# Lightweight Polymorphic Encryption for the Data Associated with Constrained Internet of Things Devices

**Conference Paper** · June 2020

DOI: 10.1109/WF-IoT48130.2020.9221296



**4 authors:**

Indira Kalyan Dutta
Arkansas Tech University
**15** PUBLICATIONS **54** CITATIONS
SEE PROFILE

Bhaskar Ghosh
University of Louisiana at Lafayette
**12** PUBLICATIONS **47** CITATIONS
SEE PROFILE

Albert H. Carlson
Austin Community College
**27** PUBLICATIONS **43** CITATIONS
SEE PROFILE

Magdy Bayoumi
University of Louisiana at Lafayette
**654** PUBLICATIONS **5,105** CITATIONS
SEE PROFILE

# Lightweight Polymorphic Encryption for the Data Associated with Constrained Internet of Things Devices

Indira Kalyan Dutta
Dept. of Computer Engineering
University of Louisiana at Lafayette
Louisiana, USA
Email: indira.dutta1@louisiana.edu

Bhaskar Ghosh
Dept. of Computer Science
University of Louisiana at Lafayette
Louisiana, USA
Email: bhaskar.ghosh1@louisiana.edu

Albert H. Carlson
Research and Development
CipherLoc Corporation
Buda, Texas, USA
Email: acarlson@cipherloc.net

Magdy Bayoumi
Dept. of Computer and Electrical Engineering
University of Louisiana at Lafayette
Louisiana, USA
Email: magdy.bayoumi@louisiana.edu

***Abstract*—The Internet of Things (IoT) is a very new technology that promises to revolutionize modern networks and commercial/industrial/military operations. To date, there has been no serious and in-depth effort to secure the hardware and data associated with IoT nodes. One of the primary methods of data protection is encryption. Recent advances in encryption include the advent of polymorphic encryption methods that promise protection in both classical and the post-quantum environments (PQE) that can keep IoT data safe. In this paper, we describe hardware implementation of a novel lightweight encryption system with an area estimation of 1893 Gate Equivalent, based on CipherLoc polymorphic encryption for the constrained IoT equipment.**

## I. Introduction

In an effort to ease human life, numerous Internet of Things (IoT) devices are being connected every day. IoT is a network of interconnected computing devices, mechanical and digital machines, objects, animals or people, which are provided with unique identifiers and the ability to transfer data over a network and among themselves without requiring human-to-human or human-to-computer interaction [1]. Unfortunately, privacy and security of information or data shared by these devices are often ignored by both the manufacturers and the consumers. As a result, hackers take advantage of these unsecured devices through different attacks. [2].

Encryption is one of the most important means of data protection [3]. To secure the hardware and data associated with these constrained IoT devices, we propose hardware implementation of a novel lightweight encryption system based on CipherLoc polymorphic encryption [4]. The debate about hardware implementation vs. software implementation of cryptographic systems is endless [5]. Both approaches have their own advantages and trade offs. Implementations of software cryptographic systems are cost-effective and more flexible. On the other hand, hardware implementations of cryptographic systems are much faster in operation and once built, they cannot be tampered with easily. The uncontrolled memory access and the vulnerabilities imposed by the operating systems are a few of the reasons why Software cryptographic systems provide much lower level of security than their hardware equivalents [6]. We implemented our design in hardware to have stronger security and better performance. The hardware implementation of novel polymorphic encryption system is designed in a manner which is not only more secure but also area efficient.

Considerable research have been done on AES (Advanced Encryption Standard) [7], which has been considered to be relatively safe for a long time. For hardware implementation, generally the required Gate Equivalent size for a lightweight block cipher is less than 2000 [8] gates. AES designs are not lightweight. An area efficient AES requires area of about 3400 Gate Equivalents [8]. Even though a lightweight AES can be a candidate for securing the constrained IoT devices [9], 8-bit lightweight AES has significantly slower performance in constrained devices than it has in other powerful devices [8]. Apart from resource and performance constraints, AES or any other block cipher is vulnerable to frequency analysis, when they are used for encrypting long files [10][11]. Blocks of information are encrypted thousands of times with the same key, which is a weakness. To minimize the effects of frequency analysis, modes such as CBC (Cipher Block Chaining) [7] were introduced. CBC methods try to make the blocks secure by attempting to randomize the cipher text by an XOR function. Additional modes mean the addition of circuitry, which is not desirable from lightweight perspective. There are methods which can find patterns by analyzing the information bleeding through these blocks with CBC mode [12]. Side channel attacks [13][14] and exposure to some level of collisions can also make AES weak [15] and ineffective for

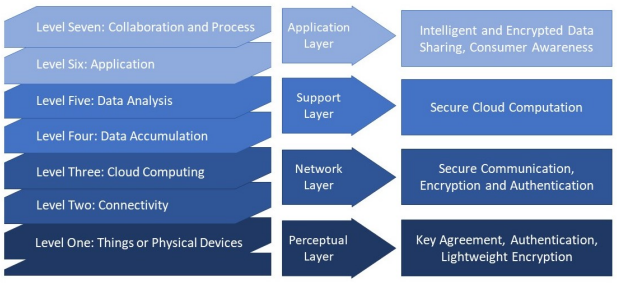


Fig. 1. Security Architecture of IoT

securing constrained and sensitive IoT devices. In our work we will show lightweight polymorphic encryption technology is an attractive alternative for constrained devices.

The Internet of Things (IoT) consists of several interconnected devices that continuously share information and data among each other. It is challenging to provide security for the hardware and data associated with these devices, because of relatively limited capabilities and resources. Our proposed lightweight polymorphic encryption system can be included inside the device from the manufacturing stage. In this paper, we describe the hardware implementation of the chosen Pseudo Random Number Generator (PRNG) and also the overall architecture of the proposed lightweight polymorphic encryption. From our design we could achieve an area estimation of total 1893 Gate Equivalent, which includes the PRNG and the proposed lightweight polymorphic encryption system.

The structure of this paper begins with an explanation of Internet of Things and the security layers of IoT network, to have a better understanding of the area of focus for our work. We then describe the polymorphic encryption scheme. Finally we discuss our design and results in two different subsections. We conclude our paper with future direction of our research and discussion.

## II. Internet of Things

The term ”IoT” was first coined by Kevin Ashton in 1999 [16], explaining that Radio-frequency identification (RFID) is the key ingredient of Internet of Things. According to Statista 2019 [17], by 2030 the number of connected devices globally, will grow to almost 50 billion, about 6 connected devices per person. Before we investigate more on the current scope for security of IoT devices, it is important to take a look at the security architecture of IoT (see Fig.1).

### A. Security Architecture of Internet of Things

*1) Application Layer:* The Application layer has two levels. In level one, the applications make real use of the physical devices data by reporting, analyzing and controlling. In next level collaboration and process take place. The level of security for this layer depends on the application itself. For medical devices or a smart lock high levels of security are needed, because they deal with very sensitive data. But security requirement for a smart bulb can be correspondingly less, because data associated with a smart bulb is not as sensitive as data associated with a medical device [18][19].

*2) Support Layer:* The Support layer includes two levels. The first one is the data accumulation level, which includes ingestion, streaming and storage of data. The next one is data analysis level, which includes data aggregation, data reporting, machine learning and data mining. With the increase of the number of IoT devices, the amount of data generation continues to increase. It is predicted that by 2025 Global Data Generation will hit 175 ZB (Zettabyte), which is 4.4 times more than current Global Data Generation [20]. Storage systems of this size require protection. Identifying the ownership of the data and also knowing when data loses its value are very important.

*3) Network Layer:* The Network layer has two levels. The first is the connectivity level, which includes different communication protocols, edge computing and data processing units [21]. The next level is the cloud computing level. Data transmission of the Network Layer depends on the nature of wireless network or mobile communication protocols. The available power in this layer is not constrained, so strong encryption between client and servers can be established.

*4) Perceptual Layer:* The end devices of the IoT network in the perceptual layer are typically very resource constrained in terms of area, storage capacity and computing power. As a result, it is difficult to integrate a complicated or heavyweight security system for these devices. Denial of service, eavesdropping, data manipulation are common attacks on IoT devices. Authentication and lightweight encryption are two of the most important measures that can be taken to protect these end-nodes [18][19][22].

Our proposed design will focus on the Perceptual layer. Security of the data associated with the end devices with limited resources is our target problem.

## III. Polymorphic Encryption

With increasingly powerful and effective attacks against current standard cryptographic systems, the cryptographic environment is changing. Because of the volume of IoT devices with no or weak security systems, it is a crucial time to consider a versatile solution. Generally, an encryption system has one algorithm every time it is used and one key for the entire encryption session. The definition of Polymorphic Encryption says, that the algorithm and the key should change every time it is used [23]. Today’s cryptographic research and industry is evolving to make its way towards Post Quantum Environment (PQE) and polymorphic encryption system is a promising candidate for the PQE [4].

### A. Polymorphic Encryption Technology

The concept of “shards” is a collection of a small part of a whole. This concept is used to the method of Polymorphic Encryption in this work (see Fig.2)[4]. Shards are continuous portions of small streams of messages, formed from a larger message. Each shard has its own information content and

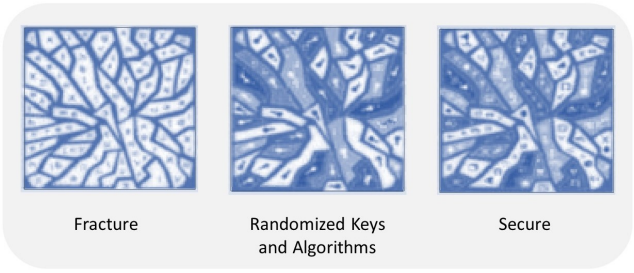


Fig. 2. Polymorphic Encryption: The Shard Concept [4]

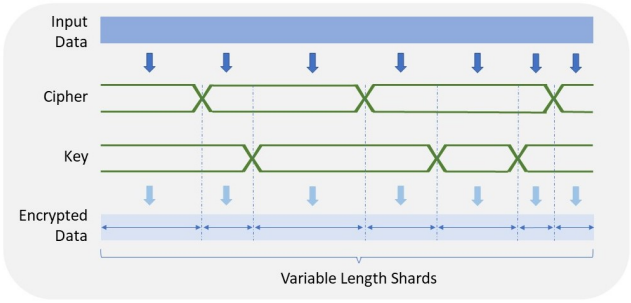


Fig. 3. Time Domain Multiplexing: Cipher-Key Pairs [4]

own unique entropy [4][24][25]. For any shard, the entropy associated with that shard has its "local unicity distance". In cryptography, unicity distance is the length of the original ciphertext needed to break the cipher by reducing the number of possible spurious keys to zero [26]. Inputs when applied to property sets allow us to rule out a certain subset of keys. From the remaining keys, there is only one correct key, the rest of which are known as spurious keys. If the local unicity distance is greater than the size of the shard, there is insufficient redundancy in that portion of the message to attack [24][27]. Shards can have different lengths and lengths can vary within the same message (see Fig.3) [28].

### B. Properties of the Shard Concept

Sharding splits up a single message into multiple segments and keeps the length of the shards less than the unicity distance. Every segment or shard has a unique key and a different cipher algorithm, which creates a series of encryption problems, which must be solved independently to decrypt the whole message. This method can run in a parallel fashion, which helps to reduce latency and accelerate the encryption process. With improvement in technology, the security provided by this versatile and scalable method will improve [4][28]. If there are $S$ shards and $|K|$ is the number of keys, the total key space would be $|K|S$ keys (Eq.1) [27].

$$|K| = \prod_{i=1}^{S} |K_i| \tag{1}$$

As a result, the resulting key space becomes computationally infeasible for brute force attacks. This scheme also speeds up the processing and the process treats each shard as a different message. The repetition of any cipher or key is purely random for a shard. The ciphers and keys can change either simultaneously or independently (see Fig.3) [28]. Every change in each cipher or key, is one time slot and these time slots don't have to be of exact size. As a result, we get multiple shards with different lengths, which carry too little information for a successful attack. In the case of brute force, the correct message is only one small part of the whole actual message [28].

The Venona attack [29] is a common attack for the ciphers which change keys. In this case the key to the attack is to focus on the random method of selecting keys and ciphers for encryption. But in polymorphic encryption system, the shards change so often that there is not a sufficient corpus that accumulates for a successful attack. The Tempest attack, a side channel attack, attacks the randomized sequence, timing and power of the hardware [13][30]. These attacks are not successful against shards, because polymorphic encryption processes multiple blocks simultaneously. As a consequence, processing multiple threads $(q_1...q_n)$ results in the observed value $(v)$ (Eq.2). For each value measured for side channel attacks, there are sets of possible combinations of random length data block, keys and ciphers, such that the encryption($E$) of a block with a cipher($c$) and a key($k$) generated the observed value (Eq.3) [28].

$$v = \sum_{i=1}^{n} q_i \tag{2}$$

$$\exists\{block, keys, ciphers\} | E_{c,k}(block) = v \tag{3}$$

To perform a meaningful attack, the attacker needs to know how many blocks are processing simultaneously or how many threads are being used in encryption at that moment. Therefore this system gives natural resistance to these side channel attacks [13].

## IV. Proposed Design

Keeping the advantages of Polymorphic Encryption over block ciphers including AES in mind, we propose to create a practical installation of Lightweight Polymorphic Encryption that resides on a typical small IoT device [10][11][12][13][14][15]. We divide our proposed design in two subsections. The Pseudo Random Number Generator (PRNG) and the Proposed Lightweight Polymorphic Encryption System.

### A. Pseudo Random Number Generator

Randomness is a common, but important concept used in cryptography [31]. The idea is to generate uniformly distributed numbers. Unpredictability, lack of bias, bit independence, non-repeating behavior and long cycle length are necessary qualities of a strong PRNG [32]. Although a true randomness is desirable there is no way to achieve it computationally that can be followed for decryption without sharing the entire sequence. We designed a PRNG to have deterministic random numbers from the decryption side. Before we designed our PRNG, the design requirements we had in our mind were,

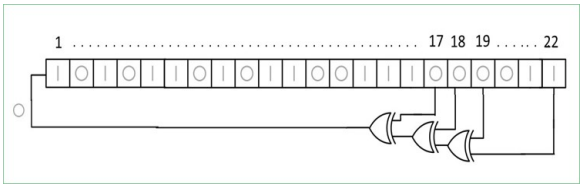


Fig. 4. 22-bit Linear Feedback Shift Register

the characteristics of a strong PRNG, easy to implement in Hardware and lightweight properties. In this work, we have considered a hybrid design consisting of a Linear Feedback Shift Register (LFSR) and Cellular Automata- Rule 30 (CA-30) (see Fig.5). We chose this hybrid design to increase the cycle length, without spending a lot of hardware.

An LFSR is a linear recurrent PRNG. It uses a sequence of Flip-Flops (FFs) as shift registers and generates one bit per iteration. The FFs are connected with their neighbors. In each iteration the binary value shifts once and the last register produces the output. A feedback loop is designed, which is determined by a characteristics polynomial [31]. The feedback is XORed (bitwise binary exclusive OR function) with each other and also with the first bit, this is called tapping. In our design we used a 22-bit LFSR with a feedback polynomial $x^{22} + x^{19} + x^{18} + x^{17} + 1$ [33], which means the taps are at the 22nd, 19th, 18th and 17th bits (see Fig.4). The 16-bits were selected from 22-bits, to have more unbiased outputs.

Cellular Automata (CA) is a discrete model, which was proposed as formal models of self reproducing robots. A basic one dimensional CA PRNG will produce random outputs with an internal state machine, which can be a Boolean function rule [31]. In this design we have incorporated CA Rule 30 with LFSR to increase the cycle length of the PRNG. From Fig.5, the truth table of Rule 30 implies, three consecutive bits at time, $T$ and the new state of the center bit at time, $T+1$. The state of center bits or cells is $(00011110)b$ in binary, which is $(30)d$ in decimal and hence the name [34]. We chose a 30-bit CA-30 and 16-bits were selected from 30 bits, to have more unbiased outputs.

The outputs from LFSR and CA-30 are then XORed to create the final random number generation. The reason we chose this hybrid design is to increase the cycle length without spending a lot of hardware. The hybrid design increased the cycle length dramatically to $2^{52} - 1$. We designed in Verilog register transfer language (RTL) and simulated our design in Modelsim and ran statistical tests on the output. Fig.6 shows the distribution of the generated random numbers. The generator generates very uniformly distributed random numbers. Most of the numbers are close to the average. This proves unpredictability and lack of bias properties of a strong random number generator. $500K$ random numbers from the generator were tested using Dieharder testing suite [35] and passed 10 statistical tests including Birthdays Test, 32x32 Binary Rank Test and Count the 1s Test (See Fig.7). We also synthesized our design with Synopsys Design Vision synthesis suite to achieve an area estimation. The estimated

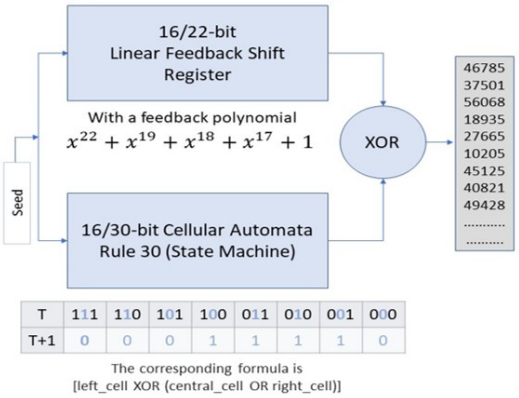


Fig. 5. Proposed Pseudo Random Number Generator: A hybrid between LFSR and CA-30


Fig. 6. Relative frequency of generated random numbers versus the average uniform distribution

```
#=============================================================================#
#            dieharder version 3.31.1 Copyright 2003 Robert G. Brown          #
#=============================================================================#
   rng_name    |           filename             |rands/second|
     file_input|                        lfsrca500|  2.46e+06  |
#=============================================================================#
        test_name   |ntup| tsamples |psamples|  p-value |Assessment
#=============================================================================#
# The file file_input was rewound 27 times
   diehard_birthdays|   0|       100|     100|0.40953054|  PASSED
# The file file_input was rewound 479 times
  diehard_rank_32x32|   0|     40000|     100|0.02932856|  PASSED
# The file file_input was rewound 1472 times
         diehard_dna|   0|   2097152|     100|0.30521095|  PASSED
# The file file_input was rewound 1484 times
diehard_count_1s_str|   0|    256000|     100|0.78157936|  PASSED
# The file file_input was rewound 1738 times
diehard_count_1s_byt|   0|    256000|     100|0.91847066|  PASSED
# The file file_input was rewound 1743 times
 diehard_parking_lot|   0|     12000|     100|0.55881466|  PASSED
# The file file_input was rewound 1746 times
    diehard_2dsphere|   2|      8000|     100|0.08429224|  PASSED
# The file file_input was rewound 1748 times
    diehard_3dsphere|   3|      4000|     100|0.05372960|  PASSED
# The file file_input was rewound 2206 times
        diehard_sums|   0|       100|     100|0.93197056|  PASSED
# The file file_input was rewound 2226 times
       diehard_runs|   0|    100000|     100|0.05291228|  PASSED
       diehard_runs|   0|    100000|     100|0.11232656|  PASSED
```

Fig. 7. Results of 10 statistical tests from Dieharder: Random Number Generator testing suit

area Gate Equivalent of the proposed PRNG is 1129 with 45nm technology (Fig.8).

### *B. Proposed Lightweight Polymorphic Encryption System*

Our proposed lightweight polymorphic encryption system works with 128-bit cipher block and 80-bit input key. We have considered two different blocks as polymorphic algorithms. The first is XOR and permutation and the second is XOR and substitution. While the polymorphic technology gets rid of the redundancy of an encrypted message, by choosing simple

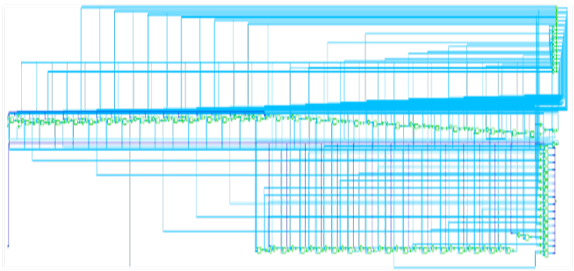

Fig. 8. Hybrid Random Number Generator LFSRCA30: Schematic Diagram synthesized by Synopsys Design Vision synthesis suite

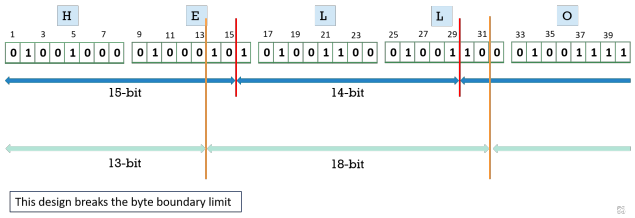


Fig. 9. Example Condition 3: Start sharding with an odd bit

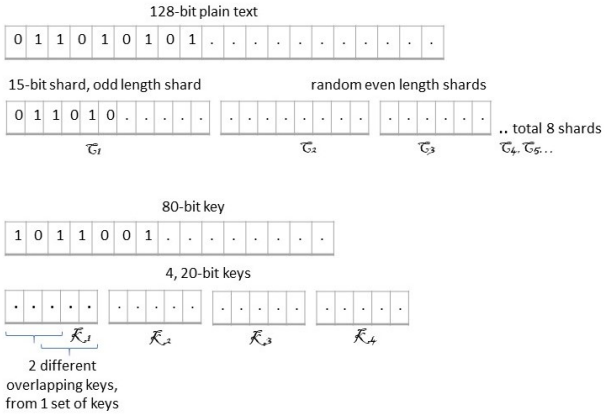


Fig. 10. Proposed Lightweight Polymorphic Encryption System : Sharding and Key Distribution

blocks like permutation and substitution, our design gives us the desired lightweight property. 80-bit key gets divided into 4 parts and each 20-bit gets used for two shards with two different encryption blocks. We get two different overlapping keys from one key, total eight keys (see Fig.10). For sharding there are three conditions applied. As we are designing in a constrained environment, we impose the following constraints for the system.

Condition One: Divide 128-bit block into eight shards: 128-bit input data gets divided into eight random length shards. Once we load the key and divide it into four parts, from those four parts we achieve total eight overlapping keys. The input data gets divided into eight random length shards, so that each shard has a unique key from available eight keys.

Condition Two: Shard lengths have an upper limit boundary of 20-bits, which means less than or equal to 19-bits. As the shard size is less than 20-bits, choosing two keys from 20-bit key, gives us enough flexibility (see Fig.10).

Condition Three: Start sharding with an odd bit (see Fig.9). Strategically, we have started with odd number of bits, as the length of a shard. This gives us an advantage of breaking one single letter and encrypt the message more strongly. As an example, "$TO$" in hexadecimal is "$544f$" and in 16-bit binary it is ""0101010001001111". If we can "shard" at 15th bit, then we can make sure two consecutive letters are not predictable in any way. Once we start with an odd sharding, after that we can continue with random even length sharding (see Fig.10). This way it will always land on an odd bit and the design can break the byte boundary limit, which eventually break the letter and will provide the encryption system more entropy [25]. After experiments we have calculated that there are 128 possible ways of sharding which will fulfill all three conditions. If we didn't have a constrained environment, we could have more number of possible ways of sharding.

As encryption blocks, we have chosen very simple permutation, $P$ and substitution, $S$ blocks (see Fig.11), which results in cost effective area estimation. From Fig.11, we can see that our design doesn't require multiple rounds for a complete encryption. This polymorphic method is achieving more entropy [25] through the random length sharding and keys; ergo, this design achieves good performance.

We designed proposed LPE in Verilog register transfer language (RTL) and simulated our design in Modelsim (Fig.12) and synthesized with Synopsys Design Vision synthesis suite to achieve an area (Fig.13). The estimated area Gate Equivalent of the proposed lightweight polymorphic encryption system is 764. Including the PRNG our complete design's area estimation is 1893 Gate Equivalent, which is less than the 2000 Gate Equivalent threshold [8].

## V. Conclusion and Future Work

In our research, we proposed hardware implementation of a novel lightweight polymorphic encryption system based on CipherLoc polymorphic encryption. For the end devices of the IoT network, which are generally constrained in nature, heavyweight security systems are not desirable. Hence, our goal is to achieve the optimal security solution for these

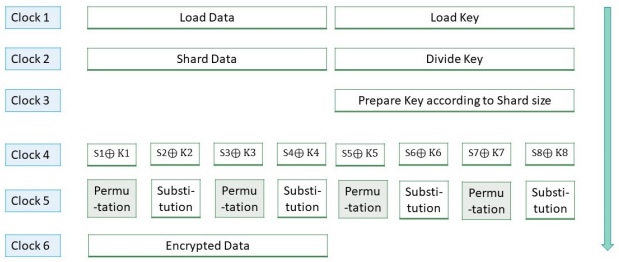


Fig. 11. Proposed Lightweight Polymorphic Encryption System : Encryption Flow

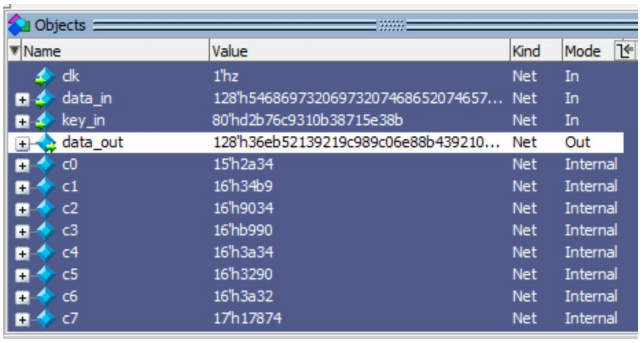


Fig. 12. Modelsim Simulation Output of Proposed Encryption System

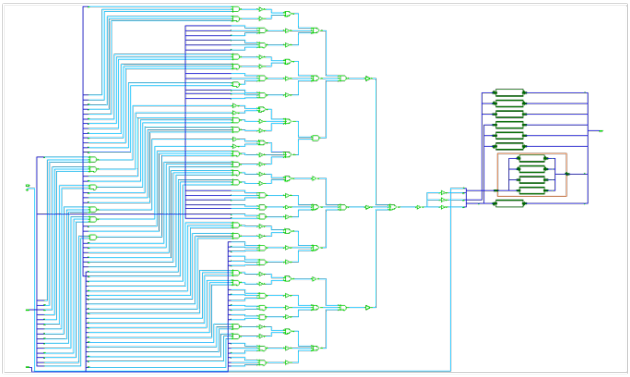

Fig. 13. Proposed Lightweight PE System: Schematic Diagram synthesized by Synopsys Design Vision synthesis suite

constrained devices, so that, the cost of breaking the system is higher than the information retrieved from these devices by the hacker. Hardware implementation of a cryptographic system ensures better performance and stronger security than the equivalent software implementation. Also, polymorphic encryption gives a natural resistance to the most common attacks. We were able to achieve area efficiency by keeping the Gate Equivalent of our lightweight design of the PRNG and the encryption block to 1893, which is considerably lower than the 2000 Gate Equivalent threshold. Our novel design includes a pseudo random number generator and a polymorphic encryption system within the 1893 Gate Equivalent, which gives us stronger security in a lightweight manner.

For our future work, we will perform a more comprehensive security analysis on our design and exhibit how the increased capabilities of IoT devices can protect the important data created and stored on such devices. We are living in a world where data or information is the most valuable asset, yet the most vulnerable. With our work, we aim at making this world of vulnerable data more secure.

## REFERENCES


[1] TechTarget. Cutting edge: It's guide to edge data centers. 2019.
[2] Bill Curtis. Iot device security concerns could limit iot growth. 2019.
[3] Keith Kirkpatrick. Protecting industrial control systems. *Communications of the ACM*, 2019.
[4] CipherLoc Corporation. Technology overview. 2017.
[5] N. Sklavos and O. Koufopavlou. Mobile communications world: Security implementations aspects - a state of the art. *CSJM Journal, Institute of Mathematics and Computer Science*, 2003.
[6] K. TOULIOU N. SKLAVOS and C. EFSTATHIOU. Exploiting cryptographic architectures over hardware vs. software implementations: Advantages and trade-offs. In *Proceedings of the 5th WSEAS International Conference on Applications of Electrical Engineering*, 2006.
[7] Bruce Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley and Sons Inc., New York, 2nd edition, 1996.
[8] Vikash Kumar Jha. Cryptanalysis of lightweight block ciphers. Master's thesis, 2011.
[9] Indira Kalyan Dutta ; Bhaskar Ghosh ; Magdy Bayoumi. Lightweight cryptography for internet of insecure things: A survey. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, 2019.
[10] Bertrand Cambou. A xor data compiler: Combined with physical unclonable function for true random number generation. In *SAI/IEEE computing conference*, 2017.
[11] Bertrand Cambou. Multi-factor authentication using a combined secure pattern, 2015.
[12] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology — CRYPTO '96*, 1996.
[13] Loai Tawalbeh, Hilal Houssain, and Turki F. Al-Somani. Review of side channel attacks and countermeasures on ecc, rsa, and aes cryptosystems. In *Journal of Internet Technology and Secured Transaction*, 2017.
[14] François-Xavier Standaert. Introduction to side-channel attacks. In *Secure Integrated Circuits and Systems (pp.27-42)*, 2010.
[15] Albert Carlson, Patrick Doherty, Isaiah Eichen, and James Gall. Using collisions to break cbc. In *ShowMeCon*, 2016.
[16] Wikipedea. Wikipedia - kevin ashton. 2019.
[17] Statista. Global business data platform. 2019.
[18] IoTSense. The layers of iot. 2018.
[19] Md Sirajuddin Inamdar and Sahadev Roy. Internet of things: Architecture, security and applications. In *International Journal of Advanced Engineering and Management*, 2017.
[20] David Reinsel, John Gantz, and John Rydning. Idc white paper; the digitization of the world. *N/A*, 2018.
[21] Khalid Elgazzar Kasem Khalil and Magdy Bayoumi. A comparative analysis on resource discovery protocols for the internet of things. *IEEE Global Communications Conference (GLOBECOM)*, 2018.
[22] Siroos Madani; Mohammad R. Madani ; Indira Kalyan Dutta ; Yamini Joshi ; Magdy Bayoumi. A hardware obfuscation technique for manufacturing a secure 3d ic. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2018.
[23] David Gloag. Information & computer security training. N/A.
[24] Albert H. Carlson. *Set Theoretic Estimation Applied to the Information Content of Ciphers and Decryption*. PhD thesis, 2012.
[25] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 1949.
[26] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. 1996.
[27] Ueli M. Maurer and James L. Massey. Cascade ciphers: The importance of being first. In *Journal of Cryptology*, 1993.
[28] Albert H. Carlson Christopher Philabaum D. Duane Booher, Bertrand Cambou. Dynamic key generation for polymorphic encryption. In *IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, 2019.
[29] John Earl Haynes and Harvey Klehr. Venona: Decoding soviet espionage in the united states (yale nota bene). *Yale University Press:New Haven*, 1999.
[30] National Security Agency. Tempest: A signal problem – the story of the discovery of various compromising radiations from communications and comsec equipment. *Cryptologic Spectrum*, 2(3):26 – 30, 1972.
[31] Mohammed Bakiri. Hardware implementation of pseudo random number generator based on chaotic iterations, 2018.
[32] Thomas Tkacik. A hardware random number generator, 2002.
[33] Tim Molteno Roy Ward. Table of linear feedback shift registers. 2007.
[34] Stephen Wolfram. *Tables of Cellular Automaton Properties*. 1986.
[35] David Bauer Robert G. Brown, Dirk Eddelbuettel. Dieharder: A random number test suite. In Duke University Physics Department, editor, *Duke University Physics Department*, 2019.