# Power Consumption of Common Symmetric Encryption Algorithms on Low-Cost Microchips

Marc Ohm[1],[2] Lars Taufenbach,[2] Karsten Weber,[2] Timo Pohl[2]

**Abstract:** In the Internet of Things (IoT), many devices are battery-operated, making them particularly susceptible to power-hungry applications. Symmetric encryption is a regularly performed task on such devices, as it ensures the confidentiality of the data they send. While previous work has compared the power consumption of common symmetric encryption algorithms on commodity hardware, no such evaluation exists for low-cost microchips, which are often used in IoT devices. In this paper, we compare the power consumption of an ESP8266 executing common symmetric encryption algorithms with varying parameters such as key size, data authentication, or payload size. We find that the power consumption depends on several factors, but that overall AES-GCM has the lowest power consumption when the encrypted data is also authenticated, while Blowfish-CTR has the lowest power consumption when no authentication is applied.

**Keywords:** Power Consumption; ESP8266; Symmetric Encryption

## 1 Introduction

The Internet of Things (IoT) refers to the state in which everything is connected to a network and can communicate with each other. The IoT is a new approach that connects not only computers, laptops, and handheld devices such as smartphones and tablets to the Internet, but also everyday household items such as refrigerators, dishwashers, and heating systems. Moreover, low-cost microchips such as the ESP8266 enable users to extend its capabilities with general-purpose sensors and program it according to their needs.

Depending on the use case, the confidentiality and integrity of information received or sent by such devices may need to protected. One common way to achieve this is through the use of encryption and cryptographic signatures. However, low-cost microchips are generally low-performance, which can limit the efficiency of cryptographic functions. For example, battery-powered devices may need to conserve energy and therefore skip encryption altogether.

This paper compares various common symmetric encryption algorithms to determine their power consumption based on typical parameters. These parameters include the size of

[1] Fraunhofer FKIE, Bonn, Germany, marc.ohm@fkie.fraunhofer.de

[2] University of Bonn, Computer Science 4, Germany, {ohm,pohl}@cs.uni-bonn.de and {s6latauf,karsten.weber}@uni-bonn.de

the key and payload, message authentication, and microchip operation mode. The tested algorithms are AES, Camellia, Blowfish, and ChaCha20.

In the rest of this paper, we will first introduce some related work in Sect. 2 to delineate the scope of our research. Next, we explain our methodology and experimental setup in Sect. 3. We display the experimental results in Sect. 4 and present our conclusions in Sect. 5.

## 2 Related Work

Our research examines the measurement of power consumption for particular tasks and builds upon prior work in the field. For example, Ohm et al. [OPKV19] researched how ad-blocking has an impact on the energy consumption of networked devices. Additionally, measuring a hardware's power consumption during cryptographic operations can be utilized for side-channel attacks [RD20]. Despite this, there are ongoing efforts that concentrate on measuring the power consumption of cryptographic algorithms in particular.

In their paper titled "Performance Evaluation of Symmetric Encryption Algorithms on Power Consumption for Wireless Devices" [EKH09] Diaa Salama Abdul et al. compared the power consumption of six symmetric algorithms, namely RC2, DES, 3DES, RC6, Blowfish, and AES. They encrypted files of different sizes using a laptop and then wirelessly transferred them to another device. According to the measurements, Blowfish had the lowest power consumption. Blowfish utilizes only 16 % of the energy required by AES, with AES consuming approximately 4 µJ/byte, whereas Blowfish consumes around 0.7 µJ/byte. The time required for encrypting with Blowfish, which is at 50 ms, is significantly lower than that required by AES which is 350 ms. This study suggests that Blowfish is a good fit for energy-efficient encryption.

In 2017, Saurabh Singh et al. [Si17] conducted research on various lightweight encryption algorithms that are suitable for use in IoT environments. They compared several symmetric and asymmetric methods to evaluate their feasibility in the IoT environment. In addition, they evaluated the feasibility of a hybrid algorithm that uses both symmetric and asymmetric encryption. Based on the four nominal hardware values of data size, battery power, memory space, and computational power, as well as their corresponding computed thresholds, it is possible to determine which mode of the hybrid algorithm should be used.

In their paper "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish" by Priyadarshini Patila et al. [Pa16], the authors examine the time it takes various encryption algorithms to encrypt and decrypt files. For files up to 3 MB, Blowfish was shown to have the fastest encryption and decryption time with about 500 ms, followed by AES with about 600 ms. Blowfish also has a significantly smaller memory footprint of 9.38 kByte compared to AES with 14.70 kByte, and RSA has the largest memory footprint with 33.50 kByte. These results are consistent with the work of Diaa Salama Abdul et al. [EKH09], indicating that Blowfish is well suited for an IoT context.

We embed our research in an area that encompasses environmental impact, i.e., power consumption, cryptography, and security. To this end, we have chosen a real-world use case that uses a resource-constrained device to evaluate common symmetric encryption algorithms. Overall, we are able to confirm some findings of related work, but also identify divergences.

## 3 Methodology

In order to accurately measure the power consumption of common symmetric encryption algorithms on low-cost microchips, we need a low-cost microchip, precise measurement tools, and a concrete set of symmetric encryption algorithm implementations. The following sections present our hardware and software choices and our experimental setup.

### 3.1 Hardware Setup

There are many small single-board systems that can be used to build an IoT device. For this project, we are using an ESP8266, which is very inexpensive and therefore very popular. The exact device we are using is a NodeMCU LoLin V3 from AZ-Delivery [Gm22]. The power consumption is specified as a maximum of 500 mA, depending on the mode of the card. This is also the maximum power consumption during encryption [AZ19]. The ESP8266 has no hardware support for encryption, so all encryption algorithms are performed entirely in software.

A Power Profiler Kit II (PPK2) [NO21] is used to measure the power consumption accurately. The PPK2 can perform up to 100 000 measurements per second, taking one measurement per 10 µs time interval. The PPK2 is utilized to measure the current of the connected NodeMCU board.

To determine when an encryption process with a fixed set of parameters occurs, a particular GPIO pin is set to HIGH for 1 ms, and then returned to LOW. After the GPIO pin is returned to LOW, an additional millisecond pause is employed to prevent noise from being recorded during the switching process. This pause was not considered during the evaluation.

To decrease the energy consumption of the NodeMCU board, all pins, except for pin *D4*, are set to LOW by default. Pin D4 is linked to the board's in-built LED. The LED switches on when the pin is set to LOW and off when it is set to HIGH. To prevent an increase in power consumption during encoding, the LED was set to HIGH to keep it off. Moreover, if not indicated otherwise, we conduct all experiments in "modem sleep" mode, which deactivates the Wi-Fi module.

We decided not to deduct the baseline power consumption since microchips cannot multitask. Therefore, when determining the power cost of a specific encryption algorithm, it's necessary to consider the power consumption of the entire board during the encryption process.

### 3.2 Software Setup

To model the power consumption for encoding generic IoT data under consideration, temperatures measurements are encoded, stored, and represented as a 4-Byte float (data point) Since such a sensor may be used to provide live data (single data points) or historic data (batched data points) the power consumption for encoding data is tested in payload chunks of 1, 10, 100 and 500 data points respectively. These payloads correspond to payload sizes 4 Byte, 40 Byte, 400 Byte and 2000 Byte, respectively. To measure the impact of data authentication, measurements were conducted with and without data authentication.

To minimize the influence of different implementation qualities of encryption algorithms, we only test algorithms from a single library. We selected version 2.23 of the Mbed TLS library[1] for this study. This choice was based on the fact that it is one of the most popular libraries for embedded devices, and supports more encryption methods than other libraries such as WolfSSL 2. We opted for this version because it can be easily installed via the PlatformIO registry for ESP8266 projects and because mbedtls 3.X no longer supports certain encryption methods which we intended to evaluate.

We conducted tests on all block ciphers supported by the Mbed TLS library, which include *AES* [Na01], *Camellia* [MMN04], and *Blowfish* [Sc05]. Furthermore, we also tested the *ChaCha20* [NL18] stream cipher. The block ciphers are tested using Counter Mode (CTR) and Galois Counter Mode (GCM). Both authentication-enabled and -disabled evaluations are conducted. GCM has inherent abilities to produce an authentication tag. To authenticate in CTR mode, a Message Authentication Code (MAC) is generated by using the SHA256 hash algorithm. This is done by leveraging the `mbedtls_md_hmac()` method provided by Mbed TLS. Regarding ChaCha20, encryption is tested, including authentication, using the ChaCha20-Poly1305 variant (also referred to as *ChaCha-Poly*).

AES is a substitution-permutation network that can support key sizes of 128, 192, and 256 bits, using a block size of 128 bits. Camellia is also a Feistel cipher that can support key sizes of 128, 192, and 256 bits, using a block size of 128 bits. Another example of a Feistel cipher is Blowfish, which comes with a key size ranging from 32 to 448 bits (128 bits being the default), and has a block size of 64 bits. ChaCha20-Poly1305 is a cryptographic construct that combines the stream cipher ChaCha20 and the Poly1305 message authentication code. It uses a 256-bit key. Generally, a bigger block size enables the encryption of more data securely due to the birthday problem. Nevertheless, the key length is also of primary importance since brute-force searching (i.e., trying all potential keys) becomes unprofitable with longer keys.

Since all algorithms except ChaCha20 can operate with multiple key sizes, they were tested with keys of sizes 128 bit, 192 bit, and 256 bit. Blowfish is not supported by the GCM implementation included in Mbed TLS due to its small block size and is therefore only tested in CTR mode.

[1] https://github.com/Mbed-TLS/mbedtls

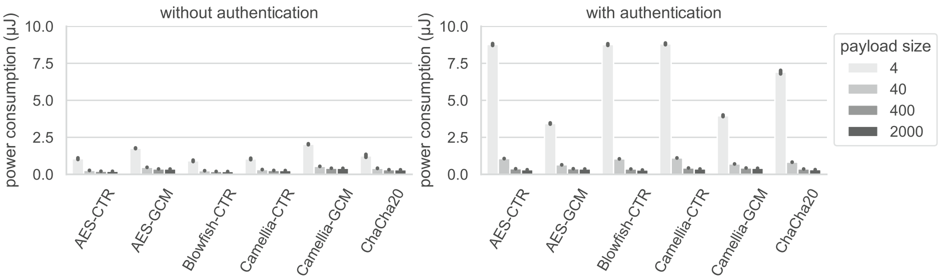

Fig. 1: Average power consumption per Byte of encryption with and without data authentication for payload sizes 4, 40, 400 and 2000. It can be seen that larger payloads are more efficient and that data authentication generally increases power consumption.

To perform encryption, all algorithms require a key and sometimes a nonce, which is a relatively small, randomly generated number. We generate 18 key-nonce pairs using the SHA512 hash algorithm, using bits 0 to 255 of the output for the key and bits 256 to 511 for the nonce. This is done to minimize the influence of a particular key characteristic on the power consumption. For keys or nonces shorter than 256 bits, the corresponding part of the hash output is truncated accordingly. As input for the SHA512 hash, 18 different strings of 14 characters each were generated using Bitwarden's password generator with all available special characters.

For each key, 100 payloads are encrypted, consisting of 25 payloads of each size (4 Byte, 40 Byte, 400 Byte and 2000 Byte). This results in 1800 measurements for each algorithm in each payload size. The number of measurements is limited by the size of the PPK2 cache, and much longer measurements would only be possible with lower accuracy.

# 4 Results

Following the methodology described above, this section describes the results of our evaluation. We first describe general trends in power consumption, and then discuss the influence of individual variables in detail.

The power consumption of encryption never exceeds 10 µJ per data point in any configuration. As visualized in Fig. 1, encrypting a single data point in a payload has dramatically higher power consumption than encrypting multiple data points at once, which can save up to 95 % of energy. The trends for different payload sizes are described below. Since only trends across different algorithms are shown, averages over all key sizes have been used.

When handling a payload of 4 Byte, AES-GCM and Camellia-GCM are the most effective encryption methods. Specifically, their average power consumption per data point is 3.44 µJ/Byte and 3.96 µJ/Byte respectively. Compared to the other encryption methods, ChaCha-Poly still has a relatively low power consumption of 6.91 µJ/Byte. However, the power consumption of block ciphers in CTR mode is significantly higher, with an average of 8.77 µJ Byte to 8.82 µJ Byte.

Even when dealing with payloads of 40 Byte, AES-GCM and Camellia-GCM remain the most power-efficient encryption methods. Specifically, their average power consumption per data point is 0.64 µJ/Byte and 0.7 µJ/Byte respectively. ChaCha-Poly consumes more power than the two aforementioned encryption methods, with an average power consumption per data point of 0.83 µJ/Byte. However, block ciphers in CTR mode still have the highest power consumption, with an average of 1.05 µJ/Byte to 1.11 µJ/Byte.

In payloads of 400 Byte, Blowfish-CTR exhibits the lowest power consumption with 0.36 µJ/Byte, followed by ChaCha-Poly with 0.37 µJ/Byte, and AES in both CTR and GCM modes with 0.38 µJ/Byte. 0.43 µJ/Byte is the power consumption by Camellia-CTR and Camellia-GCM.

In payloads of 2000 Byte, Blowfish-CTR remains the least power-consuming at 0.3 µJ/Byte, followed by AES-CTR with 0.32 µJ/Byte, ChaCha-Poly with 0.32 µJ/Byte, and AES-GCM with 0.37 µJ/Byte. At this payload size, Camellia-CTR and Camellia-GCM also have the highest power consumption with 0.37 µJ/Byte and 0.41 µJ/Byte respectively.

Moreover, we notice a significant increase in both the overall power consumption and the data authentication overhead for payloads of 4 Byte and 40 Byte in most configurations. Further discussion about the effect of the payload size is provided later on.

For all implementations, power consumption is lower without data authentication than with it. This is particularly noticeable with block ciphers in CTR mode. They consume as much as 9.9 times more power with data authentication for a payload size of 4 Byte and at most 1.47 times more at a payload size of 2000 Byte.

## 4.1 Influence of Data Authentication

To determine the power consumption of data authentication, all measurements were taken separately with and without data authentication. We calculate the average difference between the measurements in the same configuration, with and without data authentication. The increased consumption per byte for each algorithm and payload size is shown in Fig. 3 and Fig. 4.

Moreover, we have observed that the tested AEAD schemes require significantly less additional power to generate authentication tags than the schemes where the tag is generated separately, as demonstrated in Fig. 4. AEAD schemes display an average power consumption increase of 3.22 % for higher data volumes, whereas the schemes that use additional MAC generation require 40 % to 53 % more power consumption. The separately generated MAC methods consume a significantly higher amount of power for larger payloads with a

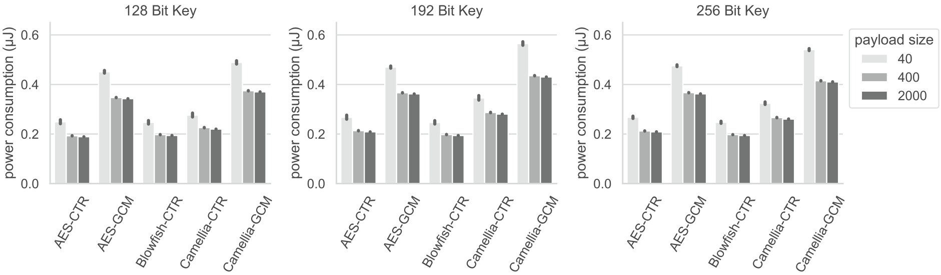

Fig. 2: Average power consumption per Byte of encryption without data authentication for payload sizes 40, 400 and 2000 grouped by key size and payload size. The payload of 4 Byte requires significantly more energy than other payload sizes and ChaCha20 has a fixed key size of 256 bit and thus both are ommited from the figure.

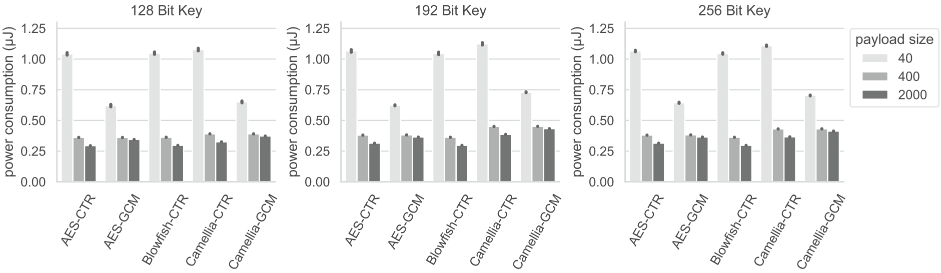

Fig. 3: Average power consumption per Byte of encryption with data authentication for payload sizes of 40, 400 and 2000 grouped by key size and payload size. The payload of 4 Byte requires significantly more energy than other payload sizes and Cha Cha20 has a fixed key size of 256 bit and thus both are ommited from the figure.

difference of over 174 μJ between payloads of 4 Byte and 2000 Byte, and thus their share of power consumption diminishes less for larger payloads compared to the tested AEAD methods.

For payloads of 4 Byte, data authentication doubles the power consumption of payload ciphers using GCM, while MACs generated with Poly1305 and SHA256 increase the power consumption of encryption by a factor of 5 to 9. Even with payloads of 40 Byte, authentication still doubles or quadruples the power consumption for all methods except AES and Camellia in GCM mode. It is only with payloads of size 400 Byte that the power consumption approaches that of unauthenticated encryption with the tested AEAD methods.

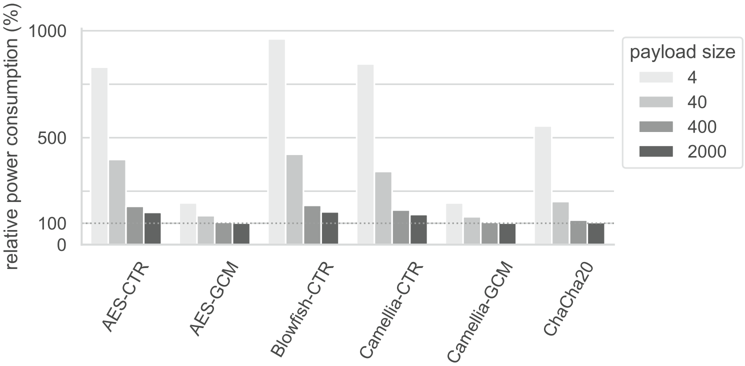

Fig. 4: Power consumption of the data authentication per Byte and in comparison to no data authentication.

### 4.2 Influence of Payload Sizes

As illustrated in Fig. 1, Fig. 2, and Fig. 3, larger payload sizes are generally more efficient. The difference between consecutive payload sizes should be considered since they offer the most significant potential for power savings. Encrypting payloads of 40 Byte instead of 4 Byte reduces power consumption by a factor of 7.4 on average, while encrypting 400 Byte instead of 40 Byte results in a reduction of 2.3, and encrypting 2000 Byte instead of 400 Byte reduces consumption by a factor of 1.13. Encrypting with payloads of size 2000 Byte can reduce power consumption by an average of 94.83 % compared to 4 Byte for all algorithms combined, with 94.15 % reduction already achieved by encrypting with 400 Byte payloads.

### 4.3 Influence of Key Size

Lastly, we assess the impact of the key size on power consumption as depicted in Fig. 5. Our findings show that there is a measurable but not significant increase in power consumption for larger keys.

Using no data authentication reduces power consumption by approximately 20 %. No discernable linear relationship exists between power consumption and key sizes. Increasing the key size does not result in a proportionate increase in power consumption. AES-CTR and Blowfish-CTR consume nearly the same amount of power. Regardless of the mode, Camellia-GCM consumes an equal amount of power on average between key sizes of 192 bit and 256 bit. Enabling authentication widens the gap between the algorithms more than when authentication is disabled.

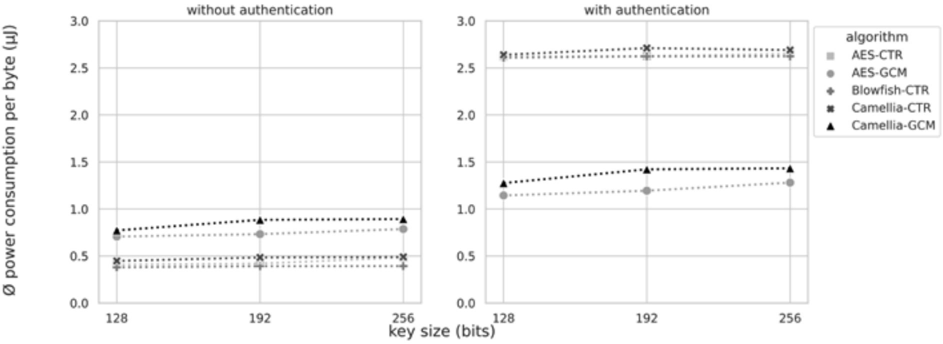

Fig. 5: The influence of key sizes on the power consumption per Byte during encryption. Only marginal changes are measured.

## 5 Conclusion

This paper evaluates and compares multiple common symmetric encryption algorithms on the ESP8266 microchip. In support of this goal, power consumption is measured during the encryption process. Experiments are conducted to evaluate the impact of several variables such as the encryption algorithm, payload size, data authentication mechanisms, and key size.

When selecting an encryption method, user data should be considered since the efficiency of an algorithm relies on the amount of encrypted data at a time. It was observed that all tested algorithms demonstrated more efficacy when encrypting larger payloads. However, it is crucial to pay special attention to the data authentication process since it can increase power consumption by up to 900 % depending on the application.

If data authentication is not considered, Blowfish performs best on average across all payload sizes. This correlates with the findings reported in related studies. It is worth noting that only for larger payload sizes (400 Byte and 2000 Byte), AES-CTR is more efficient than Blowfish-CTR.

If data authentication is required, and small payloads need to be encrypted, then AES should be used in GCM mode since it has the lowest power consumption for payload sizes of 4 Byte, 40 Byte and 400 Byte. For a payload size of 2000 Byte, AES in CTR mode with an additional MAC is more efficient. Our research found that Blowfish in CTR mode is almost as efficient as the AES algorithms in all payload sizes, although this contradicts related work. However, due to security concerns, we recommend using AES [BL16].

Furthermore, it was observed that "modem sleep" reduces power consumption by approximately 60In addition, this mode significantly reduces power consumption with minimal effects on runtime. Furthermore, certain low-cost microchips, such as the ESP32, already have hardware accelerations for AES and some cryptographic hash functions and should be taken into consideration during selection. However, chips without hardware accelerations are still commonly used.

## Bibliography

[AZ19] AZ Delivery: User guide - NodeMCU LoLin V3. https://cdn.shopify.com/s/files/1/1509/1638/files/Betriebsanleitung-AZ-LoLin.pdf?v=1605856686, 2019.

[BL16] Bhargavan, Karthikeyan; Leurent, Gaëtan: On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN. In (Weippl, Edgar R.; Katzenbeisser, Stefan; Kruegel, Christopher; Myers, Andrew C.; Halevi, Shai, eds): Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016. ACM, pp. 456–467, 2016.

[EKH09] Elminaam, Diaa Salama Abdul; Kader, Hatem M Abdul; Hadhoud, Mohie M: Performance evaluation of symmetric encryption algorithms on power consumption for wireless devices. International Journal of Computer Theory and Engineering, 1(4):343, 2009.

[Gm22] GmbH, AZ-Delivery Vertriebs: AZ-Delivery - Your expert for microelectronics. https://www.az-delivery.de/, 2022.

[MMN04] Matsui, Mitsuru; Moriai, Shiho; Nakajima, Junko: A Description of the Camellia Encryption Algorithm. RFC 3713, April 2004.

[Na01] National Institute of Standards and Technology: Announcing the ADVANCED ENCRYPTION STANDARD (AES). https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf, 2001.

[NL18] Nir, Yoav; Langley, Adam: ChaCha20 and Poly1305 for IETF Protocols. RFC 8439, June 2018.

[NO21] NORDIC Semiconductor: Power Profiler Kit II Dokumentation. https://infocenter.nordicsemi.com/pdf/PPK2_User_Guide_v1.0.1.pdf, 2021.

[OPKV19] Ohm, Marc; Prahl-Kamps, Felix; Vogel, Daniel: The environmental impact of online advertisement. In: Adjunct Proceedings of the 33rd edition of the EnviroInfo. Shaker, 2019.

[Pa16] Patil, Priyadarshini; Narayankar, Prashant; Narayan, DG; Meena, S Md: A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish. Procedia Computer Science, 78:617–624, 2016.

[RD20] Randolph, Mark; Diehl, William: Power side-channel attack analysis: A review of 20 years of study for the layman. Cryptography, 4(2):15, 2020.

[Sc05] Schneier, Bruce: Description of a new variable-length key, 64-bit block cipher (Blowfish). In: Fast Software Encryption: Cambridge Security Workshop Cambridge , UK, December 9–11, 1993 Proceedings. Springer, pp. 191–204, 2005.

[Si17] Singh, Saurabh; Sharma, Pradip Kumar; Moon, Seo Yeon; Park, Jong Hyuk: Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions. Journal of Ambient Intelligence and Humanized Computing, pp. 1–18, 2017.