(p-1),(q-1)}$
  λ= Carmichael's function
  (c) Select generator g, either by
    i. choose from a set $Z^*_{n^2}$ which satisfy $gcd(\frac{g^{\lambda} mod n^2 - 1}{n}, n) = 1$
    ii. choose α and β from Z^*_n, compute $g = (\lambda n + 1)\beta^n \bmod n$
  (d) compute modular multiplicative inverse $\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$

  The public key generated for encryption is$(n, g)$and the private key generated for decryption is (λ, μ)

(2) Encryption
  $c = g^m r^n \bmod n^2$
  where m is the message and r a number chosen at random
(3) Decryption
  $m = (L(c^{\lambda} mod n^2)\mu) \bmod n$

2.3 HEROIC Framework

Heroic framework is capable of executing the encrypted program with the given encrypted data. The data owner sends the homomorphically encrypted data to the server. The program to be executed is converted to single instruction form with the help of a subleq assembly. The program as well as the addressing values are in the encrypted form. On receiving both data and code the server computation and return the encrypted output to the data owner. Some issues faced along with the effective solutions implemented in HEROIC framework are discussed below.

2.3.1 Encrypted Memory Addressing. In computation one instruction is allowed to manipulate the values of arguments of other instructions. In HEROIC Framework the use of single instruction architecture and above mentioned criteria brings out the need for encrypted memory addressing. As the instruction arguments are indirectly referring to the various memory locations the memory addresses need to be encrypted. However this framework support this condition of encrypted memory addressing which solves an important design issue for this system.

2.3.2 Out of range error correction. It is easier to explain this issue with help of an example. A 16 bit architecture can be used for this purpose. Addition of two numbers say -30 with -1 .The 2's complement of the two numbers are computed as Paillier scheme support only modular addition. For -30 it will be $(2^{16} - 30)$ and for -1 will be $(2^{16} - 1)$. The expected output is $(2^{16} - 31)$ but result obtained is $(2^{17} - 31)$.This type of error is called out of range error. It is detected with the help of a out of range look up memory which checks for a value above $(2^{16} - 1)$.When the value is greater than $(2^{16} - 1)$ a modular multiplicative inverse is added to (2^{16}).

2.3.3 Arguments Matching. An important concern is the indistinguishably o data from instruction argument. The argument for a single instruction should be separately matched to identify the instruction. Another important concern is the encrypted addressing used. The permuted value of each argument is present in the encrypted domain as compared with unencrypted domain. This issue is solved in HEROIC Framework by mapping the address of next instruction to the current instruction. Program counter's value can be used for this purpose.

2.3.4 Subtraction. In order to perform subtraction the framework requires an ALU that supports the framework. In Paillier scheme it performs modular multiplication of encrypted so as to get modular addition of the actual data. Therefore modular multiplicative inverse of the subtrahend is homomorphically added to the minuend to obtain the desired homomorphic subtraction.Another problem is that the multiplicative inverse of an encrypted data cannot be obtained algebraically. For this purpose a inverse look up register is used. Another method to perform subtraction is to use the one instruction set computer variant addleq. Even though it is a possible method, the difficulty in programming makes it complex.

2.3.5 Memory Addressing. With the increase in security parameter the size of encrypted data address also increases. This condition will result in uneconomical computations and these long addresses are unnecessary. Three solutions are put forward by this framework to handle addressing size. In general an unencrypted domain supports a 16 bit addressing where as the Paillier scheme security parameter could be 2048 bits wide. The lower bits which could distinguish all the memory addresses can be used rather than whole address. Another solution is found from the facts that the 2^{22} addresses 2^{16} are only used for storing data and the highest security parameter the maximum address width is 2060 bit. These gives an idea to reduce the size for memory addressing by choosing 22 bits addressing size and 16 bits content width for the first memory which point to next memory.In the second memory the content width will be large enough to store encrypted value where as the addressing size will be 16 bits not 22 bits. Furthermore an extra unit could be used to reduce memory sizing, it is the CRU (Collision Resolution Unit). CRU which requires a smaller memory converts a 22 bit address to 16 bit index for secondary memory.

2.3.6 Jump Decision. Another important issue is to determine the flow of control through instructions As the ALU is also encrypted, determining the sign of output is not mathematically possible. The framework brings in a solution. that is to use a sign look up memory which returns any encrypted value's mathematical sign.

3. CONCLUSION

The new secure framework called HEROIC Framework is introduced to secure data while outsourced to cloud. The features like single instruction architecture and homomorphic makes the system secure and flexible. The framework is capable of executing encrypted programs written in subleq assembly.For privacy and confidentiality in cloud computing, HEROIC Framework is an effective solution. Future works can done to improve the speed and efficiency of system.Different homomorphic schemes can be applied to choose the most reliable one.

4. REFERENCES

[1] Wikipedia,https://en.wikipedia.org/wiki/Outsourcing-computing

[2] C. Fontaine and F. Galand, "A survey of homomorphic encryption for nonspecialists," EURASIP Journal on Information Security, vol. 2007, no. 1, pp. 26-35, 2007.

[3] Nektarios Georgios Tsoutsos, Michail Maniatakos "HEROIC: Homomorphically EncRypted One Instruction Computer" 2014

[4] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy dopant-level hardware trojans," in Cryptographic Hardware and Embedded Systems Workshop, 2013, pp. 197-214.

[5] A. Teller, "Turing completeness in the language of genetic programming with indexed memory," in Proc. 1st IEEE Conf. Evol. Comput., Orlando, FL, USA, 1994, pp. 136-141.

[6] C. Bohm and G. Jacopini, "Flow diagrams, Turing machines and languages with only two formation rules," Commun. ACM, vol. 9, no. 5, pp. 366-371, 1966.

[7] O. Mazonka and A. Kolodin, "A simple multi-processor computer based on subleq," arXiv preprint arXiv:1106.2593, 2011

[8] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol. 21, no. 2, pp. 120-126, 1978.

[9] HSR Hochschule fr Technik Rapperswil, Homomorphic Tallying with Paillier Cryptosystem, Seminar on e Voting

[10] X. He, M.-O. Pun, and C.-C. Kuo, "Secure and efficient cryptosystem for smart grid using homomorphic encryption," in Innovative Smart Grid Technologies, 2012, pp. 1-8.