Essential Requirements of IoT's Cryptographic Algorithms: Case Study

Shubham Kumar, Zubair Ahmad Lone, and B. R. Chandavarkar

Abstract Internet of Things (IoT) devices are increasing rapidly in today's world, but the security of devices remains a major concern due to the unavailability of the memory and processing power in these devices, which is because of their smaller size. The trade-off lies between security and performance, i.e. if security is increased, which will come with high complexity and hence would deter the performance. On the other hand, if performance has to be increased, it would come with a cost in terms of security. Also, IoT devices can be used as bots as they are globally accessible without much of a security. The most secure cryptographic algorithms use a lot of resources, and in case of IoT, resources are not available on that scale, so there is a need to design a secure algorithm (lightweight cryptography) that would use less resources and hence won't affect the performance either.

Keywords IoT · Lightweight cryptography · Cryptography · KATAN · BEAN · AEL · DESL · GRAIN · Trivium · Quavium

1 Introduction

Cryptography is an art of hiding the information in data in such a way that only the intended recipient will be able to retrieve the information out of the data sent by the receiver. The information is retrieved using the key that the sender and receiver have agreed before the transferring of data. The information is converted to ciphertext using the key(encryption). The data after received by the receiver is again

S. Kumar (✉) · Z. A. Lone · B. R. Chandavarkar
Department of Computer Science and Engineering, National Institute of Technology Surathkal, Mangalore, India
e-mail: sshubhamk1@hotmail.com

Z. A. Lone
e-mail: zubair.197cs003@nitk.edu.in

B. R. Chandavarkar
e-mail: brcnitk@gmail.com

A. Kumar and S. Mozar (eds.), *ICCCE 2020*, Lecture Notes in Electrical Engineering 698, https://doi.org/10.1007/978-981-15-7961-5_16

converted plain text(decryption) using the key(decryption). The key may be of one of two types:

- Shared Key: Sender and Receiver both uses the same key for encryption as well as decryption.
- Public Key: Sender uses different key (public key) for encryption while the receiver uses different key (private key) for decryption.

These algorithms are implemented with such complex techniques that obtaining the plain text without the key is nearly impossible for the attacker. For this to happen, these algorithms use a lot of computation [1]. These algorithms are working fine until it was used for large systems with large memory and processing power. But nowadays people are moving toward mobile devices because of their small size and easy to carry. Also, smart products are increasing day by day. These devices also need some cryptographic algorithms because the receivers are at the end of these smart devices, and hence information should be available to them. For this problem to tackle, lightweight Cryptography are introduced to provide the security at the level of previously well established standard algorithms like AES, DES but with very less memory consumption and very less computations.

This paper focuses on various requirements needed to build cryptographic algorithms for these IoT devices. This paper is organised as follows: In Sect. 2, Use of Cryptography is shown in the field of IoT, In Sect. 3 lightweight Cryptography is introduced, In Sect. 4 we have shown different types of symmetric lightweight algorithms along with some examples.

2 Cryptography in IoT Devices

IoT requires an uninterrupted network inter-connectivity as well as cloud platform to manage data sharing and storage. However, the IoT, with real-time applications, includes massive data processing and transformation. Nevertheless, ICs deployed in IoT based infrastructures have strong constraints in terms of size, cost, power consumption and security [2]. According to some latest estimations, more than 18 billion IoT devices will be connected via cloud platform by 2020 and amongst those around 57% will be IoT's applications, but the guarantees of confidentiality and data protection are not entirely up to the mark yet. One of the reasons being that generally, we don't have much of the computational capacities as the devices are usually small like smartwatches, RFID tags, mobile apps, etc. [2]. The biggest challenge facing the IoT technology developers is to develop the algorithms that would use less computation, less memory and be able to secure the system as the conventional cryptographic algorithms do. The concept of LWC (Light Weight Cryptography) is a step toward that goal [1]. LWC is in its emerging phase. Nevertheless, the need of the efficient algorithms is an urgent requirement in IoT, and the measure of efficiency would include ultra-high-speed transmission, very

low latency, affordability, open-source capabilities, green networking with minimal power consumption and prevention of possible threats or attacks. so every cryptographic technique used here should consume less memory, less computing storage and less battery usage and it should deliver efficient security and confidentiality in spite of all the constraints listed [2].

There is often a trade-off between the methods used for cryptography and the overall security. More often than not LWC methods have to balance performance/throughput with the power drain and GE (Gate Equivalents).

[3] and hence, cannot perform as well as mainstream cryptography standards such as AES and SHA-256. Along with that, the method also has a low requirement for both types of memory, i.e., main memory, where the technique requires the usage of running memory to perform its operation and Secondary memory, where the program is stored on the device [4].

3 Lightweight Cryptography

To overcome various issues that are related with standard cryptographic algorithms, several new algorithms were introduced, but the trade-off between performance and security led the researchers to come up with this new kind of cryptography - lightweight Cryptography which is exclusively used for low-end devices. The goal is to provide all the functionality provided by classical cryptographic algorithms but with less computation, size and time taken as compared to classical cryptographic algorithms. i.e., It'll provide all the necessary security services like authenticity, confidentiality and integrity with less execution time and less memory utilisation.

Lightweight Cryptography is defined as the cryptography used for resource-constrained devices. As the name suggests, its feature is lightweight, which means it can be easily implemented on a small platform, software as well as hardware [5]. It is important in the field of IoTs because classical cryptographic algorithms are often slow, big or too much energy-consuming of these devices. Various lightweight cryptography created are broadly categorised in terms of key involved, i.e., Symmetric and Asymmetric. Asymmetric is used for key exchange and other similar function while symmetric cryptography is used for data transmission because of better performance in terms of time.

4 Symmetric Lightweight Algorithms

Symmetric is further subdivided into two types on the basis of bits of data received by the algorithm for transformation. They are following

- Stream Cipher: In these kinds of algorithms data are sent as a stream of bits and encryption, and decryption is performed in the same fashion, and hence data can be sent to receiver with very less latency
- Block Cipher: In these kind of algorithms, a fixed size of data is collected first and are transformed by the algorithm before transmission and hence takes much more latency as compared with stream cipher. On the other hand, it provides more security as compared with the stream cipher algorithms [4].

4.1 Symmetric Lightweight Stream Ciphers

These ciphers encrypt/decrypt data as it is coming in from of stream of bit and hence taking the plain text and providing the ciphertext continuously like a pipeline after one pass. Some of stream cipher for lightweight cryptography are following

- GRAIN: GRAIN cipher provides comparatively high security despite using minimal hardware, fewer gates. The cipher is specially designed to be implemented in IoT devices, where the resources like gate count, power consumption and memory are constrained. The cipher outputs one bit/clock. One of the advantages of the GRAIN cipher is that its efficiency can be increased by increasing the hardware. Some of the attacks that have happened over time with the GRAIN ciphers like shortcut key-recovery attack, Related key attacks and the attack that uses conditional differential cryptanalysis as a way to recover two key bits [6, 4].
- BEAN: This cipher is based on the basis of GRAIN stream cipher and hence is more optimised than GRAIN cipher. It uses S-Box along with two FCSRs. One major advantage over GRAIN is that it is software-based, i.e., does not require any additional hardware support for its implementation. It uses nearly the same memory as used by GRAIN but produces in significantly lesser time than GRAIN. The disadvantage of this cipher is that it can be attacked using distinguish-er attack and state-recovery attack, which is possible due to its weak output [4].
- Trivium: A hardware-oriented stream cipher, capable of providing a tradeoff between speed and area, takes less power without much of a difference in operating frequency. It reduces power consumption by about 20%. The most cryptanalytic results on Trivium are obtained by cube attacks and cube testers. The efficiency of the fault injection system ranges from 68% to 100% for the standard version of Trivium. Trivium implementations on FPGA are vulnerable to fault injection attack, irrespective of the implementation of the device used for the implementation of [4].
- Quavium: As name suggests, it is the successor of Trivium cipher providing scalable solution with the same key size as used before (80 bits) and same internal state (288 bits). It uses a 4-round Trivium-like LFSR. Even after increasing the complexity of this cipher, it still generates the random key nearly

as fast as Trivium does. Further, it is optimised for by decreasing the number of round from 4 to 3. it uses less number of logic gates in comparison with Trivium [7, 4, 8].

4.2 Symmetric Lightweight Block Ciphers

Unlike stream cipher, block cipher uses a block of bits and performs some computation on it before generating its output. Due to this process, it requires more time for execution in comparison with stream ciphers. The essential requirement while implementing a block cipher is the designing the Substitution-Permutation Network [9] and Feistel Network [10]. Creation of Substitution-Permutation box in the constrained device is not appreciated so much due to the limit on the memory of the device. This is the reason for not using S-box or use of small S-box in lightweight block ciphers.

- KATAN: It is the smallest known block cipher formed with less than 500 GE. More optimised version KATANTAN is more compact in hardware due to static key used and is programmed in the device and hence cannot be changed. The problem with this cipher is its speed. It gives output at 254 clock per cycle. Still, it is scalable as it can be made three times faster with an increase of negligible area [11].
- AES: AES is the standard cipher suited for software and hardware implementation with versions of 128, 192 and 256 keys. It works very well in larger systems, but not for the constrained device. So ALE (AES-Based lightweight Authenticated Encryption) [12] is introduced, which is efficient for both hardware as well as software implementation. It requires around 2500 GE, which is less than 100 GE overhead compared to plain AES-ECB in the smallest implementation available.
- DESL: Like ALE, researchers also optimised DES for lightweight devices which is strong, efficient and compact. Due to its low memory space-constrained, it is heavily used in RFID. The S-box of DES is highly optimised here, keeping in mind about the attack that may occur due to weakening the S-box. DESL [3] requires 45% less chip size and 86% less clock cycle than the standard AES algorithm with regard to RFID implementation.

5 Essential Requirements in Implementation of Cryptographic Algorithms

IoT device constraint makes several Standard cryptographic algorithms unfavourable to use. Here are some essential requirements for good cryptographic algorithms for IoT devices:

- LFSR: It requires fewer gates and produces output at higher frequency rate, and hence is lighter as well as faster for lightweight devices replacing the counter used earlier for pseudo-random number generator.
- FCSR (Feedback with Carry Shift Register): It is basically LFSR with extra memory to retain carry from one stage to another and hence providing more random sequence [13].
- The avalanche effect [14]: It states that the output should change with more than 50% on change of single bit of the input. It is mainly satisfied by algorithms like AES and DES, but in the case of IOT's cryptographic algorithms, it limits down to 28% or less.
- Substitution-Permutation Network [9]: The lesser the size of S box, the better for IoT in terms of memory but it should be large enough to tackle the possible attacks on this network.
- Feistel Network [10]: It is crucial while making a block cipher. Again the size of the Feistel network is chosen keeping the security and space provided by IoT both in mind.

6 Conclusion and Future Work

In conclusion, we say that IoT devices require not only good lightweight algorithms but also proper design for these algorithms to work fine. As the growth of IoT is increasing exponentially, we need more robust and secure algorithms that would use less resources like power, memory etc. to keep IoTs secure. IoT's are also used in smart grids now, any attack on these systems could lead to a catastrophe. Such feasible measures like those mentioned above are to be taken to minimise the unwanted effects as much as possible. The given solution is useful but does not guarantee the overall securities for these devices. We can achieve avalanche effect from 60% to 70% by using algorithms like AES, Camellia, DES, MMB, but they use more computational power, and in future, we would like to concentrate on decreasing the power consumption to make them more affordable for IoT.

References

1. Shukla A, Tripathi S (2018) A Survey on Next generation Computing IoT Issues andChallenges. Int J Pure Appl Math 118(9):45–64
2. Sallam S, Beheshti BD (2018) A survey on lightweight cryptographical gorithms. In: TENCON 2018-2018 IEEE region 10 conference. IEEE, pp 1784–1789
3. Rolfes C, Poschmann A, Leander G, Paar C (2008) Ultra-light weight implementations for smart devices–security for 1000 gate equivalents. In: International conference on smart card research and advanced applications. Springer, Heidelberg, pp 89–103
4. Gunathilake NA, Buchanan WJ, Asif R (2019) Next-generation light weight cryptography for smart IoT devices: implementation, challenges and applications. In: 2019 IEEE 5th world forum on internet of things (WF-IoT). IEEE, pp 707–710
5. Sadkhan SB, Salman AO (2018) A survey on lightweight-cryptography status and future challenges. In: 2018 international conference on advance of sustainable engineering and its application (ICASEA). IEEE, pp 105–108
6. Hell M, Johansson T, Meier W (2007) Grain: a stream cipher for constrained environments. IJWMC 2(1):86–93
7. Hosseinzadeh J, Hosseinzadeh M (2016) A comprehensive survey on evaluation of lightweight symmetric ciphers: hardware and software implementation. Adv Comput Sci Int J 5(4):31–41
8. Tian Y, Chen G, Li J (2012) Quavium-a new stream cipher inspired by trivium. JCP 7 (5):1278–1283
9. Belazi A, El-Latif AAA, Belghith S (2016) A novel image encryption scheme based on substitution-permutation network and chaos. Sig Process 128:155–170
10. Nyberg K (1996) Generalized feistel networks. In: International conference on the theory and application of cryptology and information security. Springer, Heidelberg, pp 91–104
11. De Canniere C, Dunkelman O, Knežević M (2009) KATAN and KTANTAN—a family of small and efficient hardware-oriented block ciphers. In: International workshop on cryptographic hardware and embedded systems. Springer, Heidelberg, pp 272–288
12. Bogdanov A, Mendel F, Regazzoni F, Rijmen V, Tischhauser E (2013) ALE: AES-based lightweight authenticated encryption. In: International workshop on fast software encryption. Springer, Heidelberg, pp 447–466
13. Klapper A (2004) A survey of feedback with carry shift registers. In: International conference on sequences and their applications. Springer, Heidelberg, pp 56–71
14. Ramanujam S, Karuppiah M (2011) Designing an algorithm with high avalanche effect. IJCSNS Int J Comput Sci Netw Secur 11(1):106–111