# Enhancing IoT Security by Homomorphic Algorithm

[1]V. Murugesan, [2] M. K. S. Mahesh, [3] M. Siddharth
[1][2][3]Second year-Computer Science and Engineering
Jansons Institute of Technology, Coimbatore, Tamil Nadu-India

***Abstract:*** **In general, by the concept of Internet of Things (IoT), everything real becomes virtual, which means that every person and thing has a readable, addressable and locatable counterpart on the Internet. This scenario must be protected at most, which is less expected by the current security level of IoT. Encryption had always proven to be the best way of securing the data in any system. But the current Light weight encryption in IoT provides security only in the client side as it is end to end encryption but not on the server side. This vulnerability can be patched by the advent of Homomorphic encryption which is proven below to be the best encryption technique on the server side. This latest technique allows performing operations even in the encrypted, irrespective of its type of encryption.**

***General Terms: IoT, Security, Homomorphic Encryption Algorithms, Homomorphic Encryption Schemes.***

***Keywords: Paillier algorithm, Cryptography, Homomorphic Encryption.***

## 1. INTRODUCTION

On a new computing environment called "Internet of Things (IoT)", a lot of constrained devices are connected to the Internet. The devices interact with each other through the network and provide new environment to us. In order to experience this new environment, security of constrained end nodes is important. Several significant obstacles remain to fulfil the IoT vision. First among them was a security issue. The Internet and its users are already under various attacks, and a growing economy threads with various business models that undermine the Internet's ethical use is fully focused on exploiting the current version's vital weaknesses. This does not bode well for the IoT, which incorporates many induced devices. Irony, realizing the IoT future is likely to spark novel and ingenious poisonous models. The dispute is to prevent the growth of such models or at least to mitigate and limit their impact.

## 2. COPING WITH OLD AND NEW THREAD

Not surprisingly, even a staggered approach to developing the IoT presents a daunting task for secureness. What security measures are possible as billions of intelligent things cooperate with other real and virtual entities in random and undeterminable ways? The IoT's highly scattered nature and use of fragile engineering, such as narrow-function embedded devices in non -private areas, create very weak links that poisonous objects can exploit. Easily accessible entities in unsecured zones, such as parks, city, and streets are vulnerable to exterior harm. Like including botnets, some entities would try to hinder services from the inside.

Additional threats include the existence of a domino effect between intertwined services and user profiling through data collection and other methods.

To avoid these threats, the IoT must have strong security foundations built on a holistic view of security for all IoT elements at all stages—from the identification of objects to the purely of services, from the acquiring of information to the governance of the whole substructure. All protection mechanisms must consider each object's lifecycle and services from the very beginning of that object's existence [1].

## 3. DATA AND PRIVACY

Privacy is one of the most sensitive subjects in any discussion of IoT protection. The data availability detonation has formed Big Brother-like objects that profile and track users without their permission. The IoT's anytime, anywhere, anything nature could easily turn such practices into a bad state. Users can access the unprecedented number of customized services, all of which would produce larger data, and the surroundings itself would be able to acquire information about users automatically. Although a dystopia is the bad-case situation, the IoT could absolutely exacerbate a range of unwanted situations. Social media accounts already affect a user's employability and personal interface. Imagine exponential progression for such exposure opportunities.

### *3.1 Privacy by design.*

One viable solution is seclusion by design, in which peoples would have the tools they need to manage their own data. The answer is not at a great distance from current reality. Whenever people produce a data segments, they can even use non-static consent tools that permit certain services to access as little or as much of that data as on wish. Taking that new knowledge a little further, a person in Central Park could provide a GPS service with the resultant information that he's in Washington City, but not that he's in a particular park.

*3.2 Transparency.*

Transparency is also important, since users should know which objects are managing their data and how and when those entities are depend on it. Stakeholders like service data providers must be part of this expression, by which they might make take it or leave it license statement obsolete. Businesses will change their services according to the amount of personal data the user provides.

*3.3 Data management:*

A huge issue is deciding who control the secrets. In technical terms, cryptographic algorithm and protocols protect data throughout the facilities life span, but some data might lack the resources to control such mechanisms and algorithm. Otherwise, one data management policy will not fit all environment. At the same time, there must be statement on how to manage various kinds of data as well as some statement-enforcement mechanism or algorithm. Constructing such data management statements and enforcing them is not important. It needs translating, interpreting, and optimally reconciling a list of rules, each of which may be in a many different language. And any statement should arrange with legislation on data securing, w makes a great change.

## 4. LIGHTWEIGHT ENCRYPTION:

Cryptographic technologies are advancing: new methods on attack, design and usability's are extensively studied. One of the state-of-the-art techniques is "Lightweight Cryptography (LWC)"[8]. Lightweight cryptography is a cryptographic algorithm or protocol tailored for implementation in constrained environments including RFID tags, sensors, contactless smart cards, health-care devices and so on [2]. This method of Crypting works well on securing the data from client side as it provides an efficient end to end encryption [9]. But this doesn't make the server side security much rigid and ubiquitous for transferring data among client.

## 5. HOMOMORPHIC ALGORITHM

Homomorphic encryption is the encryption scheme which means making manipulations on the encrypted data. Homomorphic encryption can be applied in any system by using various public key algorithms. When the data from the devices is transferred to the non-private area, there are many encryption mechanism to secure the operations and the collection of the data. But to manipulate data located on different remote server and to protect privacy, homomorphic mechanism is useful that allows the operations on the cipher system, which can produce the same solution after calculations as the working directly on the raw data [3].

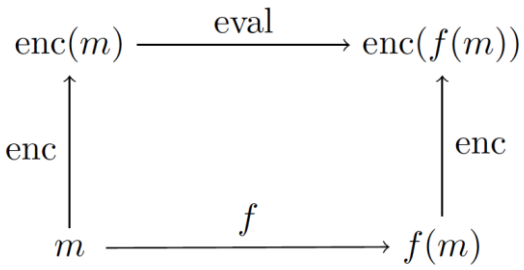


Fig 1: Manipulating Encrypting data

## 6. FUNCTIONS OF HOMOMORPHIC ENCRYPTION

In general, Homomorphic Encryption H is a group of 4 functions [11] as shown in figure 1.

H = {Key Generate, Encrypt, Decrypt, Evaluate}

1. Key generation: client will generate pair of keys public key pk and secret key sk for encryption of plaintext.
2. Encryption: Using secret key sk client encrypt the plain text PT and generate Esk (PT) and along with public key pk this cipher text CT will be sent to the server.
3. Evaluation: Server has a function f for doing evaluation of cipher text CT and performed this as per the required function using pk.
4. Decryption: Generated Eval (f (PT)) will be decrypted by client using its sk and it gets the original result.

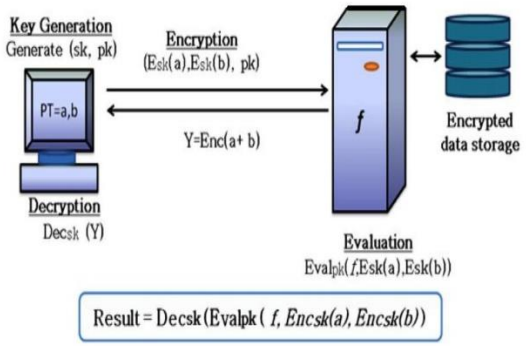


Fig 2: Homomorphic Encryption

## 7. HOMOMORPHIC ENCRYPTION IN IOT ASPECTS:

There are many types of algorithm like light weight encryption that focus only on the client side security. But IoT doesn't concerned only on the client (Things) but also the server that connects the devices. Hence this homomorphic encryption provides a rigid security while accessing the data in the server side and this makes easy manipulation of encrypted data in the server side.

Suppose there are two data from two objects OBJ1 and OBJ2 such that:

OBJ1 = $g_{m1}x_{1n} \bmod n_2$

OBJ2 = $g_{m2}x_{2n} \bmod n_2$

OBJ1·OBJ2 = $g_{m1}x_{1n} \cdot g_{m2}x_{2n} \bmod n_2$

Additive Property is: $g_{m1+m2}(x_1x_2)_n \bmod n_2$

Let us consider Paillier Cryptosystem similar to IoT working system.

**1. Key generation:**

Step 1: n = pq, the RSA modulus
Step 2: λ = lcm (p − 1, q − 1)
Step 3: g ϵ Z /n2 Z s.t. n|or dn2 (g)
Step 4: Public-key: (n, g), secret key: λ, μ

**2. Encryption of m:**

Step 1: m ϵ {0, 1... n − 1}, a message
Step 2: h ϵR Z/n Z
Step 3: c = gm hn mod n2, a cipher text

**3. Decryption of c:**

m = L (c ג mod n2) L (g ג mod n2) −1 mod n
The constant parameter,
L (gλ mod n2)-1 mod n or L (gα mod n2)-1 mod n Where g=1+ n mod n2 can also be recomputed once for all.

## 8. THE PROPOSED MODEL

*8.1 IoT Gateways:*

The IoT gateways are computational devices, Linux-based board computers (e.g., the Raspberry Pi [20] or the Beagle bone board [21]) that have the essential networking interfaces for communicating with sensors (e.g., Bluetooth and ZigBee interfaces) and can also communicate with the Internet using wireless (e.g., Wi-Fi) or wired physical connection. The gateways tends to have best computational values (usually come with at least 1GHz ARM processor and 512Mb or RAM memory) and host a complete operating system that provides PKI tools. Through appropriate important setting of the network interfaces and the Linux OS networking the gateways can apply PKI encryption to the incoming data and then forward the latter to web-based medical applications using Web
Services or other real-time communication technologies (e.g., Web Sockets).

*8.2 Cloud (Back-end) Infrastructure:*

Cloud Computing is a technique for providing easy and reliable, on demand network access to a shared group of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. The latter features make Cloud computing a very suitable model for building back-end infrastructures that support data management and visualization of IoT devices. In addition, Cloud resources can provide the essential requirements for Homomorphic encryption/decryption (like computational resources) and encryption/decryption key management [6].

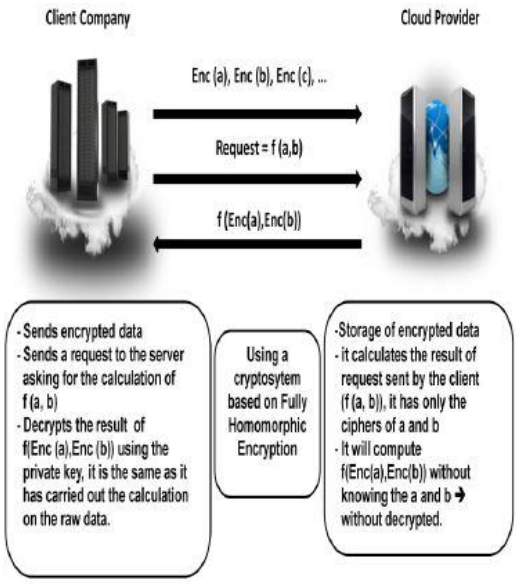


Fig 3: Encryption in server side

The data collected from varies devices are stored in cloud as encrypted data [10]. Hence while processing those data for making similar devices connect together. The data must be decrypted in the server that causes technological difficulties and also leading it in serious security issues. Hence by using homomorphic encryption in the server side, the need for decryption of data for processing is avoided and security issues in the server side is prevented.

## 9. PROPERTIES OF HOMOMORPHIC ENCRYPTION SCHEMES

Homomorphic encryption schemes have some interesting arithmetic and logical properties. Let us discuss some of the interesting properties of homomorphic algorithm.

*9.1 Re- randomizable encryption/re-encryption:*

Re-randomizable cryptosystems (Groth, 2004) are probabilistic cryptosystems with the additional property that given the public key and an encryption) of a message under the public key and a random number K it is possible to efficiently convert) 01 into another encryption) that is perfectly indistinguishable from a fresh encryption of m under the public key. This property is also called re-encryption.

*9.2 Random self-reducibility:*

Along with the possibility of re-encryption comes the property of random self-reducibility concerning the problem of computing the plaintext from the cipher text. A cryptosystem is called random self-reducible if any algorithm that can break a non-trivial fraction of cipher texts can also break a random instance with significant probability [7].

*9.3 Confirmable encryptions / fair encryptions:*

If an encryption is confirmable, it results in a mechanism to check the correctness of encrypted data without compromising on the protection of the entities. For

example, this is useful in voting schemes to convince any observer that the

## 10. SOME APPLICATIONS OF HOMOMORPHIC ENCRYPTION SCHEMES

*10.1 Protection of mobile agents:*

One of the most interesting applications of homomorphic encryption is its use in protection of mobile communications. As we discussed already, a homomorphic encryption scheme on a special non-abelian group would lead to an algebraically homomorphic cryptosystem on the finite system. Since all traditional computer architectures are based on binary strings and only require multiplication and addition, such homomorphic encryption system will offer the possibility to encrypt a whole program so that it is always executable.

***10.2 Multiparty computation:***

In multiparty computation schemes, several parties are interested in computing a common, public function on their inputs while keeping their individual inputs private. This problem belongs to the area of computing with encrypted data. Usually in multiparty computation protocols, we have a set of n >= 2 players whereas in computing with encrypted data scenarios n = 2. Furthermore, in multi-party manipulation rules, the function that should be calculated is openly known, but in the area of manipulation with encrypted data it is a private input of one party.

*10.3 Secret sharing scheme:*

In secret sharing schemes, parties share a secret so that no individual party can reconstruct the secret form the information pre-exist to it. Though, if some parties unite with one other, they might be able to regenerate the secret. In this strategy, the homomorphic property shows that the composition of the shares of the secret is equivalent to the shares of the composition of the secrets.

*10.4 Threshold schemes:*

Both secret sharing schemes and the multiparty computation strategy are examples of limiting strategy. Threshold schemes can be implemented using homomorphic encryption techniques.

## 11. ADVANTAGES

Homomorphic algorithm emerged as one of the best encryption algorithm in the server side due to handling data in the encrypted state and also due to the rigidity in nature. Because of processing encrypted data this tends to be the best reliable encrypting in cloud.

## 12. CONCLUSION

IoT is regarded as the technology going to rule the future world. In future every objects will be having its own IP address and will be connected to internet for communicating among themselves. As the communication going to be connected among infinite things, the data produced will also be infinite and the traditional methods fails to secure those infinite amount of data. In this case, Homomorphic encrypting algorithm will be greatly use and manage the encrypted data efficiently and reliably with high security concern. This encryption just makes security easier with encrypted data handling technique.

## 13. REFERENCES:

[1].Securing the Internet Of things, vol. 44, no. 9, pp. 51-58, September 2011
[2]. 1. T. Akishita and H. Hiwatari, “Compact Hardware Implementations of the 128-bit Blockcipher CLEFIA.” in *Proceedings of Symposium on Cryptography and Information Security –SCIS 2011 (in Japanese)*, 2011.
[3]. International Journal of Computer Applications (0975 – 8887) Volume 91 – No.8, April 2014
[4] “The raspberrypi board computer.” [Online]. Available: http://www.raspberrypi.org/
[5] “The beagleboard board computer.” [Online]. Available: http://beagleboard.org/
[6]. Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, July 4- 6, 2012, London, U.K.
[7]. Damgard et al., 2010; Sander et al., 1999
[8]. A. Poschmann, “Lightweight Cryptography – Cryptographic Engineering for a Pervasive World.” In *IACR ePrint archive 2009/516*, 2009.
[9]. T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, “The 128-bit blockcipher CLEFIA.” In *Proceedings of Fast Software Encryption – FSE'07* (A. Biryukov, ed.), no. 4593 in LNCS, pp. 181–195, Springer-Verlag, 2007.
[10]. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In Foundations of Secure Computation, pages 169{177. Academic Press, 1978.