Lightweight Cryptography Delivers Confidentiality and Integrity to Resource Constrained Devices and Maintains Performance

Jeffery Malavasi
Department of Computing Security
Rochester Institute of Technology
jm3378@rit.edu

***Abstract* —** Throughout the past decade we have seen an explosion of internet-connected devices with numbers estimated to be in the billions. Often these devices lack the necessary resources (physical, hardware, and software) to implement secure cryptographic functions. As a result, an enormous amount of data is transferred between computers and even third parties in plain text each day. This data is often aggregated to improve marketing tactics that directly target consumers. Lightweight cryptography is a novel suite of ciphers that aim to provide hardware constrained devices a high level of security, while maintaining a low physical cost and high performance. In this project, we are going to compare some of the recently proposed ciphers (Grain-128AEAD, GIFT-COFB, TinyJAMBU and Elephant) to existing modern algorithms (ChaChaPoly) as well as explore potential limitations to lightweight cryptography. Finally, results were validated using a performative analysis on an Arduino Due. Algorithms TinyJAMBU and GIFT-COFB performed better than ChaChaPoly while maintaining confidentiality and integrity.

***Keywords* — Internet of Things, Lightweight Cryptography, Security Goals**

I. INTRODUCTION

The Internet of Things (IoT) consists of a wide variety of devices that often lacks a conventional definition. In fact, according to NIST, "There is no formal, analytic, or even descriptive set of the building blocks that govern the operation, trustworthiness, and lifecycle of IoT" [1]. In 2020 there were over fifty billion IoT devices online, and that is expected to continue to grow at an exponential rate (Figure 1).

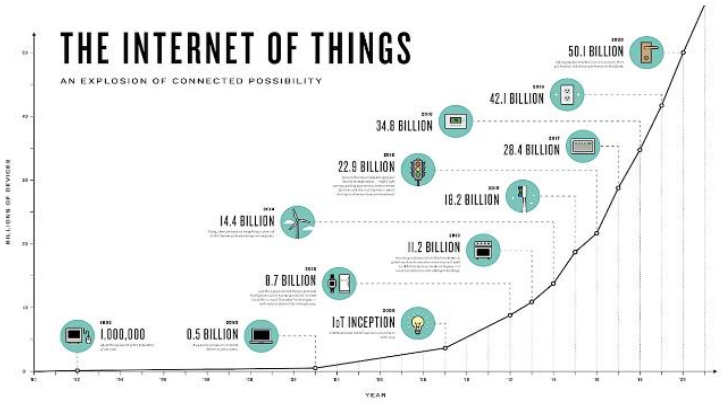

Figure 1: Emergence of IoT Devices [2]

Generally, IoT devices are tiny microsystems that collect, store, and transmit data about their local environment to a cloud network of powerful computers that can be used to aggregate and analyze the information. IoT Devices span across various domains including manufacturing, agriculture, healthcare, and consumer smart home devices [3]. Security becomes a challenge in these networks because endpoints are often resource constrained, and accessible on the public internet [4]. According to a recent report by ZScaler, over 90% of IoT traffic is sent over unencrypted channels [5]. This provides bad actors with a large attack surface to intercept and potentially modify IoT traffic before it reaches its destination. IoT devices are especially vulnerable to confidentiality-based attacks since they are often deployed in public spaces where physical access to endpoints is easily achievable [6].

A. Security Goals

Figure 2: CIA Security Triad [7]

It is quite clear that due to the exponential growth in the Network of Things ecosystem, researchers will need to quickly develop robust security controls. Additionally, bad actors have already taken advantage of these generally insecure systems. One recent example found that an adversary could remotely control an on the market insulin pump and even deliver a lethal dose [8]. When defining a security landscape, security architects rely on the CIA Triad [8]. The CIA Triad is a well-known acronym that stands for confidentiality, integrity, and availability. In this paper, we will outline how lightweight cryptography achieves the goals of confidentiality and integrity.

Confidentiality

Confidentiality aims to ensure that the correct data is always presented to the correct user. Systems that have a high level of confidentiality prevent unauthorized access to critical data. In IoT, both consumer and enterprise systems lack regulatory requirements for data confidentiality and therefore according to a recent survey around 50% of devices lack a pin or password and 70% transmit data via unencrypted channels [8]. This is especially concerning because information that is transmitted via these

types of devices is often confidential in nature, including biometrics, health records, personal habits, and even military data.

Integrity

Integrity, is responsible for detecting malicious or erroneous changes to the data stream as well as correcting changes on the fly [8]. Note that in contrast to confidentiality, it is expected that data will be manipulated and therefore proper system integrity requires compensating mechanisms. This is particularly important in IoT, as more critical infrastructure is integrating resource constrained devices into day-to-day business operations. For example, in healthcare IoT, resource constrained devices are being used on ambulances to help emergency medical technicians and paramedics monitor patient health [9]. If this information is tampered with, it could ultimately lead to the wrong treatment.

Availability

Availability specifically speaks to the need for data to be always accessible to the user when needed. This becomes more important as the adoption of the internet of things expands. For example, smart locks are used to secure home and small businesses. If these devices went offline, consumers would not be able to get into their homes or places of work. In this project, availability goals will not be discussed, as encryption does not provide a direct solution to this problem.

B. Cryptography

Cryptography has been used for thousands of years to provide confidentiality and sometimes integrity while exchanging messages between two parties. Over time as the strength of encryption improved, it often became more resource intensive creating a large gap in secure devices online. Cryptographic algorithms can be categorized as either symmetric or asymmetric. In symmetric key cryptography the same key is used to encrypt and decrypt the message creating immediate limitations: keys must be unique and exchanged over a secure channel [10]. Once exchanged however, data can be processed fast and efficiently. Additionally, keys are often kept small, which leaves them vulnerable to attack. Symmetric algorithms can be broken down into stream, where data is processed one bit at a time, or block, where data is processed in chunks. In asymmetric cryptography, different keys are used for encryption and decryption. This often requires larger keys and higher power mathematics, slowing down processing time [10].

Block Ciphers

Block ciphers are often implemented because they can process large data streams by breaking them into smaller blocks and increasing the unit of transformation [10]. Each block is processed one at a time and is combined using cipher modes of operation. There are a few different ways block ciphers can process multiple blocks at a time, but one of the most common is cipher block chaining (CBC). It is especially unique because it can provide both confidentiality and integrity to the data stream. This happens through the addition of a message authentication code (MAC), or simply a message fingerprint. The MAC allows the receiver to verify the contents of the message to ensure that they weren’t modified during transmission [10].

One of the most widely used modern cryptographic ciphers is the Advanced Encryption Standard, or AES. It was ultimately selected as the replacement for DES in 2001, after an NSA sponsored encryption competition [11]. It works using a complicated and multi-step substitution permutation network. AES allows for variable key lengths to balance performance and security. As the key length increases, so does the number of rounds in encryption and decryption. Each round consists of four functions: SubBytes, ShiftRow, MixColumn, and AddRoundKey [11]. Due to the complexity in round functions, AES can be quite resource intensive and often slow on IoT devices. This is often why most traffic in the network of things landscape is often transmitted unencrypted.

Stream Ciphers

Stream ciphers are often implemented because they can quickly encrypt and decrypt an endless data stream. This provides a high level of confidentiality, without sacrificing performance [10]. Additionally, since the decryption of any single bit is not reliant on the previous bits, any portion of the data can be accessed efficiently. Stream ciphers rely on the creation of a secure pseudo-random number generator that is used as a session key. The level of confidentiality is directly correlated to the entropy in the random number generator [10].

ChaCha, originally proposed by researcher Daniel J. Bernstein in 2008, was designed to be a faster and more lightweight version of AES [12]. It is most related to the Salsa20 family and mainly differs from AES by being a stream cipher instead of a block cipher. This helps to improve its performance and efficiency. ChaCha takes in a 256-bit stream and arranges it into a 4x4 block of 32 bits [12]. A round function is used to randomly mix the block using a combination of just three primitive functions: add, rotate and XOR. This process is repeated for a total of 20 rounds, however a relatively high security margin has been achieved with as little as 8 [12]. This keystream is then XORed with the plaintext to generate the ciphertext. To ensure that the output is different with the same data stream, a nonce is used [12]. Finally, a block number is used to quickly access a specific point in the data stream, instead of needing to rely on previously encrypted bits.

C. Lightweight Cryptography

Lightweight cryptography is specifically designed to bridge the gap between modern cryptography and devices that lack the resources necessary to be able to properly secure data transmission. Largely, these functions attempt to achieve three main goals: operate using minimal cost (physical area, memory utilization, power consumption), perform with low latency and high speed, and provide high levels of security [4]. It is important to note that lightweight does not mean a sacrifice in encryption strength, but instead refers to the use of small block sizes ($\leq$ 64 bits), small key lengths ($\leq$ 80 bit), and simple round functions [4]. In February of 2019, the National Institute of Standards and Technology called for public submissions for lightweight cryptography candidates and has recently narrowed the selection down from the original thirty-six submissions to just ten finalists [13]. In this paper we will examine how four of those finalists: Grain-128AEAD, GIFT-COFB, TinyJAMBU, and Elephant provide confidentiality and integrity while maintaining sufficient performance.

The remaining paper is structured as follows: Section II outlines the adoption of lightweight cryptography, as well as provides a comparative analysis of existing encryption techniques in the IoT space. In Section III, techniques for providing confidentiality and integrity in lightweight cryptography are discussed, and in Section IV, a performative analysis is done on the following ciphers: Grain-128AEAD, GIFT-COFB, TinyJAMBU and Elephant. Finally, Section V provides a conclusion as well as identifies the future work needed to advance the field of lightweight cryptography.

II. RELATED WORK

A. Challenges of Lightweight Cryptography

Researchers S. Mansour *et al.* [6] identify both the rapid increase in IoT devices, as well as a lack of a secure authentication mechanism [6]. Through the combination of lightweight cryptography and hardware-based security approaches, the paper proposes a secure and efficient authentication architecture for resource constrained devices. Researchers were able to improve efficiency by reducing the number of messages being exchanged. Additionally, they eliminated key storage as well the need for a third party certificate authority in order to improve the security of the system [6].

In 2019, Thapliyal, Sourav, *et. Al.* [14] identified the lack of a stable algorithm for resource constrained devices and provided a model that could be used to simplify existing algorithms to increase confidentiality in smart home and IOT devices. The authors utilized a hybrid design that harnessed speed from symmetric cryptography and secure key exchange from asymmetric crypto [14]. The authors identified three parameters to measure the quality of their designs: cost, safety and productivity [14]. Often researchers need to select between these measures, but in this paper the authors attempted to provide a system that balanced all three. The paper also outlines the various challenges that IoT devices face. Firstly, these devices often lack stable power sources and operate on low voltages. Devices can be located in areas that can't be easily accessed such as inside the human body [14]. Secondly, these devices often lack any interface or any means to integrate between systems. Examples of this include smart light bulbs and appliances. Finally, they lack security and privacy as they are unable to perform modern cryptographic operations [14].

Later that year, researchers Gunathilake, Nilupulee *et al.* [15] identified some of the challenges associated with implementing lightweight crypto while predicting an even larger expansion in IoT due to the emergence of 5G [15]. This is due to its ability to provide high speed, low latency data processing while lowering costs [15]. Most notably, a lack of sufficient research into the design of lightweight cryptographic systems. Additionally, since IoT devices require both hardware and software protection, it is difficult to create a strong cipher that addresses both needs efficiently.

B. Applications of Lightweight Cryptography

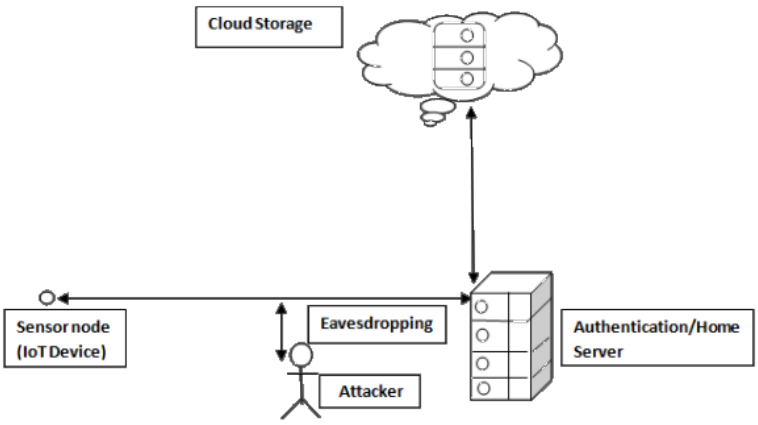

Figure 3: Common IoT Architecture [16]

Researchers Z. A. Pindar, J. O. Fayomi, N. H. Waziri, *et al.* [17] present a novel lightweight cryptographic design that provides both data confidentiality and integrity. In this paper, a variation of MAC uses a Permutation-Quasigroup string transformation in order to provide error detection [17]. When measuring performance of their proposed system on a Raspberry Pi, they found that even messages as large as 128 bits used less than 5% of the processor resources and messages could be hashed in just a few seconds [17]. Additionally, the system had an accuracy of greater than 95% across all error types (single error, single transposition error, and jump transposition errors) [17].

In another paper, researchers identify that IoT traffic, especially at home, is typically sent via unencrypted channels (Figure 3) [16]. This presents an immediate threat to both confidentiality and integrity, as an attacker can simply intercept traffic transmitted between the sensor and the server. This would ultimately lead to modification of data that is uploaded to the cloud service [16]. To mitigate this vulnerability, the researchers proposed a two-fold solution that provided confidentiality, as well as preserved data integrity. The first step encrypted the data using a lightweight cryptographic scheme. Next, the message was hashed and both the ciphertext and the message digest were embedded into an image using steganography [16]. The server was then able to use reverse steganography to obtain and decrypt the original message, which could be verified before uploading to the cloud [16].

Another application of lightweight cryptography is to incorporate it into existing IoT communication protocols. Message Queuing Telemetry Transport (MQTT) is a common IoT protocol that is commonly used in smart home as well as other industries [18]. An MQTT client publishes information to a broker, where other devices can subscribe to for updates. Unfortunately, as the researchers noted, the protocol itself lacks any security controls and often these devices cannot simply layer TLS on top of the protocol [18]. In order to resolve this, the researchers proposed a system that would first encrypt the payload using a lightweight cipher and then hash the message using a lightweight function [18]. The message digest is then published to the broker instead of the plain text message ensuring data confidentiality and integrity [18].

C. Performance of Lightweight Cryptography

Researchers Saldamli, Ertaul, and Shankaralingappa strengthen the previous work done on preserving data integrity through lightweight Message Authentication Code [19]. In this paper, a performative analysis is done on various lightweight algorithms based on power and memory consumption, as well as execution time. When compared to a modern algorithm, researchers found that the lightweight protocols reduced execution time by a factor of 100 and power consumption by a factor of 1000 [19].

In another work, researchers S. Kim *et al.* [20] aimed to provide a performance reference for designing and testing lightweight cryptographic algorithms. Through the creation of a test environment, Kim and Kim measured the execution time of the Elliptic Curve Digital Signature Algorithm's sign and verify steps [20]. After testing multiple IoT devices, they found that software-based optimizations were far more important to performance than environmental changes such as hardware. This shows that lightweight cryptography can be easily adapted to the diversity in the ecosystem [20].

Engineer and Shah further test the performance of lightweight cryptography on resource constrained devices [21]. In their experiment, they use a voice recognition application running in a simulation, to measure speed and memory consumption. Additionally, the further tested their results in a real world model, using an Arduino Uno [21]. They found a significant reduction in speed, when using

lightweight cryptography versus AES [21]. In simulation, there was also a signification reduction in memory usage, however this was not replicated on the physical system [21].

Finally, researchers Sarker, Gia, Tenhunen, *et al*. test some of the most common lightweight cryptographic algorithms on various resource constrained devices [22]. In addition to some of the previously measured parameters, this paper also analyzes power consumption in order to address the IoT challenge of limited power/battery life [22]. Researchers found that memory usage was extremely dependent on algorithm design, especially with key size and the number of processing rounds [22]. This highlights the need for more selectivity in algorithm choice, as compared to the more legacy cryptographic standards (AES/TLS/SSL) [22].

III. OUR APPROACH

A. Lightweight Cryptography Architecture

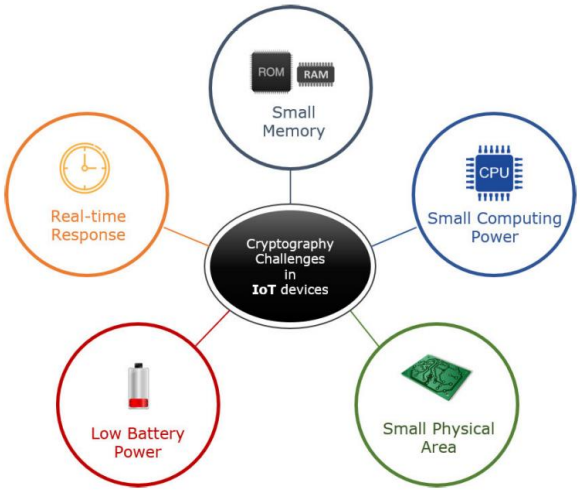

Figure 4: Lightweight Cryptography Challenges [4]

A well-designed lightweight cryptographic algorithm must be able to operate well under challenging conditions (Figure 4). IoT devices are often resource constrained, but still require confidentiality and integrity because they transmit sensitive data. Additionally, these systems rely on responsiveness, requiring the ability for real-time data transmission from IoT sensors [4]. Due to this, performance impact becomes especially important when designing a lightweight algorithm.

In general, block ciphers tend to be more common than stream with lightweight cryptography [4]. This is due to two main reasons. Firstly, block ciphers can efficiently provide both confusion and diffusion, whereas stream ciphers tend to only provide confusion. Secondly, block ciphers are harder to reverse because their round functions offer more operations than just XOR, commonly seen in stream ciphers.

The majority of lightweight cryptographic algorithms are based on the Authenticated Encryption with Additional Data (AEAD) primitive [13]. This is because it can provide both confidentiality and integrity in the same data stream. AEAD algorithms take in a key, plain text, nonce, as well as an optional parameter (additional authenticated data). If the additional authenticated data parameter is not null, it will also be required for the decryption operation. The nonce is used to prevent replay attacks, ensuring that the same

plaintext will not encrypt to the same ciphertext. AEAD designs output the ciphertext, as well as an authentication signature (MAC). The MAC is used to verify the message integrity, ensuring that the data was not tampered with or corrupted during transmission [23].

To maintain performance, cryptographers have a few ways to reduce the strain on lightweight devices. Firstly, by reducing the block size ciphers will consume less memory. This is especially important as IoT devices often only have kilobytes of RAM. However, this needs to be carefully considered, since the reduction of block size also decreases the security margin. This is because a smaller block size will have less overall permutations (2^{32} vs 2^{64}) [23]. Secondly, researchers have also learned that by reducing the key size, efficiency can be greatly improved as encryption often relies on exponential or logarithmic operations [23]. Thirdly, round functions are often simplified to reduce complexity. One way this is accomplished is through the reduction in bit size of substitution boxes. While simpler rounds can improve efficiency, they often must be balanced with more iterations of rounds to maintain security. Finally, in a similar fashion, key scheduling algorithms are simplified in order to reduce memory and power consumption [23].

B. Lightweight Stream Ciphers

Grain-128AEAD was chosen as an example of a lightweight stream cipher. The Grain family was originally introduced in 2008, and this cipher uses an AEAD scheme in order to provide confidentiality and integrity [13]. It has a 128 bit, 96 bit, and 64 bit key, nonce and tag respectively. Researchers Hell *et al.* [24] closely based the cipher on Grain-128a, with the intent to make minimal changes as the security of Grain-128a was already well-respected.

C. Lightweight Block Ciphers

GIFT-COFB, designed by Banik *et al.* [25],[26], is a block cipher with a 128 bit key, nonce and tag. It also provides confidentiality and integrity through the use of an Authenticated Encryption with Additional Data scheme [13]. GIFT-COFB uses 40-rounds and is designed using a substitution permutation network.

TinyJAMBU is another lightweight block cipher that was designed by Wu *et al.* [27]. It has a variable key length of 128, 192, or 256 bits and uses a 96/64 bit nonce and tag. TinyJAMBU is a permutation based cipher that utilizes shift registers [27]. However, TinyJAMBU currently lacks an appropriate security margin, only 12% [13], so it will need to be modified before it can be further adopted. Researchers have theorized that through the introduction of more rounds, the algorithm will be sufficiently secure.

Elephant, proposed by Beyne *et al.* [28] is made up of three variants: Dumbo, Jumbo and Delirium. Each variant uses the same key and nonce size (128 bits and 96 bits respectively), but vary in the size of the MAC, permutation type and the number of rounds [13]. Elephant is a much heavier set of ciphers and uses an encrypt-then-MAC method to provide integrity.

D. Lightweight Cryptography Performative Analysis

Each of the above ciphers will be compared to a modern cipher (ChaChaPoly) to validate their performance on resource constrained hardware. Using an Arduino Due, algorithms will be implemented

to specification using a repository created by researcher Rhys Weatherley in C [29]. Each algorithm will encrypt and decrypt a series of 128 byte and 16 byte packets 1000 and 3000 times respectively. The encryption speed (bytes/second) as well as a relative speed factor (ratio to modern cipher) will be reported.

Table 1: Specifications of Test Hardware

Model	Arduino Due
Microcontroller	AT91SAM3X8E
CPU Architecture	32 Bit ARM
Clock Speed	84 MHz
Operating Voltage	3.3V
Flash memory	512 KB

IV. VALIDATION OF APPROACH

The aim of the experiment was to determine if there were any performative benefits to the proposed lightweight cryptographic ciphers that were recently announced by NIST. When comparing block and stream ciphers to the modern ChaChaPoly block cipher, there was a generally a significant increase in speed with block ciphers over stream (Figure 5). This was expected as the literature review also reflected greater success in block ciphers. Additionally, Figure 5 helps to illustrate that the additional overhead required to decrypt block ciphers is much greater than encryption. In Figure 6, each operation was averaged together to determine a relative speed for every algorithm. From this, we can see that TinyJAMBU (up to 192 bits) and GIFT-COFB were between 15% and 20% faster than ChaChaPoly. As expected the slowest algorithms were Grain-128AEAD and Elephant, as they have a significant overhead compared to other lightweight ciphers.

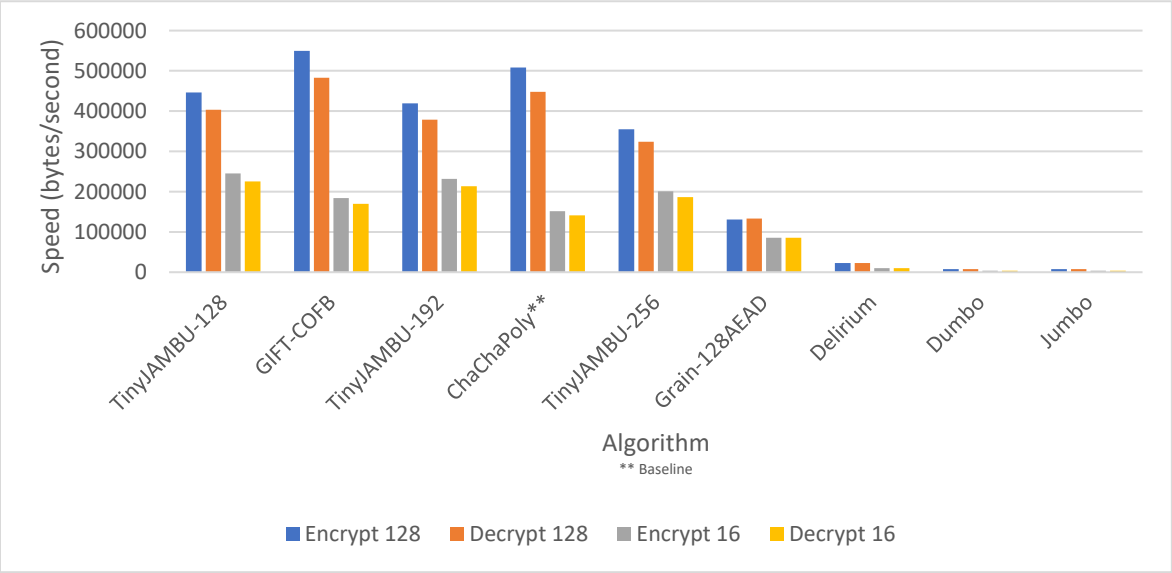

Figure 5: Throughput of Lightweight Cryptographic Algorithms.

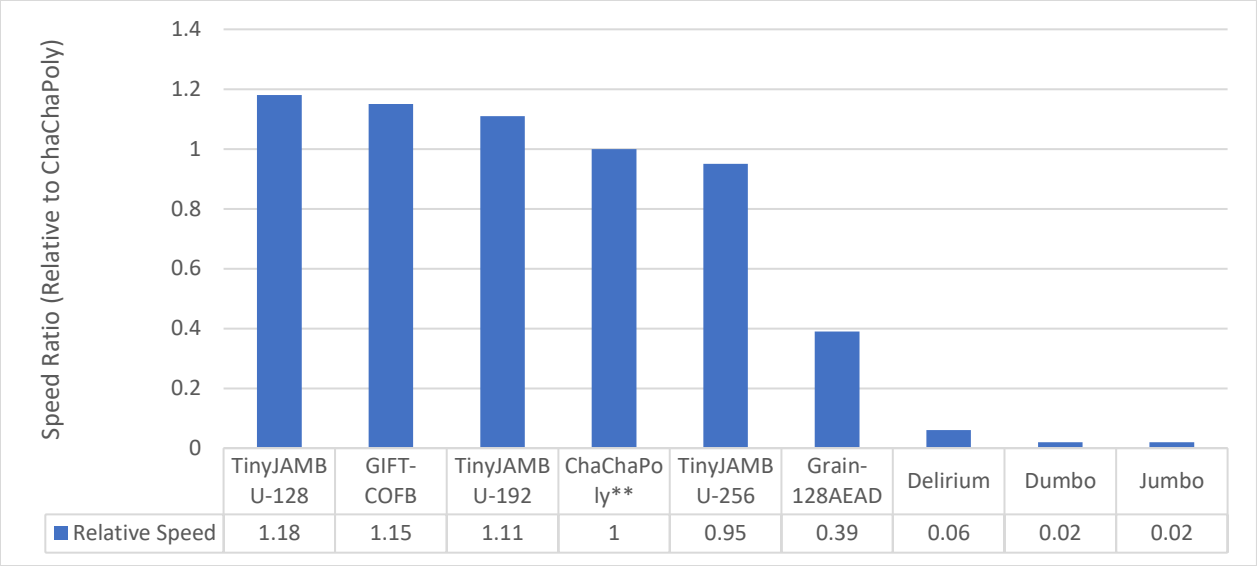

Figure 6: Relative Speed of Lightweight Encryption Algorithms.

V. CONCLUSION

This paper aimed to determine if lightweight cryptography could provide confidentiality and integrity while maintaining performance on resource constrained devices. It was found that the lightweight cryptographic algorithms, TinyJAMBU and GIFT-COFB meet or surpass these goals. Both algorithms performed better than the baseline, whereas Grain-128AEAD and variants of Elephant were slower. Due to the vast diversity in hardware in the IoT space, it will be important to have multiple algorithms to choose from when designing a secure solution. Further research will need to be completed to help to identify which algorithms work best on other resource constrained systems. Additionally, more performance-based metrics should be tested such as: memory utilization, CPU utilization, and power usage.

References

[1] J. M. Voas, "Networks of 'things':," National Institute of Standards and Technology, Gaithersburg, MD, NIST SP 800-183, 2016. doi: 10.6028/NIST.SP.800-183.

[2] M. S. Turan, "Challenges in Lightweight Crypto Standardization," p. 38, 2015.

[3] S. K. Y. Donzia, H.-K. Kim, and B. Y. Shin, "Study on Cloud computing and Emergence of the Internet of the Thing in Industry," in *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)*, Nov. 2018, pp. 334–337. doi: 10.1109/NICS.2018.8606834.

[4] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities," *IEEE Access*, vol. 9, pp. 28177–28193, 2021, doi: 10.1109/ACCESS.2021.3052867.

[5] "IoT in the Enterprise | Zscaler Blog," *Zscaler*. https://www.zscaler.com/blogs/security-research/iot-traffic-enterprise-rising-so-are-threats (accessed Nov. 14, 2021).

[6] S. Mansour and A. Lauf, "Hardware Root Of Trust for IoT Security In Smart Home Systems," in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*, Jan. 2020, pp. 1–2. doi: 10.1109/CCNC46108.2020.9045412.

[7] "The CIA Triad – Interests and Insights." https://blog.jamestyson.co.uk/the-cia-and-dad-triads (accessed Nov. 20, 2021).

[8] H. Raad, *Fundamentals of IoT and Wearable Technology Design*, 1st ed. Wiley, 2020. doi: 10.1002/9781119617570.

[9] H. N. Saha, N. F. Raun, and M. Saha, "Monitoring patient's health with smart ambulance system using Internet of Things (IOTs)," in *2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON)*, Aug. 2017, pp. 91–95. doi: 10.1109/IEMECON.2017.8079568.

[10] Schneier, Bruce, *Applied cryptography: Protocols, algorithms, and source code in C*, 20th ed. Wiley, 2017.

[11] "FIPS 197, Advanced Encryption Standard (AES)".

[12] D. J. Bernstein, "ChaCha, a variant of Salsa20".

[13] M. Sonmez Turan *et al.*, "Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process," National Institute of Standards and Technology, Jul. 2021. doi: 10.6028/NIST.IR.8369.

[14] S. Thapliyal, H. Gupta, and S. K. Khatri, "An Innovative Model for the Enhancement of IoT Device Using Lightweight Cryptography," in *2019 Amity International Conference on Artificial Intelligence (AICAI)*, Feb. 2019, pp. 887–892. doi: 10.1109/AICAI.2019.8701377.

[15] N. A. Gunathilake, W. J. Buchanan, and R. Asif, "Next Generation Lightweight Cryptography for Smart IoT Devices: : Implementation, Challenges and Applications," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, Apr. 2019, pp. 707–710. doi: 10.1109/WF-IoT.2019.8767250.

[16] R. Das and I. Das, "Secure data transfer in IoT environment: Adopting both cryptography and steganography techniques," in *2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Sep. 2016, pp. 296–301. doi: 10.1109/ICRCICN.2016.7813674.

[17] Z. A. Pindar, J. O. Fayomi, N. H. Waziri, B. M. Abdulhamid, and S. Jamel, "A Lightweight Message Authentication Code for Virtual Work in Future Smart Cities," in *2020 IEEE European Technology and Engineering Management Summit (E-TEMS)*, Mar. 2020, pp. 1–5. doi: 10.1109/E-TEMS46250.2020.9111859.

[18] S. Iyer, G. V. Bansod, P. N. V, and S. Garg, "Implementation and Evaluation of Lightweight Ciphers in MQTT Environment," in *2018 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Dec. 2018, pp. 276–281. doi: 10.1109/ICEECCOT43722.2018.9001599.

[19] G. Saldamli, L. Ertaul, and A. Shankaralingappa, “Analysis of Lightweight Message Authentication Codes for IoT Environments,” in *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*, Jun. 2019, pp. 235–240. doi: 10.1109/FMEC.2019.8795359.
[20] Y.-S. Kim and G. Kim, “A Performance Analysis of Lightweight Cryptography Algorithm for Data Privacy in IoT Devices,” in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct. 2018, pp. 936–938. doi: 10.1109/ICTC.2018.8539592.
[21] M. Engineer and A. Shah, “Performance Analysis of Lightweight Cryptographic Algorithms Simulated on Arduino UNO and MATLAB using the Voice Recognition Application,” in *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, Dec. 2018, pp. 1–7. doi: 10.1109/ICCSDET.2018.8821126.
[22] V. K. Sarker, T. N. Gia, H. Tenhunen, and T. Westerlund, “Lightweight Security Algorithms for Resource-constrained IoT-based Sensor Nodes,” in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Jun. 2020, pp. 1–7. doi: 10.1109/ICC40277.2020.9149359.
[23] K. A. McKay, L. Bassham, M. S. Turan, and N. Mouha, “Report on lightweight cryptography,” National Institute of Standards and Technology, Gaithersburg, MD, NIST IR 8114, Mar. 2017. doi: 10.6028/NIST.IR.8114.
[24] M. Hell, T. Johansson, W. Meier, J. Sonnerup, and H. Yoshida, “Grain-128AEAD - A lightweight AEAD stream cipher,” p. 37.
[25] S. Banik *et al.*, “GIFT-COFB,” May 2021.
[26] S. Banik, “GIFT-COFB: NIST LWC Second-round Candidate Status Update,” p. 7.
[27] H. Wu and T. Huang, “TinyJAMBU: A Family of Lightweight Authenticated Encryption Algorithms (Version 2),” p. 40.
[28] T. Beyne, Y. Long Chen, C. Dobraunig, and B. Mennink, “Elephant v2,” May 2021.
[29] R. Weatherley, “Lightweight Cryptography Primitives: Main Page.” https://rweather.github.io/lightweight-crypto/index.html (accessed Nov. 14, 2021).