Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 9071969, 31 pages
https://doi.org/10.1155/2023/9071969

Review Article

A Comprehensive Review of Lightweight Authenticated Encryption for IoT Devices

Zainab AlJabri, Jemal Abawajy, and Shamsul Huda

School of Information Technology, Deakin University, Geelong, Victoria 3217, Australia

Correspondence should be addressed to Jemal Abawajy; jemal.abawajy@deakin.edu.au

Received 20 March 2022; Accepted 23 August 2022; Published 21 February 2023

Academic Editor: Chin-Ling Chen

Internet of Things (IoT) is a promising technology for creating smart environments, smart systems, and smart services. Since security is a fundamental requirement of IoT platforms, solutions that can provide both encryption and authenticity simultaneously have recently attracted much attention from academia and industry. This article analyses in detail state-of-the-art lightweight authenticated encryption (LAE) targeted to IoT systems. This work provides a thorough description of the algorithms, and the study systematically classifies them to facilitate understanding of relevant intricacies of the schemes. Among reviewed algorithms, there is a trade-off to retain design security, resources cost, and efficient performance. ACORN is the effective scheme on various platforms in terms of utilization of resources and power consumption, while MORUS and AES-CLOC are the fastest in hardware platforms. However, they are susceptible to misuse despite their resistance to side channel attacks. In contrast, JOLTICK, PRIMATESs, COLM, DeoxysII, OCB, and AES-JAMBU are provably resistant to nonce misuse. The challenges for possible future research are summarized. Overall, the article provides researchers and developers with practical guidance on various design aspects and limitations as well as open research challenges in the current lightweight authenticated encryption for IoT.

1. Introduction

The Internet of Things (IoT) refers to a paradigm in which physical objects (IoT devices) autonomously communicate with each other and connect to the Internet via embedded devices such as sensors and actuators [1]. It is estimated that there will be approximately 75.44 billion IoT connected devices in 2025 [2]. These IoT devices enable automation, monitoring and controlling, remote processes, data collection and analytics to drive insights, generate workflows, optimize processes, and more. With these capabilities, IoT is transforming a wide range of industries including healthcare, agriculture, transportation, and the energy sector [3, 4] with massive potential improvement in sustainable social and economic development. For example, an IoT-enabled smart transportation network will improve productivity, reduce costs, and enhance public safety and security [5, 6]. IoT is revolutionizing many aspects of healthcare, such as enabling remote patient care, which benefits patients, their families, healthcare providers, and insurance companies [7]. Similarly, IoT is deployed in the energy industry to modernize infrastructure, improve operational efficiency and reliability, provide consumers with affordable energy, and enable industry to monitor and access energy sources [8]. Table 1 lists the acronyms and abbreviations throughout the paper.

While IoT provides ample opportunities for digital transformation, it poses significant security threats [9–11]. These threats include distributed denial of service attacks [10], botnets [12], privacy and confidentiality breaches [11], and IoT-targeted malware [13]. IoT data flows through the network and requires data encryption and authenticity to ensure confidentiality (C) and integrity (I). For example, ensuring C of vital signs observation from a remote patient monitoring system is crucial. Although it is important for medical practitioners to trust the signs observation they receive as an authentic data, IoT devices have very minimal security settings in place. Consequently, there is hardly any encryption facility for securing sensitive and critical information. Furthermore, there is no built-in mechanism to ensure data authenticity. Since tampering with IoT data

Table 1: List of acronyms and abbreviations.

Acronyms	Abbreviation	Page
IoT	Internet of Things	1
LAE	Lightweight authenticated encryption	1
C	Confidentiality	1
I	Integrity	1
AE	Authenticated encryption	2
ASIC	Application service integrated circuit	2
FPGA	Field programmable logic array	2
PKI	Public key infrastructure	2
SSL	Secure socket layer	2
TLS	Transport layer security	2
ECC	Elliptic curve cryptography	2
MAC	Message authentication code	2
SSO	Single sign on	2
ECDH	Elliptic curve Diffie Hellman	2
CoAP	Constrained application protocol	2
NIST	National Institute of Standards and Technology	2
CASEAR	Authenticated encryption security applicability and robustness competition	2
AVR	Atmel microcontroller	3
MSP	Mixed signal microcontroller	3
ARM	Advanced reduced instruction set computer	3
AMD	x86 instruction set	3
LoRaWan	Low range wide area networks	5
AMQP	Advanced message queuing protocol	5
MQTT	Message queue telemetry transport	5
IND	Indistinguishability	6
NM	Non-malleability	6
IND-CPA	Indistinguishability-chosen plaintext attack	6
IND-CCA	Indistinguishability-chosen ciphertext attack	6
NM-CPA	Non-malleability-chosen plaintext attack	6
INT-PTXT	Integrity-plaintext attack	6
INT-CTXT	Integrity-ciphertext attack	6
E	Encryption oracle	6
D	Decryption oracle	6
K	Key	6
KM	Known message	6
KC	Known ciphertext	6
CM	Chosen message	6
CC	Chosen ciphertext	6
AD	Associated data	8
AEAD	Authenticated encryption with associated data	8
EaM	Encrypt-and-MAC	9
EtM	Encrypt-then-MAC	9
MtE	MAC-then-encrypt	9
CTR	Counter mode	9
GCM	Galois counter mode	9
CWC	Carter Wegman counter mode	9
HTTP	Hypertext transfer protocol	9
UDP	User datagram protocol	9

Table 1: Continued.

Acronyms	Abbreviation	Page
DTLS	Transport layer security	9
IPsec	Internet protocol security	9
IAPM	Encryption modes with almost free message integrity	10
OCB	Block cipher mode of operation for efficient authenticated encryption	10
EPBC	Efficient error-propagating block chaining	10
CLOC	Low overhead counter feedback mode	10
CCM	CBC-MAC	10
XOR	Exclusive-OR gate	10
ROM	Read only memory	11
RAM	Random access memory	11
LUTs	Lookup tables	11
GE	Logic equivalent	11
NAND	Not-AND	12
Bps	Bits per second	12
V	Variable	15

could have a profound impact, it is a practical necessity to minimize the manipulations and the exposure of sensitive information to malicious parties.

With the growing number of IoT-born attacks, the need for securing the IoT and data has recently received a significant attention from the industry and research community [13]. Generally, the dominant basis for security is the application of cryptographic algorithms to ensure C and I of information. IoT devices are constrained by the area footprint, power consumption, energy, and throughput [14]. Therefore, the conventional cryptographic algorithms are computationally expensive, which is potentially unreasonable for conveying restricted device requirements. The study in [15] highlighted the importance of lightweight cryptography for IoT devices and particularly LAE for improving IoT device security. This has generated a substantial amount of literature with promising cryptography schemes as security solutions for the IoT. However, these published studies are commonplace and researchers and developers should focus on collating them for use.

LAE promises better efficiency and security compared to conventional cryptography, and is suitable for IoT applications restricted by resources, power consumption, and energy. Explicitly, it combines data C and authenticity services in one algorithm. Less prerequisite than separated encryption composition. It encrypts and authenticates the messages in order to protect users and secure data network communication. Typically, it can be integrated between sensing layer and network layer, within various connections of network layer, between network and service layers, and between service and interface layers.

Existing reviews on authenticated encryption (AE) have been published [15–20]. These studies are proposed for industrial IoT, smart cards, sensors, smart low-resource devices, smartphones, and power grid systems. However, these studies are fragmented in terms of security, not comprehensive enough to enhance understanding of relevant aspects, and focusing on general high-level performance issues. The security threats and parameter specification were not considered in [17]. Data authenticity that occurs in transit or at storage cannot be detected because data integrity countermeasure is not considered among the reviewed schemes. Schemes' functionality criteria and their security requirements were not discussed in [15]. The study in [18] did not include compelling security justification and instead discussed cryptographic schemes, which are proved to be no longer safe for use [16, 21]. One recent review [22] has studied lightweight cryptography, considering two separate schemes as lightweight block cipher primitive for encryption, and lightweight hash function for authenticity.

The work of [17] focused on block cipher algorithm performance in application service integrated circuit (ASIC). Similarly, the focus of [18] is lightweight block cipher implementation in ASIC and field programmable logic array (FPGA). Block cipher algorithms were reviewed in [19], but did not consider the resources cost of the schemes by imposing computationally expensive schemes for IoTs such as Twofish [23]. The study by [24] evaluated the performance metrics of lightweight cryptography. However, the LAE scheme was not considered in these reviews as a built-in solution. The review [25] determined the performance of CASEAR algorithms for IoT but did not consider security threats imposed due to IoTs are being placed in uncontrolled environment. The survey in [20] explored Dexoys, MORUS and POET studied in FPGA, Intel Core i7-4770 and Intel Core i5-3210M platforms, but MORUS and POET are prone to nonce misuse and provable forgery attacks, respectively, [26]. In [20], the scheme Dexoys has two forms with diverse nonce misuse feature, but this work surveyed the version threaten to nonce manipulation attack. Overall, LAE were surveyed for efficiency performance metrics and eliminating security threats and resources cost [15, 20].

Targeting IoT applications or resource-constrained devices is usually seen from performance perspectives either for low power, low area metrics, or high throughput, disregarding the security of the algorithms. Most existing reviewed schemes are based on lightweight symmetric algorithms (cryptographic scheme where two parties share the same key) [27, 28], while LAE should be further studied for IoT. In [27] and [28], lightweight symmetric encryptions were designated for restricted IoT; however, a communication header can be manipulated without detection. Thus, they require additional hashing scheme to validate data I, which adds to the computational expenditure. To ensure C and I objectives, we need to validate data and provide security controls for protection against different types of IoT attacks. Nonetheless, it is very challenging to deploy endpoints security controls such as malware guards and conventional security. Conventional endpoints security controls cannot be applied as the IoT gateway network, and local endpoint networks have different protocols [29]. Constrained devices also restricted the ability to use a public key infrastructure (PKI) based authentication scheme such as Secure Socket Layer (SSL) and Transport Layer Security (TLS). This is due to the expenditure imposed by the PKI hardware implementation, the SSL certificate maintenance, and key management for a large number of IoT devices [30].

Lightweight identity approach was introduced by eliminating PKI certificate requirement [31]. However, the algorithm is based on certificateless elliptic curve cryptography (ECC), hash-based message authentication code (MAC), and secure hashing algorithm, which are conventional mechanisms requiring computational resources. Authenticating a large number IoT nodes in a hostile environment that stores secret keys imposes physical security loss, which in turn will affect the entire network and jeopardize the IoT system [32]. Manual installation of common symmetric keys is very difficult for numerous numbers of devices. To reduce the workload of key management, single-sign-on (SSO) based approaches can be applied that require user interaction, and numerous devices require self-authentication. The application of SSO-based protocols is limited since it requires a user response [33]. Self-authentication can be combined with elliptic curve Diffie Hellman (ECDH), and the initialized key, yet this approach requires high computation compared to lightweight cryptography [34].

The cloud-assisted IoT system includes an IoT gateway, cloud, and service server falls under the TCP/IP protocol [35]. However, the IoT nodes and the IoT gateways have different protocols, for instance the constrained application protocol (CoAP). Although CoAP is a popular lightweight IoT protocol, it is susceptible to many well-known attacks [36, 37]. Unfortunately, IoT nodes are constrained by resources and due to their heterogeneous nature, deploying conventional mechanisms may not be practical. Recently, the research community has shown interest in AE due to National Institute of Standards and Technology (NIST) calling for lightweight ciphers and the finished AE security applicability and robustness competition (CASEAR). NIST has been conducting the lightweight cryptography project to select one or more LAE and hashing schemes suitable for constrained environments including IoTs. In 2019, fifty-seven candidates have been submitted to the project which fifty-six schemes are accepted as a first round. Many schemes eliminated for the second round due to their limitation on providing security criteria and implementation characteristics including performance and cost. In the second round, thirty-two schemes were qualified. Recommended schemes are expected to resist side channel attack to provide additional security level against physical attacks. Such criteria are significant for IoT devices because they are deployed in hostile environment and unattended places. Thus, devices are protected from replication and manipulation [38].

Internal reviewers as well as the wider community conduct analysis and evaluation of candidates. After rounds of candidates' elimination, ten become finalist recently. These are ASCON, Elephant, GIFT-COFB, Grain-128AEAD, ISAP, PHOTON-Beetle, Romulus, SPARKLE, TinyJAMBU, and Xoodyak. NIST encourages public evaluation and publication of schemes security, implementation, and performance. According to [39], TinyJAMBU employs minimum resources compared to ASCON, Romulus, GIFT-COFB, and Xoodyak, but supports a low throughput. Xoodyak, in contrast, achieves the highest throughput to facilitate low-latency application requirement but resources are expensive. Of these schemes, ASCON, GIFT-COFB, and Romulus have shown a reasonable trade-off between resources and performance. ASCON was a winner in the CASEAR competition besides ACORN. Both schemes have been targeting constrained devices in terms of security, applicability, and robustness. However, schemes are yet to be evaluated for IoT platforms. As well, the standardization process is ongoing, with calls to investigate LAE preferences for IoTs. Subsequently, LAE is a promising option for IoT devices and motivates future research into lightweight cryptography.

In this paper, we present a review of state-of-the-art LAE for IoT taking into account various aspects of the algorithms including potential attacks. The cipher design parameters such as block size, and key size are crucial parameters for providing security, in addition to other attacks targeting AE such as nonce misuse attacks and side-channel attacks. This study proceeds based on the existing approaches to address the challenges of providing secure and effective AE solutions for IoT applications. The review provides a thorough description of the algorithms and systematically they are classified to facilitate an understanding of the related intricacies and relevant aspects. It discusses how these ciphers resist various design attacks by considering the design characteristics and underlying primitives. We also introduce a system model and threat model for AE applicability in IoT.

Our study comprehensively reviews a benchmark of LAE and promises the security based on the literature. It also reviews the effective performance of IoT based on several platform-related metrics, with an observation of the relevant challenges. The contributions of this review are as follows:

(i) We present a generalized IoT system model that is suitable for explaining AE within different model layers. We highlight various IoT challenges based on the system model, system requirements, and introduce threat model with applicable IoT applications

(ii) We present a comprehensive state-of-the-art LAE method for IoT platforms. We review the construction of AE design classifications and underlying primitives. We reviewed and examined the schemes based on their design security including C and I security, nonce misuse property, and side channel attacks

(iii) We compare discuss the challenges of LAE performance benchmarks and introduce functionality criteria, platform awareness, and resource limitations as factors for comparison fairness. Moreover, we comparatively reviewed the most prominent performance in different platforms including FPGA, ASIC, 8 bits Atmel microcontroller (AVR), 16 bits mixed signal microcontroller (MSP), 32 bits advanced reduced instruction set computer (ARM), and x86 instruction set (64 bits AMD).

(iv) We discuss the challenges and key issues in LAE for IoT, which will be useful for future research

The remainder of this paper is organized as follows. Section 2 introduces the IoT system model and Section 3 IoT system requirements. Section 4 models schemes' threats. Section 5 introduces the state-of-art LAE algorithms and their limitations, with reference to the existing literature on the design, classification, and basis primitives. Section 6 comprehensively discusses the algorithms' functionality, while Section 7 comparatively discusses the security vulnerability, functionality, and performance aspects. Section 8 discusses open problems and finally Section 9 concludes this review.

2. System Model

To illustrate this work's system requirements, we have modelled a four-layer IoT architecture compiled from the International Telecommunication Union IoT architecture [40]. The model includes sensing layer, network layer, service layer, and interface layer, where they interact according to the use-case. Each layer illustrated in Figure 1 should provide sufficient data for I and C in transmission besides other securities such as access control and availability. The IoT data are transmitted between the layers composed of a data header and a payload depending on the technology. For example, low range wide area networks (LoRaWAN) constitute the most applied technology for transmitting sensor data with a maximum packet size of 255 bytes includes 13 bytes header [41].

The system architecture must deliver functional guarantees and security requirements for the IoT to bridge the gaps between physical devices and virtual world. Interconnection of IoT devices enables information collection, aggregation, exchange, processing, and proper data interpretation. At the sensing layer, existing embedded devices are enabled to collect information and exchange them with each other. For instance, low-power sensors attached to a patient facilitate remote monitoring of medical devices (i.e., wearable devices and smart sensors), and send the stored record to the network layer. The encapsulated record in this example includes alerts, medication dosage, condition status, and risky private information [42].

The main concerns in IoT sensing layer determination include things size, cost, resources, energy consumption, hybrid network, and communication [43]. Devices are equipped with sensing capability such as radio frequency identity tags, sensors, and actuators. These devices should be designed with the aim to minimize resources as well as deployment costs. Heterogenous issues involve communication via hybrid networks to enable information exchange between things. IoT is expected to interconnect with industrial networks to facilitate smart services. At the sensing layer, the devices have limited power consumption and restricted resources.

The network layer plays a vital role in connecting the things together and exchange sensed data with surrounding awareness. Data aggregation, encapsulation, and routing with regard to the network protocol are processed in the network layer. Various networks are connected and several protocol types serve to connect low-power nodes together, such as IEEE802.15.4, IEEE 802.15.1, and LoRaWAN. Reliable communication facilitates the encapsulation and routing, such as CoAP, advanced message queuing protocol (AMQP), and message queue telemetry transport (MQTT). The network varieties cause security issues, deployment difficulties, and communication issues. Of these concerns, data C and user privacy are critical due to mobility, complexity, and deployment [43]. Since some IoT devices are physically placed in untrusted environments, they risk user identification and threaten to device manipulation. Attackers can create faulty messages, which disturb the network's functionality and isolate devices from the network.

The service layer consists of middleware devices to provide collaborative IoT services related to identification, authorization, aggregation, decision support, and reactions. These technologies cooperate with services and IoT applications to provide a cost-effective product. Several types of hardware and software utilized with supported service protocols help to achieve user objectives. In this layer, the IoT demonstrates middle service activities. Service organization and providers develop various standards to undertake these activities. The service layer facilities the activities based on common application, service protocols, and application programming interfaces. The services include processing, analysis, integration, management, security, and user interface. These services input processed information, collaborate, and provide results to the user application layer. Security of this layer should be able to protect the operations from privacy leakage of location tracking, information manipulation, spoofed information, faulty routing path, and more.

The interface layer overcomes various technology vendor interconnections, where the searching service is integrated. This layer helps to identify and match application requirements. Application maintenance demands secure remote configuration of I and C transmission between the

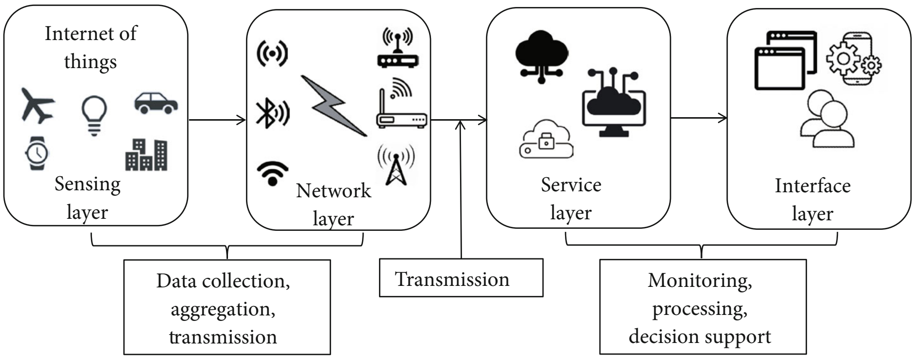


Figure 1: Typical IoT system architecture.

layers, secure software update and download, and administrator authentication. Users then view results and decisions employing an application on their smartphones, and personal computers [42, 44]. Most IoT devices are constrained and designated security solutions should consider resources, energy efficiency, and performance. The light computation security schemes at constrained devices are in high demand; however, proposing a secure solution with these aspects prior to deployment is critical [45].

3. System Requirements

The aim of the AE is to ensure C of message (i.e., the contents of transmitted message will not be read by unauthorized recipient) and I (i.e., the message have not been modified during transmission by any means) of the messages exchanged between the genuine sender and receiver [46]. IoT application collects, exchanges data, and transmits the data to remote servers such as cloud computing for further processing and critical decision-making. The data is critical and personal and for example in health-related information, it requires stringent C against data disclosure [47]. Maintaining data I provides a means of detecting unauthorized data manipulation [48]. Such assurance is important for IoTs because they are prone to receiving forge data. Forged data is applicable since devices can be accessed, managed, and connected to several things from various places [49].

C prevents the disclosure of messages to parties not authorized to view the message or the decision [50]. To satisfy the C requirement, AE defines data C by three notions: it must be capable of Indistinguishability (IND) and Non-malleability (NM). IND and NM prevent against Indistinguishability-chosen plaintext attack (IND-CPA), Indistinguishability-chosen ciphertext attack (IND-CCA), and Non-malleability-chosen plaintext attack (NM-CPA) [51, 52]. Furthermore, I detects the forge message and/or tampered data. It also thwarts IoT functionality from false response that disturbs the IoT operation [53]. To enable I detection requirement, the check must be guaranteed from the time data has been created, transmitted or stored by illegitimate users. It must also be able to defend against Integrity-plaintext (INT-PTXT) and Integrity-ciphertext (INT-CTX) attacks [54].

4. Threat Modeling

Threat models for AE as stated in CASEAR as well as NIST target conventional confidentiality and integrity proofs, and leave nonce robustness and side channel attacks as optional threats [55]. Definitions of AE security assume that received encrypted data will go through the decryption oracle and the whole plaintext is recomputed [56]. Using a conventional algorithm, data recovering, recovering keys, manipulation of encrypted data constitute threats. These threats are examined using properties of IND and NM besides different integrity violations against INT-PTXT and INT-CTX. However, using these models, it has been identified that real world threats including information leak via side channel attacks cannot be captured [57]. In practice, nonces also have been breached via repetition or manipulation [58]. To address these risks, we expand our model to include two more threats. First, nonce is controlled by adversary and nonce randomness is not guaranteed. Second, the packet header will not be encrypted for routing purposes but can be manipulated by the adversary. Thus, our model threat can be seen a practical way to assess data cryptography violations.

The system shown in Figure 2 describes a communication channel from an authorized sender A to intended receivers B and F, and the adversary H, where the ability of the adverse situation is characterized by the link capabilities. The message is transmitted from nodes through a gateway to the Internet in order to be accessed by users. The transmitted message is composed of a payload, packet header, and a tag. In this analysis, the focus is on how the attacker successfully breaches the message encryption and gains knowledge, which in turn lead to content or/and successful verification of a faulty payload or/and a faulty header tag [59].

The adversarial model is defined by IND-CPA, IND-CCA, INT-PTXT, INT-CTXT, and NM-CPA, fault attack, and forgery attack. The adversary targets message C and I by launching these attacks on LAE. The threats of breaching schemes lead to access transmitted message, key recovery, message manipulation, and packet header manipulation. These threats are achievable with prior knowledge related to the scheme as summarized in Table 2. These include the knowledge of encryption oracle (E), decryption oracle (D),

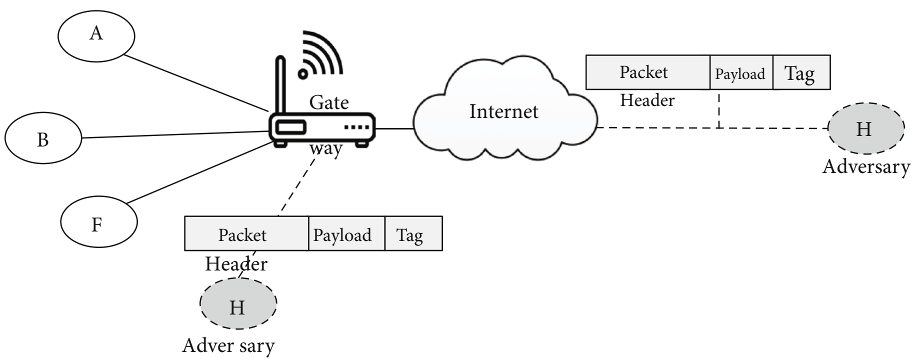


Figure 2: Adversaries source of threats as *H* while *A*, *B*, and *F* are honest participants.

Table 2: Attacks of algorithms with confidentiality or integrity breach and adversaries knowledge.

Attacks	Type		Knowledge							Threat[a]
	C	I	EO	DO	K	KM	KC	CM	CC	
IND-CPA	√	×	√	×	×	√	×	√	×	(a)
IND-CCA	√	×	×	√	×	×	√	×	√	(b)
NM-CPA	√	×	√	×	×	√	×	√	×	
INT-PTXT	×	√	√	×	×	√	×	√	√	
INT-CTXT	×	√	×	√	×	×	×	√	√	
Fault	√	×	√	√	×	×	×	×	×	(a), (b)
Forgery	×	√	×	√	×	×	×	√	√	(c), (d)

[a] Threats are (a) recover the secret key, (b) recover the message, (c) manipulate packet header, (d) manipulate message.

key (K), known message (KM), known ciphertext (KC), the chosen message (CM), or the chosen ciphertext (CC).

Breaches of C occur when any of IND-CPA, IND-CCA, NM-CPA, and fault attacks is violated. These lead to data disclosure, where eavesdropper reveals data without permission and infers some useful information that consequently helps to establish a series of attacks [60]. It can also be used to compromise the device and obtain privileges. For example, an adversary who can intercept and exploit a smart home locker can access house controls and result in multiple thefts by taking the owner benefits for future theft [61]. In home utilities, users share habits, lifestyles, browsing interests, and activities that can be revealed by the attacker. IND has been extensively studied in [62–64]. It ensures that an adversary who chooses two messages—Ma and Mb, has a ciphertext of one of them—cannot distinguish which message corresponds to the ciphertext.

IND-CPA is breached when the adversary has access to encryption oracle and one CC results from two KM. The adversary has to distinguish CC corresponding to KM in order to breach the message C. The vulnerability of this attack allows the attacker to recover the messages underlaying a selected ciphertext or the secret key and then disclose the message. For example, *H* node is impersonating *A* node intended to send a message to *B*, and the message includes a content related only to A node. He/she can manipulate the packet header to present *A* node if he/she breaches IND-CPA. Furthermore, IND-CCA is subverted by an adversary using the machine decryption to answer his/her own requests and break into the underlying system that chose the ciphertext. This was queried according to a previously known message and ciphertext pairs [65, 66]. Maintaining this knowledge, he/she obtains a bilateral proof of knowledge from a user that allows him/her to generate the same proof of knowledge to another user. For example, *H* node has the decryption machine oracle of *B* node and receives a message from F node, and then is able to playback some of the previously known pairs that were received by *B* node earlier. An NM-CPA violation implies the non-malleability and a C threat. Attacker H node reconstructs a ciphertext that is differentiated from a known message to compromise NM-CPA [51, 67]. The attacker has the same knowledge of the chosen plaintext attack and manipulates the packet header to present himself as honest *A*, *B*, or *F*.

Fault attack is a technique to inject or modify errors using voltage, power, glitching clock, and laser fault injection, which threatens C [68]. An interesting consequence leads to an error output that can be analyzed to reveal secret information, such as recovering cryptographic secret keys [69]. To be effective, it is implemented through several approaches such as timing attack, differential and simple power analysis, statistical fault attack, differential fault analysis, and collision fault analysis [70, 71].

The schemes data I is breached when INT-PTXT, INT-CTXT, or Forgery attacks are violated. Breaking the I of the message achieved by adversary *H*, who generates a ciphertext that is mapped to a meaningful message that has never been generated previously by the honest user *A* and present himself as *A*. This attack requires knowledge of the encryption oracle, chosen message, and its corresponding ciphertext. In this scenario, if the algorithm does not protect the security of INT-PTXT, H changes the packet header to be the same as A by sending this encrypted message to B or F without detection.

For the attacker perpetrating INT-CTXT, the I of the message requires the production of a ciphertext that is never generated previously, irrespective of whether the plaintext is new or meaningful. In this scenario, the adversary enquires about the ciphertext by accessing the decryption oracle. If the algorithm is not secure against ciphertext I, the adverse H manipulates the ciphertext sent by B to a different ciphertext that has never been transmitted before to the honest A, and A accepts. An example is an attacker modifying the alert in a smart healthcare system to a fault alert stating that the patient is in danger, or alters patient medicine configuration to increase the dosage. It is thereby accepted as a lawful alert and causes complications or even death [72]. In a smart city, the risk of manipulation can switch off the device causing an interruption [73].

Forgery attacks infringe on I assurance. It occurs when the mechanism is prone to causing message payload manipulation but the message tag is accepted as valid [46]. The attack exploits the algorithm leaked state information to improve the differential success [74, 75] and modify message payload and its packet header [76]. The adverse H launches a chosen plaintext forgery by obtaining the corresponding ciphertext of different consecutive blocks of the last algorithm block. Then, forges messages occur. This can be achieved by deleting one ciphertext block and duplicating another block or substituting the blocks. The forgery in this scenario is then not detected where the decryption results in the same I tag as the original legitimate message payload.

Fault attack is a technique devised to inject or modify erroneous using voltage, power, glitching clock, and laser faults injection threatens the confidentiality and leads to key recovery [68]. An interesting consequence is error output, which can be analyzed to reveal secret information like recovering cryptographic secret keys [69]. To be an effective attack it is implemented through several approaches, for example timing attack, differential and simple power analysis, Statistical Fault Attack, Differential Fault Analysis and Collision Fault Analysis [70, 71].

4.1. Other AE-Related Threats.

Extracting cryptographic key is a vital threat resulting from several attacks including fault attack, forgery attack, and node impersonation attack [77, 78]. Physical access to the IoT devices increases the chance of inducing faults so a device operates in abnormal configuration, which results in a leakage. An example for this is leaking a secret key in the power source while the device operates [79].

Since the key can be stored in the IoT node, it is prone to clone attack that mimics the legitimate key materials and other information [80]. It makes it possible for a replica node to participate in the network with similar capabilities to an authorized node showing knowledge of the key. Key agreements without a secure connection is also pose a threat if there is not well-established asymmetric algorithm or secure pairing methods authenticating IoTs and preventing nodes impersonation and other node threats [77, 78]. If a message is encrypted by the key stored in an impersonated node and sent to the destination, the latter will validate the authenticity of the sender and retrieve the information successfully since it contains the knowledge of the key [81].

The IoT connectivity problem highlights the agreements problem such that different network technologies for mobile communication (2G, 3G, 4G, and 5G) and wireless network (Bluetooth low energy, WiMax, and LoRaWAN) used to communicate sensors data to users applications [81]. There are various key agreement schemes studied in the literature: ElGamal [82] and ECC [83], Lattice based asymmetric cryptosystem [84] and Password based authentication [78]. Nonce plays a significant role in protecting modern LAE schemes against the threat of two identical messages being encrypted into two related ciphertexts [85]. When schemes are unsecure against nonce misuse, the attacker manipulates the nonce or repeats them. In other algorithms, designers assume specific nonce configuration upon implementation, which does not withstand attacks [86]. The vulnerability of these threats is message patterns leak; the messages become inevitable, and hence recovered by the attacker. In some algorithms, the damage is worse [87].

5. State-of-the-art Lightweight Authenticated Encryption and Their Limitations

LAE was designated to consume less computation compared to conventional algorithms in current communication protocols. It tolerates a small footprint, minimum power consumption, and low energy. Additionally, it provides a desirable security level and can be easily integrated with existing protocol algorithms and restricted embedded devices. The encryption and decryption engines with interfaces are shown in Figure 3. The encryption oracle takes the message, key, and a nonce as inputs to produce a ciphertext and an I tag. The decryption oracle is used to verify the tag I and decrypt the ciphertext. It uses the shared secret key, and the same nonce to return either the message as plaintext or a special symbol to indicate counterfeit ciphertext.

AE schemes were constructed to encrypt and authenticate the message only; however, as they emerged in application, there was a need for additional data to be authenticated but not encrypted. Routing headers in the TLS protocol, for example, should be kept clear so that the packet is routed to the destination. If the header is encrypted, the routers cannot read the routing details and hence it is difficult to forward the packet to the final destination [88]. For this reason, there is a need to provide AE protection and authenticity to secret data and the authenticity of other associated data (AD). This leads to the existence of AE with associated data (AEAD), where the headers or nonsecret data are called AD [89–91].

The AE design schemes attractive to researchers and many schemes have been devised. Today, these schemes are targeted by designers and standardization authorities for their capabilities and design computation lightness. The widely paradigm "generic composition" combines the

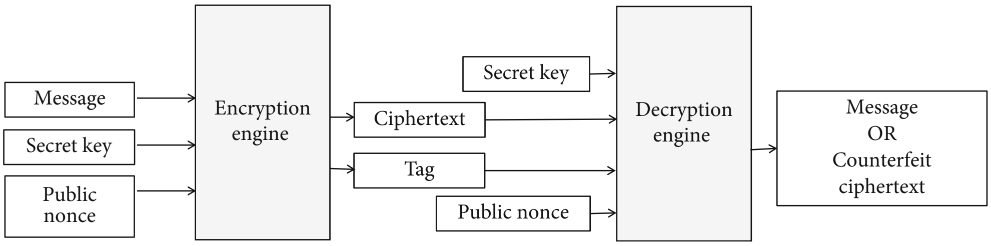


Figure 3: Authenticated encryption and decryption engine interfaces.

encryption algorithm with MAC [92]. However, many studies demonstrate that AE schemes can be more efficient than the generic composition [93]. As a result, various schemes are in demand for desirable features like robustness against nonce misuse, plaintext leakage, performance, and resistance to side channel attacks.

Among various AE schemes, the treatment of the public nonce should be properly protected to ensure I. The treatment of nonces can be clarified by using compositions such as MAC-then-Encrypt (MtE), Encrypt-then-MAC (EtM), and Encrypt-and-MAC (EaM) [56]. In the case of EaM and Encrypt-then-Mac (EtM), the nonces have to be encapsulated for I verification [94]. This is mainly because an attacker can manipulate a tag without being detected, where the MAC can still be verified. However, MtE does not require the nonce's inclusion since any change in it is detected by generating an invalid MAC. Accordingly, threats of inadequate implementation are increasingly higher in EaM and EtM [95].

Efficiency of scheme can be enhanced by implementing them in parallel. However, selecting the mode of operation, algorithm different computations, and the properties of an algorithm affects the resources cost. For example, in the same hardware platform fully parallelizable AES in Counter Mode (CTR) requires an estimation of 90 K gates, whereas the Galois Counter Mode (GCM) mode needs an additional 30 K gates and Carter Wegman Counter Mode (CWC) requires an additional 100 K gates for I [96, 97]. These key criteria have to be considered when industrial engineers require the best possible specifications for an IoT platform.

The popular IoT protocol CoAP is deployed as a lightweight protocol but it is insecure. It employs a hypertext transfer protocol (HTTP) translation based on the packet loss of the user datagram protocol (UDP). Several security studies on CoAP reviewed in [36, 37, 98] reported attacks including, but not limited to, DoS, spoofing attack, sniffing, hijacking, cross-protocol attacks, parsing attacks, amplification attacks, replay attacks, and relay attacks. To address these attacks, encryption is deployed using Datagram Transport Layer Security (DTLS) and Internet Protocol Security (IPsec). However, binding CoAP with DTLS or IPsec is deploying conventional cryptographic, which increase the computation expenditure, and does not address the scarce resource problem. Furthermore, it is revealed the handshake messages cause fragmentation. These drawbacks highlight the importance of protection using lightweight solutions.

5.1. Authenticated Encryption Algorithm Taxonomy.

We present an AE taxonomy that is divided into two-pass and single-pass based on the number of runs for data processing [99]. The taxonomy constructs a single-key and two-keys category based on the number of secret keys required to provide C and I components, which are either one identical key or two different keys. This classification is shown in Figure 4, where each category is then further constructed according to the scheme primitive as block cipher scheme, generic composition, tweakable block cipher, or permutation-based scheme. The algorithms, which are constructed based on these schemes, are listed under the group.

The single-pass scheme involves processing one run to compute the encryption and tag using one key, or two separate keys for each run. This scheme is a single pass-through on data to obtain the ciphertext and the tag. A typical approach uses a single key for the encryption/MAC engine or two keys: one to generate a ciphertext and the other for the authentication tag. This strategy reduces the computation overhead compared to the two-pass approach by approximately half of the required efficiency. IAPM (Encryption Modes with Almost Free Message Integrity) [100] and COFB [101] are under the one-key group while OCB (Block Cipher Mode of Operation for Efficient Authenticated Encryption) [102], and EPBC (Efficient Error-Propagating Block Chaining) [103] operate under the two-key scheme. IAPM and OCB are parallelizable, in contrast to EPBC and COFB, which are unparallelizable. OTR [104], OCB3 [105], TAE [106], PFB [89], AEZ, Dexoys, and Joltik are classified as tweakable one-key which are parallelizable except PFB.

The two-pass scheme is grouped into a generic composition of existing algorithms using two distinct keys and the AE mode of operation, which employs a single key. Generic composition is a technique combining conventional encryption and MAC algorithms to provide C and I service. It is classified based on how the integration functions in three modes MtE, EtM, and EaM. MtE calculates the integrity tag (MAC) of the message using the sender's first key (K1) by the sender. The MAC is concatenated with the message and then encrypted under a different key (K2). On his behalf, the receiver decrypts the ciphertext to recover the plaintext and MAC. Message MAC is calculated and verified against the acknowledged MAC tag. The message is accepted only if the tag is verified. In EtM, the sender encrypts the message, computes the MAC and appends it for exchange. At the receiver's end, the received MAC is verified against the calculated

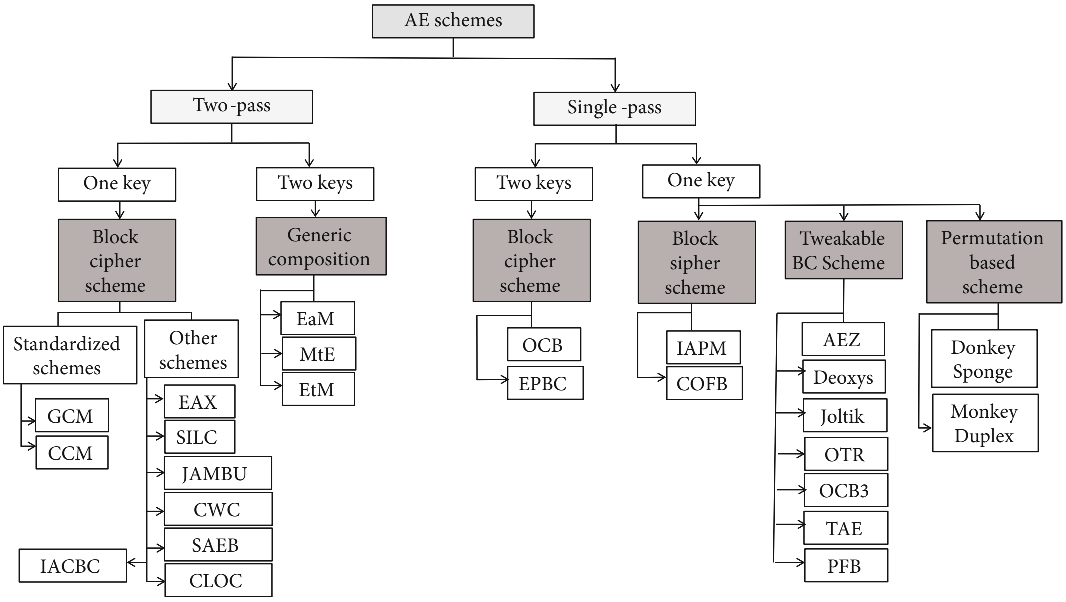


Figure 4: Taxonomy of authenticated encryption schemes.

MAC and then decrypts the ciphertext for the message. Referring to the EaM mode, the MAC tag and ciphertext are deduced simultaneously. The receiver decrypts the ciphertext and verifies the computed message MAC with the received MAC for I assurance. The EtM composition is commonly adopted since it is standardized by NIST and strongly unforgeable in terms of C and I. It is adopted in network protocols such as IPsec, SSH, and TLS. However, its construction can potentially create expensive overhead, as it requires two runs of high computations [56, 69].

The one-key category uses the same key for encryption and MAC modules. These modules are based on the cryptographic mode of operations applied to existing algorithms [100, 107]. Four modes are in this category, namely, counter with CBC-MAC (CCM) [108], EAX [109], GCM [96], and CWC [110], which are standardized by NIST as methods for block cipher on AE (ISO/IEC 19772:2009) except CWC. SAEB [111], Compact Low overhead Counter Feedback Mode (CLOC) [112], and JAMBU [113] are online, they use only exclusive-OR gate (XOR) operations, and not parallelizable. Various categories of AE schemes are shown in Figure 5. They are classified based on the underlying modules (i.e., C and I provided by separate components), number of secret keys (i.e., can be more than one), and number of data runs (i.e., how many data runs are essential to provide C and authenticity).

5.2. Criteria for Comparing LAE'S. In this section, the criteria involved in the algorithm comparison are specified. They were chosen based on algorithm design properties, functionalities, security vulnerabilities, performance, and resource requirements.

(1) *Design parameters*. There are four parameters of each algorithm including the size of the Message or Plaintext M, Key K, Nonce or Initialization Vector N, AD, and Authentication tag T. When the key size is small, the key space is small and exhaustive key search becomes feasible and easy to perform. If the algorithm key is 128 bits then, key space is approximately 3.4 × 1038 [114]. This large number protect against brute force attack vulnerability, regardless of the algorithm's sophistication [115]

(2) *Online*. The algorithms is online if there is no dependency relation between the ciphertext generated from a block and an earlier input block to the encryption oracle or a post input block to the encryption oracle. Such feature allows the sender to encrypt plaintext blocks before subsequent plaintexts or the plaintext lengths are known. Similarly, the receiver decrypts ciphertext blocks online in the order they were computed during encryption. It reduces the waiting time required to start a computation and enhances the algorithm's efficacy rate [116]

(3) *Parallelizable*. The encryption or decryption of an algorithm is parallelizable if the i^{th} block operates independently of the remaining j^{th} block such that $i \neq j$. The feature enhances design efficiency and improves its throughput. It is indicated as either encryption E being parallel or decryption D or both. Parallelized designs have an advantage particularly when it comes to providing a range of transmission rates [117]

(4) *Intermediate Tag*. The tag that enables the receiver to detect a mismatch of authenticity during the earlier stage after initial block decryption will considerably enhance time-efficiency for discarding un

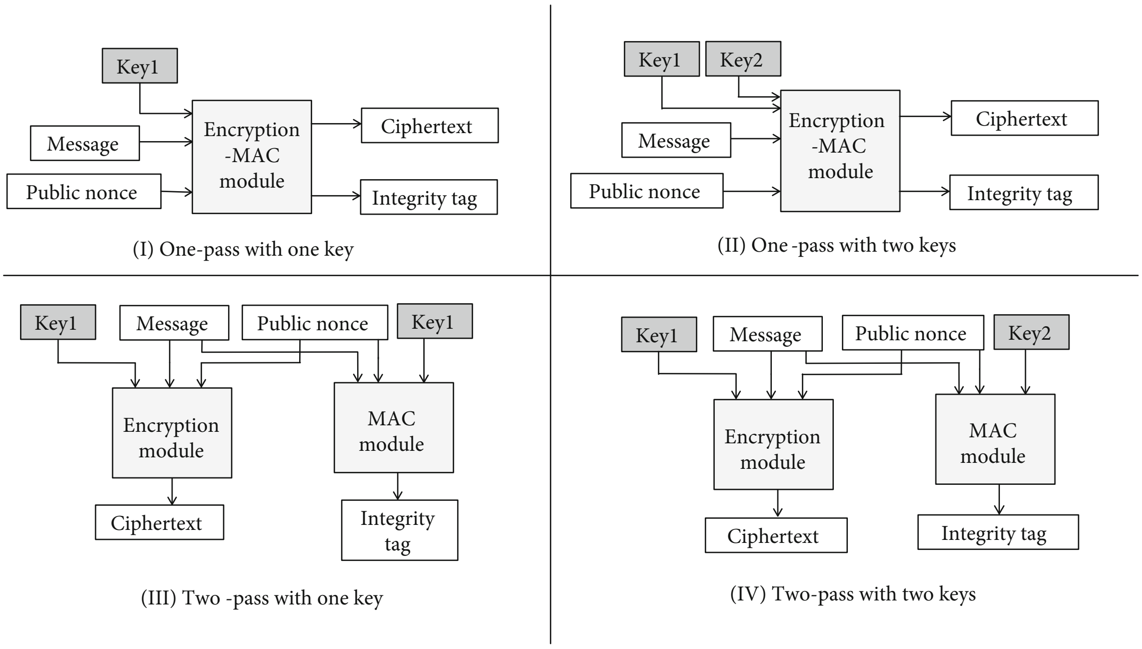


Figure 5: AE schemes modules with interfaces (I) one-pass with one key, (II) one-pass with two keys, (III) two-pass with one key, and (IV) two-pass with two key.

authenticated data without the need to decrypt all the message block [99]

(5) *Inverse-Free*. This feature exists when an inverse operation of the encryption namely a decryption is not necessary to decrypt the ciphertext. In other words, the algorithm uses the same encryption engine for decryption, which significantly determines the amount of dedicated size for the implementation on chips [99]

(6) *Area and Memory*. Number of registers in Read Only Memory (ROM) and Random Access Memory (RAM) are sufficient metrics for memory cost indicating the utilized area of the algorithm where intermediate values are stored. In contrast, lookup tables (LUTs), Slices (consist of LUTs and flip-flop gate), and fundamental figure logic equivalent (GE) give measurements of benchmark size for FPGA and ASIC. However, GEs imprecisely map into platform-specific metrics. For instance the GE for ASIC is equivalent to two-input not-AND (NAND) gates, while the FPGAs LUT is equivalent to six two-input NAND gates. The issue of mapping gap has been discussed in the literature [117, 118]

(7) *Power Consumption*. Power consumed plays a key role in performance measurement for restricted devices since battery shortage in IoT devices is still challenging [42]. A security algorithm with low computation consumes less power, hence, the battery discharges slowly. On the other hand, energy consumption signifies power being consumed on a periodic basis, which increases proportionally to the power. Therefore, power dissipation in a lightweight device ensures its capability for sufficiently operating the algorithm

(8) *Throughput*. This specifies the number of processing bits per second (bps) of input data within the algorithm. Higher throughput while preserving power consumption and utilized area is still questionable in IoT devices [119]. The acceptance of the throughput against area-power trade-off is based on the use case and the device restrictions

(9) *Code Size*. This denotes the code storage memory implied on ROM. Restricted devices limit the memory storage and any access between ROM and RAM while being executed [120]. Hence, the high access between memories lead to significantly increased device overheads

(10) *Execution Time*. This is indicated using the waiting time from the start process of the encryption and until the authentication tag is output. There is a correlation between AE design type and execution time. For example, a two-pass scheme doubles the execution time of a one-pass scheme using the EtM approach. As a result, execution time is affected by the algorithm design, such that some two-pass schemes require more computations and dependencies

(11) *Algorithm Size and IoT Device Resource*. The algorithm's computation required adequate metrics from the perspective of available resources. Such figures correlated to how small is the algorithm to fit into devices, which indicated in terms of utilized

area, power, code size, and execution time with reasonable throughput rate and number of cycles [60]. A small amount of resources left for security algorithms is evident in many IoT devices. Despite security protocols inherits in network layer for example TCP/DTLS1.2 within CoAP or Internet Protocol security IPsec encapsulated as MQTT, they are inadequate for IoT constrained storage (i.e., Radio Frequency Identification, Wireless Sensor Network, Microcontroller, and Near Field Communication), low power devices with limited battery capacity (i.e., smart cards and sensors) and computing power [121, 122]. Therefore, the range of encryption and authenticity algorithms supplied for an application are limited in view of device capability [123, 124]

6. Algorithms in the Review

In this section, we introduce the reviewed algorithms, which are grouped into block cipher-based, stream cipher-based, permutation-based and encrypt, and MAC algorithm. Details of the algorithm and their functionality features are discussed briefly.

6.1. Block Cipher-Based Lightweight Authenticated Encryption (BCAE). AES-GCM is an AEAD in GCM. It is a current NIST standard for AE which is an inherit feature of TLS [125], SSH [126] and IPsec [127]. Algorithm C is achieved using an incremental counter based (CTR). This incremental counter is encrypted, then XORed with the block message. The size of initial IV for the message is 96 bits, which is padded with an additional 31 bits of zero values. Despite being practically deployed, it does not withstand against nonce misuse and other attacks. To overcome this weakness, a modified version defined as AES-GCM-SIV [128], although never been considered before for IoT, is a proper recommendation when resources are limited.

AES-CLOC is an AES-128-based AEAD that facilitates CLOC. It takes 128 bits key, 96 bits nonce, 240 bits plaintext, 112 bits AD, and 128 bits tag to generate 240 bits ciphertext. Specifically, CLOC was designed to overcome NIST methods for AE, which were only combined with block cipher primitive. Additionally, it reduces the precomputation overhead, which improve the performance for short input data compared to the standardized modes, where less calls are required for the encryption/MAC engine. For example, if a message consists of a one block nonce, one block AD input data, and one block plaintext, CLOC requires 4 calls to the cipher engine, instead of 7 calls in EAX and 5 or 6 in CCM.

JOLTIK is a LAE scheme based on AES block cipher designed especially for hardware applications. It is tweakable to 64 bits cipher, and distinct keys are suggested with four sets of parameters to achieve a 64 bits security for C and authenticity. The sets' sizes were formatted as <key size-tag size> to be 64-64, 80-112, 96-96, and 128-64. The algorithm is composed of two parts, these are encryption and verification where the encryption takes a variable length message, variable AD, fixed length nonce, and a K bits key to generate m bits cipher and an authentication tag. All sets have been mathematically proved to provide security even when a nonce is reused. As the scheme is designed for hardware, it is efficient in hardware platforms and utilizes a small area.

SCREAM is a tweakable block cipher that work on 128 bits data processed using 128 bits key with AD. SCREAM is simple in design and encourages good performance on different architecture. The designers introduced a masking countermeasure against side channel attacks. The security level achieved by design is 128 bits. Low overheads are another feature so that no additional cipher calls were used for the masking generation against side channel attack. Besides, compact design allows a fully parallelization with minimum ciphertext size. There are two recommended parameters for SCREAM that are computationally secure with the provided security level, either 10 steps with single key security, or 12 steps with related key security.

SILC-AES is an authenticated block cipher based with two versions SILC-AES and SILC-LED. SILC-AES encrypts 64 or 128 bits message with 128 bits key length, 96 bit or 64 bits nonce, and 64 I tag. With provable security, it is optimized and evaluated on standard hardware, and generated a small footprint with low transmission rate suitable for low-rate applications especially when throughput is not essential. Similarly, SILC-LED, which operates on 64 bits message, 80 bits key, 48 bits nonce, and 32 bits I tag is recommended for a low data transmission and limited battery device.

AEGIS is an AEAD that is suitable for protecting protocol packets. AEGIS-128 is constructed as five AES rounds to process 128 bits message, while AES-256 is constructed as six AES rounds. Both versions are argued by the designer to function at a high security level but this has not been proven, and besides they are not robustness against nonce reuse.

COLM is a block cipher based on AEAD features. It is designed to achieve online misuse resistance and fully parallelizable. COLM uses an AES-128 with a key and state of 128 bits length. C and I of the design achieve a security level of 64 bits even when the nonce repeated, which withstand against different attacks.

Deoxys-v1.41operates using a key length of either 128 bits or 256 bits to construct two modes: DeoxysI and DeoxysII. DeoxysI requires nonce respecting where nonce never repeat, while DeoxysII resists nonce misuse and nonce can repeat without affecting the algorithm's security. The tag size is 128 bits and a 64 bits for the first version, while it is 120 bits for the second version. However, security degrades for C and authenticity with respect to nonce repetitions on the second version. Deoxys perform well with reference to software implementation.

AEZ is an AEAD block cipher mode of operation. It is constructed as two versions with respect to the number of rounds; these are 4 and 10 rounds. The designers argue it is the easiest to use, however it is complex in implementation. The difficulty comes from the mode complexity of connecting two unrelated mechanisms. The scheme security has been attacked using side channel attacks as analyzed in [129].

AES-JAMBU is an AE based on AES-128 block cipher that operates on 128 bits key constructed on the JAMBU mode of operation. The designers compared JAMBU to other existing modes of operation and claimed that its resistance to nonce misuse is similar to CFB mode, which maintains a strong level of security. In contrast, JAMBU is the most lightness mode in terms of computation and provides bijective n-bit authentication security to the 2n bits block size.

Tiaoxin is a family of nonce resistance scheme and it operates on 128 bits message, 128 bits key, 128 bits nonce, AD of 128 bits, and a 128 bits tag. The security level of is claimed to be 128 bits for C and authenticity. However, it is not resistant with respect to nonce repeating and can be used to recover state bytes and to compromise the C. Besides, if an adversary knows a key, he can easily generate a tag collision.

6.2. Stream Cipher-Based Lightweight Authenticated Encryption (SCAE). ACORN is constructed based on stream cipher that processes bit-by-bit; however, it processes 32 steps in parallel, which makes it faster in hardware and software platforms [130]. The encryption and authentication shares 293 bits state encrypted via 128 bits key, 128 bits nonce to generate a 128 bits tag. The authentication deploys a concatenation of six linear feedback shift registers. The major characteristics of ACORN v1, v2, and v3 are online, parallelizable, inverse-free, and robust. The first two versions are mathematically are proven to be insecure. The work in [131] revealed that ACORN_v1 is not resistant to state collision, where two distinct input messages produce the same ciphertext. The [131] attack was deployed on a standard PC. The fault attack in [132] fully recovers two initial states of ACORN_v2 with time complexity, then establishes key recovery and forgery attack. Thus, ACORN_v2 is insecure because of the nonlinear feedback function that has been replaced by the filtering function in ACORN_v3 and provides a larger security margin against guess-and-determine attacks. A side-channel attack has been established on ACORN_v2 by [133] to recover the full key, which is then addressed by an additional masking countermeasure. It is practically tested on an ARM processor and proved its resistance to [133] attack.

The MORUS family has two internal sizes, 640 bits and 1280 bits, and two different key sizes, 128 bits and 256 bits that construct three recommended parameters, MORUS-640-128, MORUS-1280-128, and MORUS-1280-256. The tag size reaches a maximum of 128 bits and can be shorter. However, the designers recommend using a 128 bits tag so that the I will be 128 bits, and the C reaches the number of key bits. The three parameter sets, however, are not resistant to nonce misuse and the security withstands it only if a reused nonce is encrypted with a changing key. However, this algorithm has not been mathematically proven against the designers' claimed security strength.

6.3. Permutation-Based Lightweight Authenticated Encryption (PBAE). ASCON_v1.2 is constructed as a sponge-based scheme operating on a variable length input to produce a fixed length output. The operation state of 320 bits includes 128 bit key, 128 bit nonce, 128 bit tag, 64 bit data block, and 12 and 6 rounds for two intermediate permutations. It is been the winner of CASEAR competition and a candidate for ongoing NIST standardization. There is much research community interest in the efficiency metric for various platforms, given that ASCON is oriented for hardware platform. These include unprotected versions against differential power attacks as reported in [134], and protected versions in [130, 135].

PRIMATEs-80 and PRIMATEs-120 are two variants of AEAD and consist of three modes of operation namely APE, HANUMAN, and GIBBON. It takes an input a key generated by key generator, AD, message, and nonce to generate a ciphertext and I tag. All algorithms can resist attacks with security levels of 80bits and 120bits, which has been mathematically proved by designers. PRMATE permutations are defined by different round constants, which are generated by 5 bit LFSR and various round numbers.

NORX is a family of authenticated ciphers with scalable architectures so that the extent of parallelism and tag size are arbitrary. NORX operates on 128 bits or 256 bits with 128 bits or 256 bits key, where the same security level holds. However, it cannot resist nonce reuse although it performs well on both software and hardware platforms.

Ketje is a family of the AEAD scheme based on sponge. It takes as input a secret key and a nonce, then some AD that are authenticated but not encrypted and a plaintext. A ciphertext and authenticating tag are produced to ensure the data and the packet headers security. The design aimed to serve memory constrained devices and assume the nonce is unique for each communication to achieve security. Thus, it is unsecure to implement if an adversary repeats the nonce and recovers the data.

The Keyak family is another version of the Ketje with similar intermediate operations. The similarity is that it takes a unique secret input but with its AD that are not encrypted and a plaintext. It then produces a ciphertext and a tag that authenticates both the AD and the plaintext. The recipient party who has the same-shared secret key can decrypt and authenticate the ciphertext. The designers claim it resists to nonce misuse if an adversary reuses the nonce that cannot retrieve the key or any internal state.

6.4. Encrypt and MAC Scheme. OCB is an authenticated mode of operation that is fully parallelizable. This mode is standardized by NIST for lightweight methods based on block cipher. Due to OCB characteristics to encrypt an arbitrary message length into a ciphertext with minimal length, this means using cheap offset computations and key setup. It also resists nonce misuse. These features are suitable for restricted resources devices and low cost IoT applications.

7. LAE Comparisons and Discussions

We present a comparison of various LAE using criteria noted in Section 6 part B criteria. These features affect an algorithm's design efficacy, security, and resource measures. Algorithms were eliminated from the review due to vulnerability threats or insufficient studies done on them. For example, AES-COPA and ElmD are mathematically proven to be insecure in [35] and [36], respectively. SliScp [136] has not received sufficient attention in the literature, although it was published for a while. WAGE [137], ACE [138], and Elephent [139] were recently designed and excluded due to insufficient analysis on them. COMET [140] and MixFeed have been threatened by weak keys as demonstrated in [141].

7.1. Security Vulnerabilities Comparison. Table 3 compares algorithms based on their security strength. The security strength is evaluated based on the availability of mathematical proof of their C and I, nonce misuse robustness, and side channel attacks. Algorithms, which are proven to maintain message C and I, are resistant to nonce misuse and resistant to side channel attacks beside other security attacks, are desirable. If IoT nodes are captured and altered to repeat the nonces the data will be revealed. Many protocols are compromised due to nonce misuse such as Wi-Fi Protected Access 2 (WPA) and Wired Equivalent Privacy (WEP). Thus, in terms of security, algorithms, which can resist nonce misuse are highly desirable.

The study shows that AES-JAMBU, Tiaoxin, AEGIS, and MORUS_v2 are not mathematically proven to be secure against C and I attacks. An interesting observation is that ASCON_v1.2, JOLTIK, PRIMATEs, COLM, DeoxysII, OCB, and Keyak maintain their message C and authenticity. However, ASCON_v1.2 is not robustness against nonce misuse because nonce repetition for two varying messages can reveal their differences. Furthermore, excess reuse of the nonces releases the algorithm internal state, which can be identified using structural attacks.

From the perspective of nonce resistance, whereby the nonce can be modified or altered without affecting the security of the algorithm, JOLTICK, PRIMATESs, COLM, DeoxysII, OCB, and AES-JAMBU are secure. Selecting these is more powerful in terms of security for uncontrolled environment. However, the nonce should be unique for every encryption query to prevent message leakage in ACORN_v3 ASCON_v1.2 AES-CLOC, SCREAM, SILC-AES, SILC-LED, AEGIS, NORX, AEZ, Ketje_v2, and Keyak. These schemes fail to deliver the C when the adversary manipulates the nonce. Thus, they are suitable for controlled environments where nonce randomness is preserved to ensure data C and authenticity.

JOLTIK has two versions, and one of them is nonce-misuse resistance. In this version, a birthday bound security is proven for reused nonce for 64 bits for plaintext C, I of plaintext, I of nonce, and I of AD. This provides the 64bits security with only one query to the block cipher. Similarly, OCB3, AES-GCM, and PRIMATEs in APE mode are proven for reused nonce. As far as we know, AES-GCM, ACORN_v3, AES-CLOC, Tiaoxin, and AEZ have been studied and found to be insecure against fault attacks. A key problem of these algorithms is that fault attack recovers the key and reveals the message information. Hence, they are not recommended for IoT devices because they can be cloned when they are placed in untrusted environment.

COLM and DeoxysII have the same 64 bits size for birthday bound security. COLM security bound is supported with ELmD and COPA security proofs. In contrast, AES-JAMBU was analyzed with repeated nonces for two identical messages, and the first two blocks can be revealed. Although, both ELmD and COPA are threatened by forgery attacks based on tag guessing, such that the attack does not violate their birthday bound security.

7.2. Functionality Comparisons. Table 4 summarizes the algorithms' functionalities in bits, namely, message block size, key length, nonce length, tag size, online, parallelism, required inverse for decryption, intermediate tags, and if its support AEAD. DeoxysI, DeoxysII, NORX, AEZ, and Keyak take variable plaintext length and variable AD. Unlike others, these algorithms have the potential to be deployed on various protocols with different data communication size. The IPv6 and IEEE 802.15.4, for instance, differ on the payload size that has to be protected and the associated header length.

More concretely, the frame size in IEEE802.15.4 is 127 bytes such that the maximum header length is 25 bytes, which leaves 102 bytes for the payload. It supports a scenario for AE where only 86 bytes of payload encrypted, 25 bytes for header authenticity, and 16 bytes reserved for the integrity tag [162]. In contrast, IPv6 frames defined in RFC 2640 use AE to support the encryption of 1224 bytes payload and 40 bytes of AD.

It is observed that AES-GCM, JOLTIK, DeoxysI, DeoxysII, and OCB support the need for encryption and decryption engines for the processing, in contrast to other algorithms, which do not. This functionality affects the resources required for implementation so that algorithms can have one engine for encryption and decryption or two separate oracle engines. Despite the shortcomings of the need to inverse algorithms, JOLTIK, DeoxysI, DeoxysII, and OCB encryption and decryption engines are parallelable.

7.3. Performance Comparisons. Table 5 compares the algorithms' features with corresponding hardware performance metrics. In the literature there seems to be some confusion when comparing algorithms' implementation. To overcome this problem, we explain two factors when reviewing the performance of the algorithms.

(1) *Platform-Awareness*. The variation of the benchmarked platform is the basis for comparison. Different technologies have their own device mappings, leading to technology-specific performance. For example, the metrics of implementing SILC in ASIC are differentiated from FPGA. Building units based on the FPGA approach are mostly done on LUTs rather than logic gates, the most fundamental hardware metric, while the ASIC area is measured in mm^2. An algorithm built on the same technology is analyzed for a specific platform type [163]

(2) *Resources Limitation*. Targeting restricted devices, small area, low power consumption, and throughput have to be considered. For example, the approach, which achieves low power requires shortage battery, withstands longer until a battery replacement is essential, simultaneously the algorithm footprint has to be minimum

Table 3: Authenticated encryption algorithms security comparison.

Algorithm	Provable confidentiality	Provable authenticity	Nonce misuse resistance	Side channel attacks
AES-GCM [142]	√	×	×	Cache attack [143], Fault attack [70]
ACORN_v3 [144]	√	√	×	Fault attack [132]
ASCON_v1.2 [145]	√	√	×	×
AES-CLOC [112]	√	√	×	Fault attack [146]
JOLTIK [147]	√	√	√	×
PRIMATEs [148]	√	√	√	×
SCREAM [149]	√	√	×	√
SILC-AES [150]	√	√	×	×
SILC-LED [150]	√	√	×	×
AEGIS [151]	×	×	×	×
COLM [152]	√	√	√	×
DeoxysI [153]	√	√	×	×
DeoxysII [153]	√	√	√	×
OCB [102]	√	√	√	×
NORX [154]	√	√	×	×
AES-JAMBU [155]	×	×	√	×
Tiaoxin [156]	×	×	×	Fault attack [157]
AEZ [158]	√	√	×	Fault attack [129]
Ketje_v2 [159]	√	√	×	×
Keyak [160]	√	√	×	×
MORUS_v2 [161]	×	×	×	×

We observe that BCAE algorithms receive much more interest on performance testing when targeting Field Programmable Gate Array (FPGA) and Application Specific Integrated Circuit (AISC). Similarly, rather than ASIC, FPGAs were mostly targeted by performance developers. The hardware built as ASIC technology is relatively more expensive than the FPGA platform and ASIC usually is a production platform while FPGA serves as a validation platform. On the other hand, BCAE algorithms are easy to design and deploy compared to permutation-based and stream cipher-based algorithms.

From the metric perspective, area, power, and throughput are reported for various performance levels while TimexArea and its efficiency are not considered. The area computation is mapped differently based on the platform so that LUTs represented FPGA while mm^2 stands for ASIC. Such differences ensure area fraction within a device is differentiated from other platforms because the selected device featured the accessible resources. Furthermore, a wide range of algorithms with several schemes are investigated, yet tested platforms vary and consequently so do the available resources. Benchmarking the algorithm potentially is affected by testing platform, hardware or software design architecture, algorithm arithmetic, algorithm characteristics (i.e., parallelizable, inverse-free, online), and attacks countermeasure (i.e., side channel masking). Thus, a comparison should be aware of the differences between various platforms.

Table 6 compares algorithms for software performance in 8 bit AVR, 16 bit MSP, 32 bit ARM, and amd-64 bit. It includes the family, mode of operation, speed for encryption E, and decryption D in cycles/bytes, ROM, and RAM in bytes, cycle count in cycles, type of implementation and platform, and the name of the protocol if it is validated to work within the protocol.

The ROM and RAM memories besides cycle count were not reported in many of implementations, while the speed of the algorithm was done to assess software performance. A key reason is the capabilities that software devices have where 8bits platforms are of less interest for practical applications. We observe from Table 6 that speed computation varies, where some developers indicate their implementation include encryption only engines some include encryption and decryption, others have not indicate the number of engines. This variation leads to unfair conclusions that an algorithm is performing well by implementing the encryption engine only. The encryption processing time is also affected and can be doubled for a full encryption and decryption engines. For practical IoT, where the algorithm is implemented in IEEE 802.5.4 or IPv6, a few algorithms, namely AES_GCM, ACORN_v2, ASCON_v1.2, NORX, and Ketje are validated on both protocols. Other algorithms are not validated for the maximum payload and I tag length and nonetheless supported by these protocols.

7.4. BCAE Performance Comparison. AES-CLOC, SCREAM, AES-JAMBU, AES-SILC, Aegis, AEZ, JOLTIK, Tiaoxin, COLM, DeoxysII, and LED-SILC were benchmarked on ASIC and FPGA with performance shown in Figure 6 and Figure 7, respectively. Based on area utilization, AES-CLOC [20], Scream-10 [149], and AES-JAMBU [48] are

Table 4: Characteristics of authenticated encryption candidates, v means variable.

Algorithm	Message	Key	Nonce	Tag	Online	Parallel (E/D)	Inverse free	Intermediate tag	AEAD
AES-GCM	128	128	96	128	√	√/×	×	×	√
ACORN	128	128	128	128	√	√/√	√	×	√
ASCON	128	128	128	128	√	×/×	√	×	√
CLOC	128	128	96	64	√	×/×	√	×	√
JOLTIK	128	128	32	64	√	√/√	×	×	√
PRIMATEs	80	80	80	80	√	×/×	√	×	√
	120	120	120	120	√	×/×	√	×	√
SCREAM	128	128	96	128	√	√/√	√	×	√
SILC-AES	128	128	64 96 112	128	√	×/√	√	×	√
Aegis	128	128	128	128	√	√/×	√	×	√
	256	256	256	128	√	√/×	√	×	√
COLM	128	128	64	128	√	√/√	√	√	√
DeoxysI	V	128	64	128	√	√/√	×	×	√
DeoxysII	V	128	120	128	√	√/√	×	×	√
OCB	128	128	128	64	√	√/√	×	×	√
NORX	V	96	32	96	√	√/√	√	×	√
SILC-LED	64	80	48	32	√	×/√	√	×	√
AES-JAMBU	128	128	64	64	√	×/×	√	×	√
Tiaoxin	128	128	128	128	√	√/√	√	×	√
AEZ	V	384	128	128	×	√/√	√	×	√
Ketje	128	128	128	128	√	×/×	√	√	√
Keyak	V	288	V	128	√	√/×	√	√	√
MORUS	128	128	128	128	√	×/×	√	×	√

the lightest in ASIC while AES-JAMBU, AES-SILC, and LED-SILC in [164] utilized the smallest footprints in FPGA. AES-SILC in [25] consumed as little as 5.98 mWatt utilizing 3004 LUTs, while in [164] dissipated higher power of 9200 mWatt and substantiality less area of 1160 LUTs were computed. The major variations in the benchmarks are the platform type of FPGA where Spartan-6 dissipated an extremely high amount of power to process the ciphers.

Due to design simplicity and a few interfaces of AES-CLOC and Scream-10 in ASIC, they achieve higher throughput compared to the other of 6840 and 4577 Mbps, respectively. AES-JAMBU [48] reported 0.058 mm^2 and 3.39 mWatt, which is enormously less than consumption of AES-CLOC, which was reported as 18.79 mWatt. Scream [25] and JOLTIK [25] utilized 0.114 mm^2, 0.842 mWatt and 0.178 mm^2 and 0.96 mWatt, for area and power, respectively. For throughput, in contrast Scream approached a higher bandwidth of 128 Mb per seconds while JOLTIK sent 20 Mbps. Thus, Scream can be used for high bandwidth applications like Wi-Fi-based wireless sensor network, whereas JOLTIK is recommended for low throughput applications like smart Bluetooth [165].

AES-GCM dissipated 1666 mWatt, however in terms of area, GCM mapped into smaller FPGA LUTs. AES in GCM is a two-pass scheme-facilitating headers AD authentication, which is inherent in IPsec and TLS. It is, however, vulnerable to forgery attacks on the I tag [166], where certain cyclic keys can be repeated. Using AES-GCM-SIV [128], instead of AES-GCM, accelerates the instructions and is predicated to perform faster.

Aegis [60], AES-CLOC [20], and AES-SILC [60] were the highest throughput with 8650 Mbps, 6840 Mbps, and 6400 Mbps, respectively, whereas DexoysII [60], JOLTIK [25], and COLM [60] had 18.63 Mbps, 20 Mbps, and 23.75 Mbps, respectively, as the lowest throughputs in Figure 5(b). Of these, AES-SILC and Aegis consumed 4.36 and 7.52 mWatt while maintaining a low area of 0.0677 mm^2 and 0.1661 mm^2. In contrast, DexoysII and COLM consumed significantly less, i.e., 0.0988 and 0.0177 mWatt, respectively, but inefficient area footprint being 531.91 mm^2 and 505.05 mm^2, also, respectively.

AES-CLOC and SCREAM tested on 8bit AVR as illustrated in Figure 8. Based on the resources, ROM memory assessments AES-CLOC [112] used less bytes compared to SCREAM-10 [149], i.e., 2980 bytes and 3221 bytes, respectively. In contrast, the bytes utilized as RAM memory number much less for SCREAM with 80 bytes measured while AES-CLOC required 362 bytes. Variations come from the experiment measurements so that CLOC was measured for encryption and decryption engines while SCREAM was not.

The algorithms' performance in 64 bit ARCH were measured in terms of the speed, meaning that Aegis, AEZ and AES-SILC are the fastest algorithms and required only

Table 5: Authenticated encryption hardware performance metrics.

Algorithm	Family	Mode	Area	Power (mWatt)	Throughput (mbps)	Implementation
AES_GCM [167]	BCAE	GCM	9167 LUTs	1666	×	Zynq-7000 PYNQ
ACORN_v2 [25]	SCAE	×	0.035 mm^2	0.163	8	CMOS UMC 130
ACORN32 [60]	SCAE	×	0.0169 mm^2	3.130	34040	TSMC 65 nm
ACORN_v2 [25]	SCAE	×	476 LUTs	0.582	8	Zynq-7000 XC7Z020
ACORN32 [167]	SCAE	×	7342 LUTs	1646	×	Zynq-7000 PYNQ
ACORN [164]	SCAE	×	418 LUTs	9200	1225.5	Spartan-6
ASCON GMU [60]	PBAE	Monkey duplex	1408.4 mm^2	0.0235	3.310	TSMC 65 nm
ASCON [25]	PBAE	Monkey duplex	0.083 mm^2	0.655	106.67	CMOS UMC 130
ASCON [25]	PBAE	Monkey duplex	1312 LUTs	2.160	106.67	Zynq-7000 XC7Z020
ASCON [167]	PBAE	Monkey duplex	7726 LUTs	1648	×	Zynq-7000 PYNQ
ASCON [164]	PBAE	Monkey duplex	684 LUTs	×	60.1	Spartan-6
AES-CLOC [112]	BCAE	CLOC	5628 LE	×	400.7	FPGA cyclone IV
AES-CLOC [25]	BCAE	CLOC	0.544 mm^2	2.858	128	CMOS UMC 130
AES-CLOC [25]	BCAE	CLOC	2767 LUTs	3.766	128	Zynq-7000 XC7Z020
AES-CLOC-GMU [20]	BCAE	CLOC	0.0140 mm^2	18.79	6840	TSMC 65 nm
JOLTIK [25]	BCAE	TAE	1325 LUTs	1.380	20	Zynq-7000 XC7Z020
JOLTIK [25]	BCAE	TAE	0.178 mm^2	0.96	20	CMOS UMC 130
PRIMATEs [25]	PBAE	GIBBON	1187 LUTs	3.547	66.67	Zynq-7000 XC7Z020
PRIMATEs [25]	PBAE	GIBBON	0.106 mm^2	1.064	66.67	CMOS UMC 130
Scream-10 [149]	BCAE	TAE	17292 um^2	×	4577	65 NM CMOS
SCREAM [25]	BCAE	TAE	0.114 mm^2	0.842	128	CMOS UMC 130
Scream [25]	BCAE	TAE	2235 LUTs	4.106	128	Zynq7000 XC7Z020
AES-SILC [150]	BCAE	SILC	15675.5 GE	×	764.12	90 nm ASIC
AES-SILC [25]	BCAE	SILC	0.187 mm^2	2.345	128	CMOS UMC 130
AES-SILC [48]	BCAE	SILC	0.1031 mm^2	7.000	640	CMOS UMC 130
SILC-GMU[60]	BCAE	SILC	0.0677 mm^2	4.360	6400	TSMC 65 nm
AES-SILC [25]	BCAE	SILC	3004 mm^2	5.980	128	Zynq7000 XC7Z020
AES-SILC [164]	BCAE	SILC	1052 LUTs	9200	76.6	Spartan-6
AES-SILC [164]	BCAE	SILC	1198 LUTs	×	48.1	Zynq7000XC7VX485T
AES-SILC [164]	BCAE	SILC	1160 LUTs	×	59.13	Zynq-7000 XC6VLX760
LED-SILC [164]	BCAE	SILC	872 LUTs	8400	15.1	Spartan-6
AES-CLOC [164]	BCAE	CLOC	1604 LUTs	1089	68.7	Spartan-6
AES-CLOC [164]	BCAE	CLOC	1306 LUTs	×	45.72	Zynq-7000 XC7VX485T
AES-CLOC [164]	BCAE	CLOC	1282 LUTs	×	52.03	Zynq7000XC6VLX760
Aegis_128 [167]	BCAE	×	17323 LUTs	2139	×	Zynq7000 PYNQ
Aegis_265[167]	BCAE	×	19716 LUTs	2039	×	Zynq7000 PYNQ
Aegis-GMU [60]	BCAE	×	0.1661 mm^2	7.520	8650	TSMC 65 nm
DeoxysII [167]	BCAE	XEX	10681 LUTs	1738	×	Zynq7000 PYNQ
DeoxysII [60]	BCAE	XEX	531.91 mm^2	0.0988	18.63	TSMC 65 nm
DeoxysII	BCAE	TAE	14107 GE	×	×	×
AES-OCB [167]	BCAE	OCB	10432 LUTs	1683	×	Zynq7000 PYNQ
OCB-GMU [60]	BCAE	OCB	0.1442 mm^2	27.42	4920	TSMC 65 nm
NORX [60]	PBAE	Monkey duplex	0.1231 mm^2	19.51	57400	TSMC 65 nm
NORX [48]	PBAE	Monkey duplex	0.1039 mm^2	4.370	2400	CMOS UMC 130
NORX [164]	PBAE	Monkey duplex	1424 LUTs	1280	2989.0	Spartan-6
AES-JAMBU [164]	BCAE	JAMBU	191 LUTs	737	×	Zynq7000XC7VX485T
AES-JAMBU [164]	BCAE	JAMBU	244 LUTs	713	×	Zynq7000XC6VLX760
AES-JAMBU [60]	BCAE	JAMBU	0.3887 mm^2	3.110	3170	TSMC 65 nm

Table 5: Continued.

Algorithm	Family	Mode	Area	Power (mWatt)	Throughput (mbps)	Implementation
AES-JAMBU [48]	BCAE	JAMBU	0.0580 mm^2	3.390	128	CMOS UMC 130
Tiaoxin [60]	BCAE	×	0.0140 mm^2	9.360	1115320	TSMC 65 nm
Tiaoxin [48]	BCAE	×	0.2282 mm^2	11.68	4270	CMOS UMC 130
AEZ-GMU	BCAE	XEX	0.1186 mm^2	22.07	2980	TSMC 65 nm
Ketje_jr [60]	PBAE	Monkey wrap	0.0172 mm^2	3.270	14550	TSMC 65 nm
Ketje_sr [60]	PBAE	Monkey wrap	0.0276 mm^2	4.710	29090	TSMC 65 nm
MORUS [25]	SCAE	×	0.27 mm^2	2.830	256	CMOS UMC 130
MORUS [60]	SCAE	×	50965 um^2	×	114.8 Gbps	TSMC 65 nm
MORUS [161]	SCAE	×	179 slices 4122 LUTs	×	94117	FPGA Vertix-7
MORUS [25]	SCAE	×	4286 LUTs	4.899	256	Zynq7000 XC7Z020
COLM [60]	BCAE	Encrypt-linear Mix-encrypt mode	505.05 mm^2	0.0177	23.75	TSMC 65 nm
COLM [48]	BCAE	Encrypt-linear Mix-encrypt mode	0.3274 mm^2	12.08	580	CMOS UMC 130
COLM [164]	BCAE	Encrypt-linear Mix-encrypt mode	2521 LUTs	×	37.1	Zynq7000XC7VX485T
COLM [164]	BCAE	Encrypt-linear Mix-encrypt mode	2511 LUTs	×	38.9	Zynq7000XC6VLX760
COLM [167]	BCAE	Encrypt-linear Mix-encrypt mode	13861 LUTs	1796	×	Zynq7000 PYNQ

2.15 cycle/bytes, 4.57 cycle/bytes and 4.9 cycle/bytes, respectively, to process a message while operating on different modes. Conversely, JOLTIK, consumed an extremely large number of cycles of 1590.87 cycle/bytes which affected memory utilization.

Taking side channel attacks resistance into account, SCREAM design integrated a masking countermeasure that protects against manipulation and faults injection to recover the encryption key. This is recommended for 8 bits and 64 bits. Dexoys, however, is vulnerable to nonce-misuse and deemed to be unsecure algorithm in untrusted environment IoT.

7.5. SCAE Comparisons. ACORN_v2 [25, 60] and MORUS [25, 60] are stream cipher-based authenticated algorithms benchmarked on ASIC as illustrated in Figure 9. ACORN is the most area and power efficient algorithm, which is designed for resource-constrained environments and it incorporates three functions: the keystream generator, feedback bits function and state update function. However, these three functions are based on two Boolean functions of basic AND and XOR gates and are faster on FPGA and ASIC.

ACORN performance on ASIC reported by [25] is effective in terms of area and power metrics utilizing 0.035 mm^2 and 0.9 mWatt compared to [60] being 0.0169 mm^2 and 3.13 mWatt. The reason for the performance variations is the design that target efficient throughput in [60] as it was extremely higher than [25] who reported 34040 Mbps compared to only 8 Mbps. However, targeting FPGA PYNQ [167], it was the worst in terms of battery dissipation since it consumed 8200 mWatt, which was extremely higher than Spartan6 [164] and Zynq-7000 [25]. For this reason, it is not recommended for small IoT application with limited battery like medical devices that are attached to the human body.

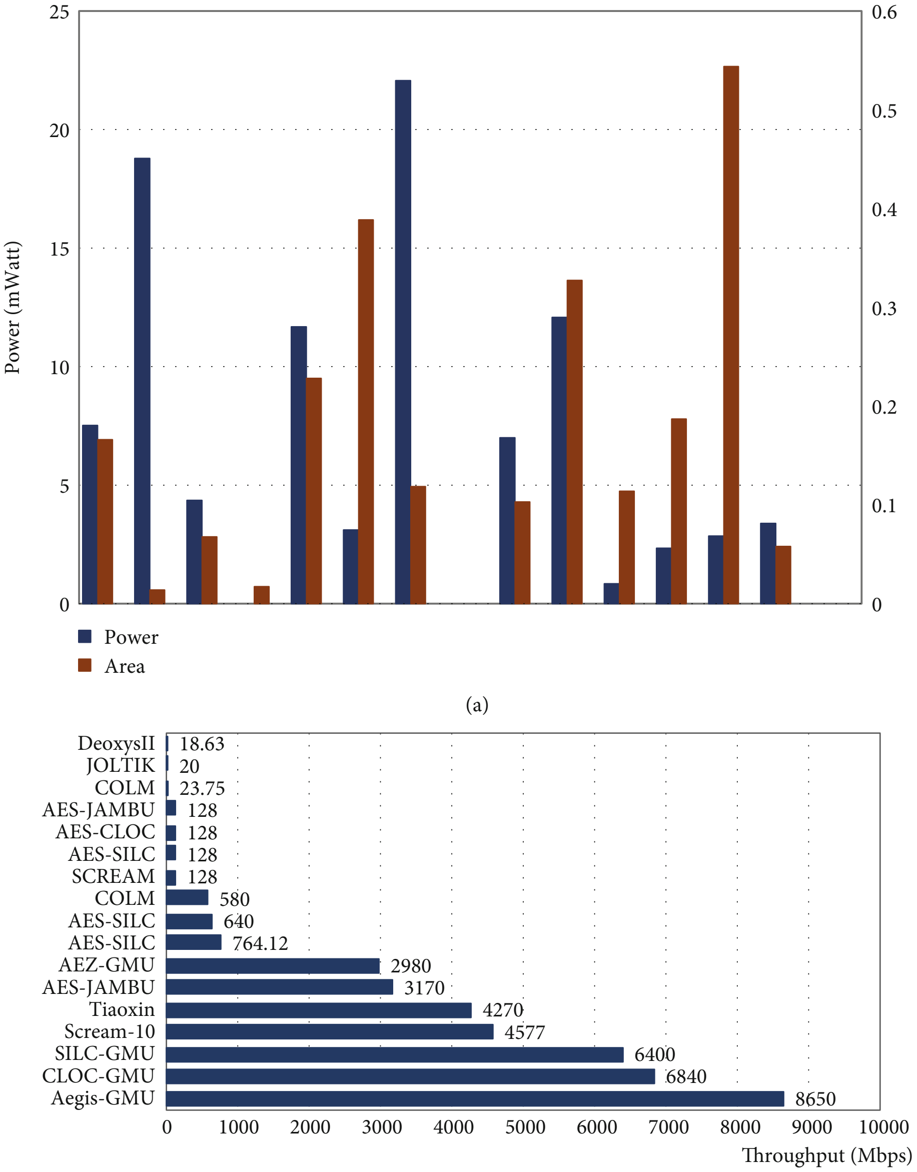
MORUS on the other hand, utilized a smaller ASIC footprint of 0509 mm^2 [60] compared to 0.27 mm^2 [25], while higher power ranging from 35.39 mWatt [60] to 2.83 mWatt [25] was consumed. For FPGA, it utilized approximately 4122 LUTs on Zynq-7000 [25] and 4286 LUTs on Virtex [161] while transmission rate reached 256 Mbps [25] compared to ACORN which transmitted 8 Mbps [25]. Hence, MORUS achieved higher throughput performance compared to ACORN.

ACORN performance was measured on 8 bits AVR, 16 bits MSP, and 32 bits ARM. The same RAM memory of 184 bytes required for the algorithms, yet the dissipation varied. As Figure 10 shows, 32 bits ARM employed the smallest number of 267168 cycles to encrypt a message; 8 bits AVR required 464381 cycles and 16 bits ARM needed 626192 cycles. Such metrics were affected by the ACORN number of rounds to generate a cipher keystream and the construction of a nonlinear feedback function.

Tiaoxin and MORUS reported the higher speed of 3.53 cycles/bytes and 4.87 cycles/bytes, respectively. ACORN of v2 and v3 processed a message as relatively the same speed such that v3 was faster by 0.31 cycles/bytes. Although Tiaoxin has been mathematically proved for its nonresistance feature against fault attack, it is recommended to add a proper masking protection layer to resist against side channel attack. MORUS on the other hand can be broken using nonce-repetition and should not be implemented unless nonce resistance is ensured.

Table 6: Authenticated encryption software performance metrics.

Algorithm	Family	Mode	Speed E/D (cycles/bytes)	ROM (bytes)	RAM (bytes)	Cycle count (cycles)	Implementation	Protocol
AES_GCM [167]	BCAE	GCM	E/D: ×	×	367	975184	8 bits AVR	IEEE 802.15.4
AES_GCM [167]	BCAE	GCM	E/D: ×	×	367	2369572	16 bits MSP	IEEE 802.15.4
AES_GCM [167]	BCAE	GCM	E/D: ×	×	367	1197073	32 bits ARM	IEEE 802.15.4
AES-CLOC [112]	BCAE	CLOC	×:750 Per 16 bytes	2980	362	1999	8 bits AVR	×
Scream-10 [149]	BCAE	TAE	×	3221 Words	80 Words	E: 7646 D: 7672	8 bits AVR	×
AES-SILC [150]	BCAE	SILC	×: 4.9	×	×	×	Intel 64 bits	×
LED-SILC [150]	BCAE	SILC	×: 40	×	×	×	Intel 64 bits	×
ACORN_v2 [168]	SCAE	×	E: 6.38 D: 6.52	×	×	×	amd64 bits	×
ACORN_v2 [162]	SCAE	×	E/D: ×	×	184	464381	8 bits AVR	IEEE 802.15.4
ACORN_v2 [162]	SCAE	×	E/D: ×	×	184	626192	16 bits MSP	IEEE 802.15.4
ACORN_v2 [162]	SCAE	×	E/D: ×	×	184	267168	32 bits ARM	IEEE 802.15.4
ACORN_v3 [168]	SCAE	×	E: 6.23 D: 6.36	×	×	×	amd64 bits	IEEE 802.15.4
ACORN [120]	SCAE	×	E: 54.13	×	×	×	amd64 bits	IEEE 802.15.4
ASCON_v1.2 [162]	PBAE	Monkey duplex	E/D: ×	×	183	534908	8 bits AVR	IEEE 802.15.4
ASCON_v1.2 [162]	PBAE	Monkey duplex	E/D: ×	×	183	619523	16 bits MSP	IEEE 802.15.4
ASCON_v1.2 [162]	PBAE	Monkey duplex	E/D: ×	×	183	83118	32 bits ARM	IEEE 802.15.4
ASCON_v1.2 [120]	PBAE	Monkey duplex	E: 16.41	×	×	×	amd64 bits	×
ASCON [168]	PBAE	Monkey duplex	E: 7.32 D: 7.38	×	×	×	amd64 bits	×
AES-CLOC [120]	BCAE	CLOC	E: 81.76	×	×	×	amd64 bits	×
JOLTIK [120]	BCAE	TAE	E: 1590.87	×	×	×	amd64 bits	×
PRIMATEs [120]	PBAE	GIBBON	E 6611.66	×	×	×	amd64 bits	×
SCREAM [120]	BCAE	TAE	E: 54.58	×	×	×	amd64 bits	×
Aegis [120]	BCAE	×	E: 2.15	×	×	×	amd64 bits	×
AEZ [120]	BCAE	XEX	E: 4.57	×	×	×	amd64 bits	×
Deoxys [120]	BCAE	XEX	E: 16.41	×	×	×	amd64 bits	×
AES-OCB [120]	BCAE	XEX	E: 20.35	×	×	×	amd64 bits	×
NORX [162]	PBAE	Monkey duplex	E/D: ×	×	207	124062	8 bits AVR	IEEE 802.15.4
NORX [162]	PBAE	Monkey duplex	E/D: ×	×	207	75727	16 bits MSP	IEEE 802.15.4
NORX [162]	PBAE	Monkey duplex	E/D: ×	×	207	16685	32 bits ARM	IEEE 802.15.4
NORX_v3 [168]	PBAE	Monkey duplex	E: 6.9 D: 6.92	×	×	×	amd64 bits	×
NORX [120]	PBAE	Monkey duplex	E:11.9	×	×	×	amd64 bits	×
LED-SILC [120]	BCAE	SILC	E: 9.9	×	×	×	amd64 bits	×
Tiaoxin [120]	SCAE	×	E: 3.53	×	×	×	amd64 bits	×
AEZ [120]	BCAE	XEX	E: 4.57	×	×	×	amd64 bits	×
Ketje [162]	PBAE	Monkey wrap	E/D: ×	×	158	311949	8 bits AVR	IEEE 802.15.4
Ketje [162]	PBAE	Monkey wrap	E/D: ×	×	158	372720	16 bits MSP	IEEE 802.15.4
Ketje [162]	PBAE	Monkey wrap	E/D: ×	×	158	148381	32bits ARM	IEEE 802.15.4
Ketje [168]	PBAE	Monkey wrap	E: 5.35 D: 5.34	×	×	×	amd64 bits	×
Ketje [120]	PBAE	Monkey wrap	E: 173.52	×	×	×	amd64 bits	×



(a)

(b)

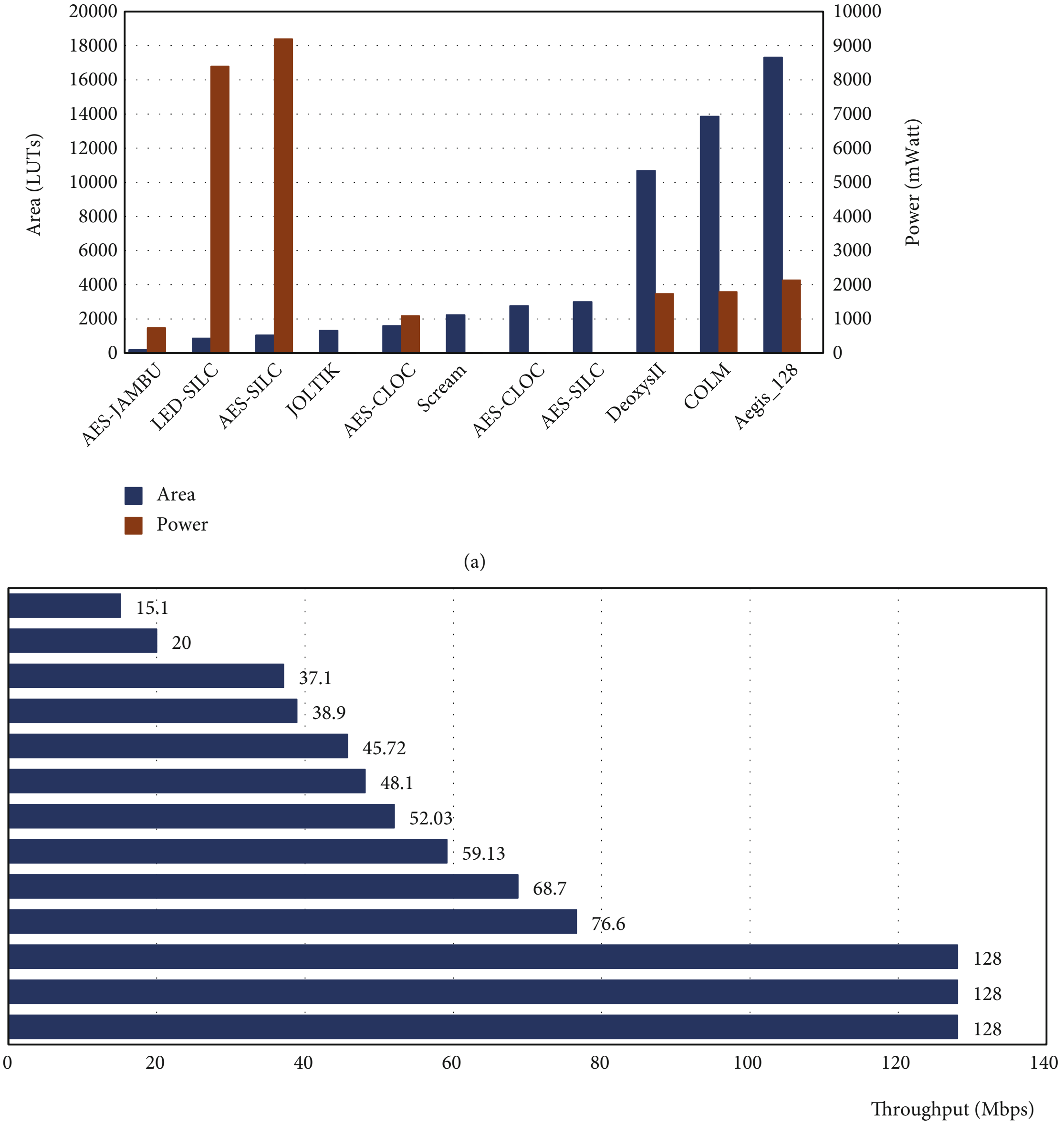
Figure 6: Metric performance of BCAE-based targeted ASIC, (a) power and area for BCAE-based targeting ASIC, (b) throughput for BCAE-based targeting ASIC.

7.6. PBAE Comparisons. NORX, Ketje_sr, Ketje_jr ASCON, and PIRMATEs performance in ASIC is shown in Figure 11(a) and 11(b). ASCON [25] outperforms other ciphers with efficient power and small implementation area. It utilizes 0.083 mm^2, consumes 0.655 mWatt in ASIC while consuming 684 LUTs [164] and 2.16 mWatt [25] in FPGA.

PRIMATE [25] is the second most efficient cipher targeting area footprint and power consumption that utilized 0.106 mm^2, 1.064 mWatt, and 66.67 Mbps in ASIC while also consuming 3.547 mWatt, 1187 LUTs, and 66.67 Mbps in FPGA. However, PRIMATE is not nonce-resistance that if nonce misuse occurs then an adversary can reveal the plaintext, unlike ASCON, which is nonce misuse resistance. Similarly, both algorithms cannot resist side channel attacks.

Ketje_jr [60] is also recommended cipher under PBAE-based category. It employs small area of 0.0172 mm^2, consumes a power of 3.27 mWatt, with high transmission of 14550 Mbps. From the optimization perspective, NORX [60] is not recommended for resource-constrained devices in ASIC or FPGA because it consumes the most of the platforms resources. The resources were reported of 0.1231 mm^2 area and 19.51 mWatt of dissipated power in ASIC while consuming 100 mWatt and area of 1424 LUTs in FPGA.

ASCON, NORX, and Ketje were benchmarked on 8 bits AVR, 16 bits MSP, and 32 bits ARM. The area utilization computed in terms of RAM were 183 bytes, 207 bytes, and 158 bytes, respectively. However, there was some fluctuation on the cycle, whereby 32bits ARM was reported with the lowest cycle count for the three ciphers as shown in



(a)

(b)

FIGURE 7: Metric performance of BCAE-based targeted FPGA, (a) power and area metrics for BCAE-based targeting FPGA, (b) throughput for BCAE-based targeting FPGA.

Figures 12(a)–12(c) to be 83118 cycles, 16685 cycles, and 148381 cycles, respectively, and these outcomes are part of IEEE802.15.4. This is due to the high processing speed of 32bits ARM compared to the AVR and MSP.

Ketje, NORX, and Ascon were the fastest algorithm in 64 bits ARM which required 10.69 cycles, 11.9 cycles, 14.7 cycles for processing AE. NORX is not resistant to nonce misuse unlike Ketje and ASCON, so consequently it is not recommended for untrusted environments even if it is reported the best performance on 64bits AMD. PRIMATEs operating in the GIBBON mode of operation were measured as having an extremely high number of cycles for processing, which reached up to 6611.66 cycles/bytes and could not resist nonce misuse nor fault attacks.

7.7. J. Encrypt and MAC Methods. OCB is an EaM scheme. The algorithm was designed as parallelizable with efficient offset calculation and low intermediate dependency that require a few cycles. The scheme overcomes the Galois Field GF(2^n) which adds an overhead to hardware implementation via modular addition, yet this requires more chip area than XOR gates. The work in [60] optimized the algorithm AES-OCB in TSMC 65 nm ASIC and utilized 0.1442 mm^2, 27.42 mWatt, and 4920 Mbps. It generated a small footprint area with high power consumption and high throughput.

On 64 bits AMD, the speed reported is 20.35 cycles/bytes. This kind of measurement derives from the fact that the algorithm required an inverted decryption, which simply adds to the cost of resources. In spite of this, it is faster on

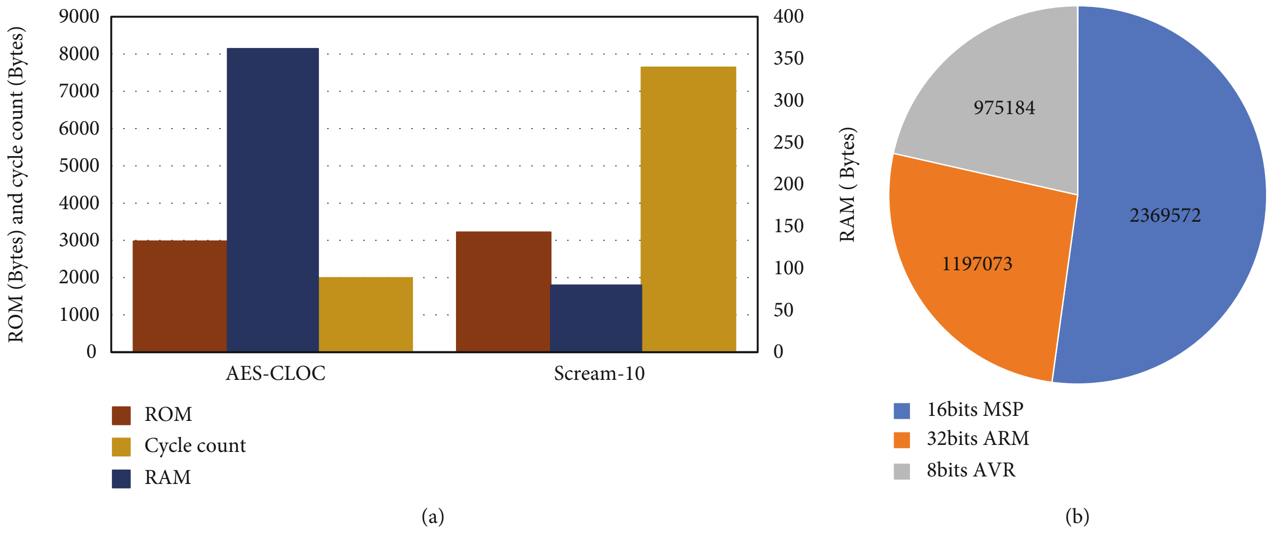


FIGURE 8: BCAE-based algorithms memory utilization and cycle count (a) BCAE-based algorithms memory utilization and cycle count, (b) cycle count of AES-GCM in software platforms.

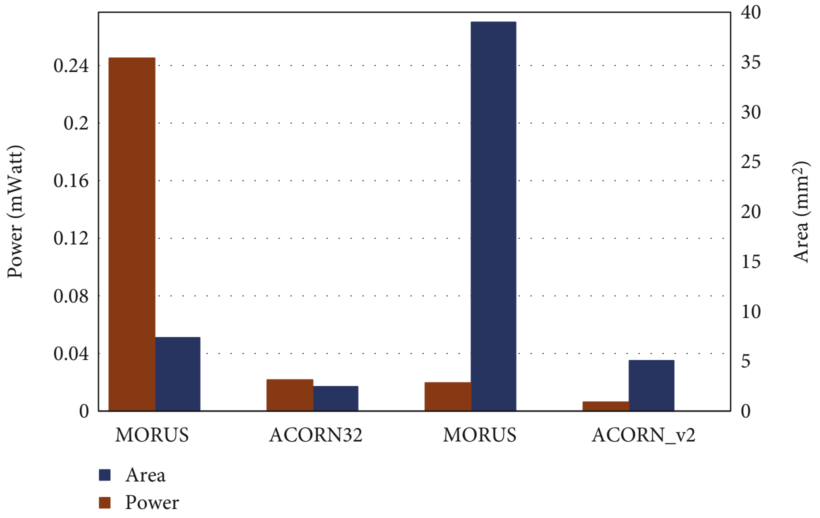


FIGURE 9: Power and area metrics performance for SCAE-based targeting ASIC.

hardware platforms since the encryption and decryption engines can be parallelized.

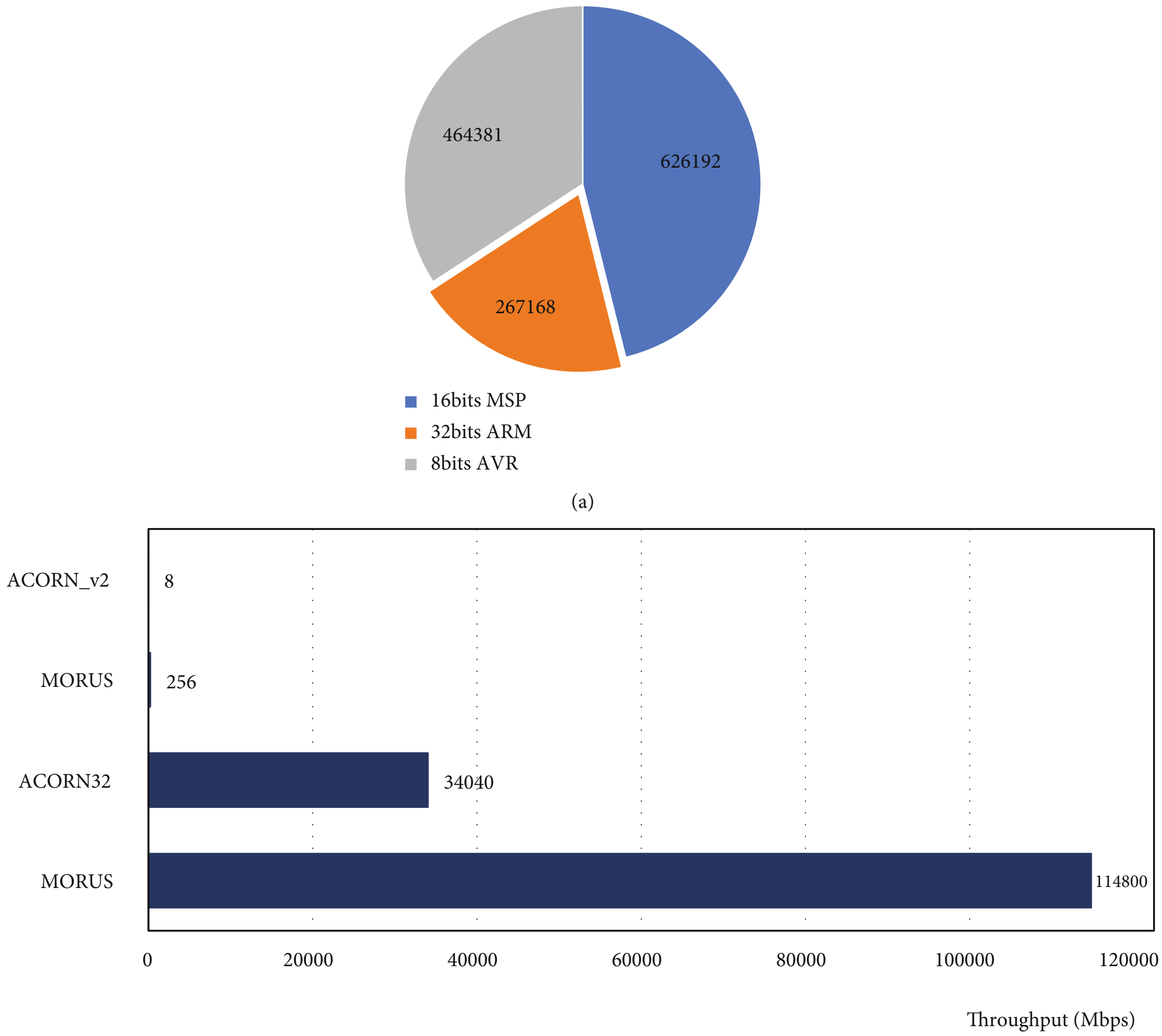
8. Open Problems

Deploying LAE to provide C and I of data on a single scheme is a future direction for smart IoT devices to follow. Data should be protected and authenticated simultaneously on transit and storage in order to protect them against unauthorized disclosure or misuse. However, there are major problems, which are explained in more detail.

8.1. Communication Limitations

(1) *Diverse Communication*. Connection to IoT devices is facilitated via a variety of wireless communications [169]. These protocols' links range from short to wide area that affects the selection of connection protocol. Establishing a cryptographic security solution should consider many properties of these protocols to ensure a practical and flexible usability. A proper LAE solution that comprehensively takes into account different wireless communication protocols and their limitations is a major research problem. For example, we can evaluate the specification of LoRaWAN, IEEE 802.15.4, and IPv6 and propose an LAE scheme based on their specification requirements. Then, benchmarking the solution for IoT devices is a future work that will address the scalable resilient requirement, which ease IoT devices' connectability to the network

(2) *Multiple Protocols*. IoT devices use different protocols to communicate. Although many protocol



(b)

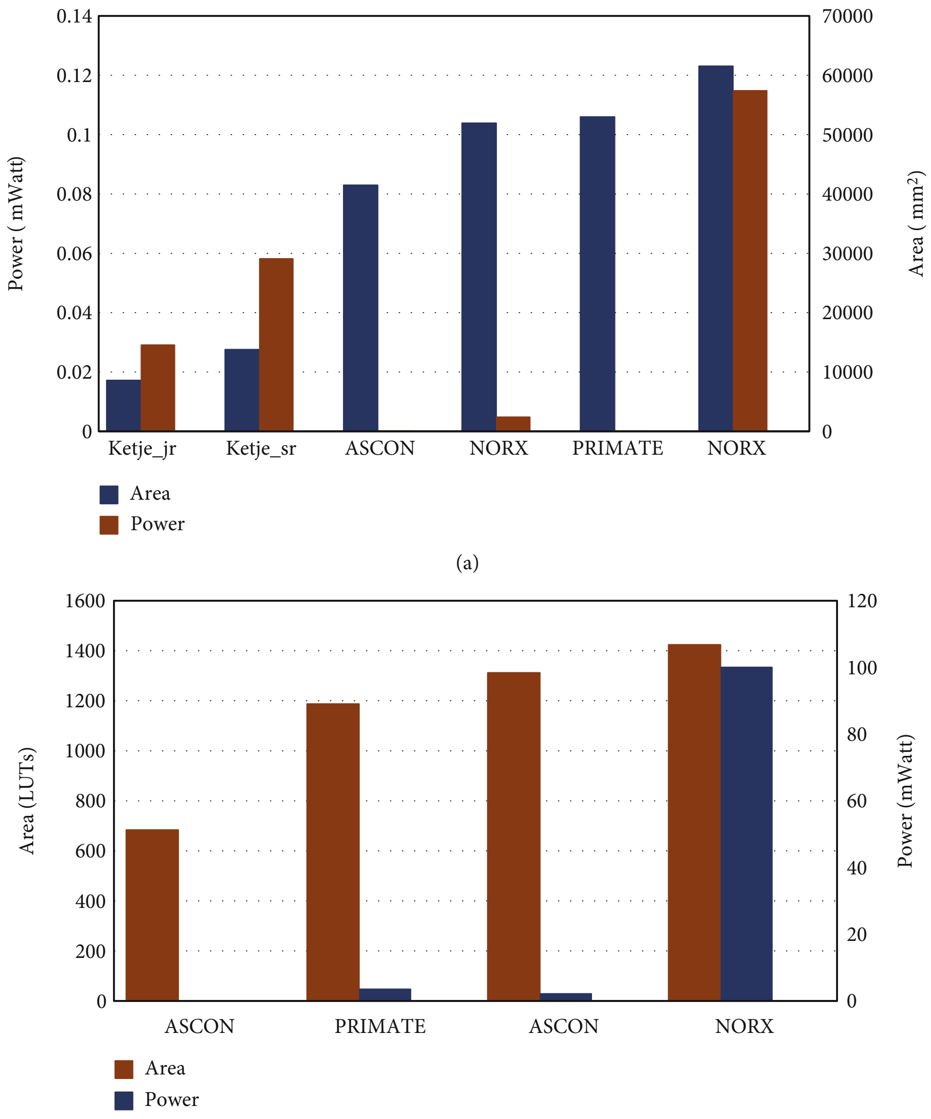
Figure 10: Metrics performance of SCAE-based targeting ASIC (a) power and area for SCAE-based targeting ASIC, (b) throughput for SCAE-based targeting ASIC.

standards exist for various IoT applications, there is no unified protocol yet [170]. As a result, the protection of data should be properly assured against unauthorized access to devices, when data is stored within them, and when data is exchanged between different devices. In these scenarios, users' data should not be manipulated nor accessed which requires multiple layers of data protection to ensure security features. A lightweight protection scheme is a significant way to curtail expenditure. However, exposure of long term keys threat is to be addressed through a lightweight protocol similar to [171]. This protocol delivers a validated perfect forward secrecy such that an adversary cannot access any previous negotiated session keys if the long-term keys are exposed. Furthermore, the protocol demonstrates a strong protection against replay attacks. Adversary capturing the communicated message and resending it again will be detected immediately

(3) *Multinetwork Approach*. IoT networks differ in their architecture [172]. A framework can include a cloud service provider with IoT edge computing node while another could not. Thus, proposing a solution, which examines a specific architecture will restrict the usability but a solution investigated for an ubiquitous framework will provide the necessary network heterogeneity

8.2. Algorithms' Design Security

(1) *Security Characteristics*. LAE schemes should consistently be examined against recent attacks [173, 174]. These could be design attack for example a quantum forgery attack or implementation attack like a fault



(a)

(b)

Figure 11: Power and area for PBAE-based targeting: (a) ASIC, (b) FPGA.

attack. However, there are many attacks to be considered in the literature and a detailed security analysis will address designers' claims. An automatic application that assesses design security characteristics and extracts cryptographic mean parameters that provide up-to-date security would be a powerful tool

(2) *Attack Assessment*. Variety of security attacks tool is a useful pre-decision assessment for developers to consider [175]. Proper cryptographic attacks can be converted to an automatic tool-based strategy to assess and validate a scheme's strength. Such a tool is useful prior to implementing the algorithm into an IoT system. Furthermore, LAE algorithms with AD are more practical for examining IoT networks. Hence, researching how to convert LAE schemes into AEAD for IoT framework is required

(3) *NIST AE*. As NIST standardizes AE under the lightweight cryptography project [55], there are 32 candidates announced in round 2. Recently, the finalists were announced and these were ASCON, Elephant, GIFT-COFB, Grain-128 AEAD, ISAP, PHOTON-Beetle, Romulus, SPARKLE, TinyJAMBU, and Xoodyak. Security analysis, attacks assessment, and platform performance comparisons are still being researched, which contributes directly to the standardization process. Comparisons of new schemes done in this paper lead the way to more research on this topic

8.3. Platforms Limitations. Proposing an encryption and authenticity solution scheme should consider devices'

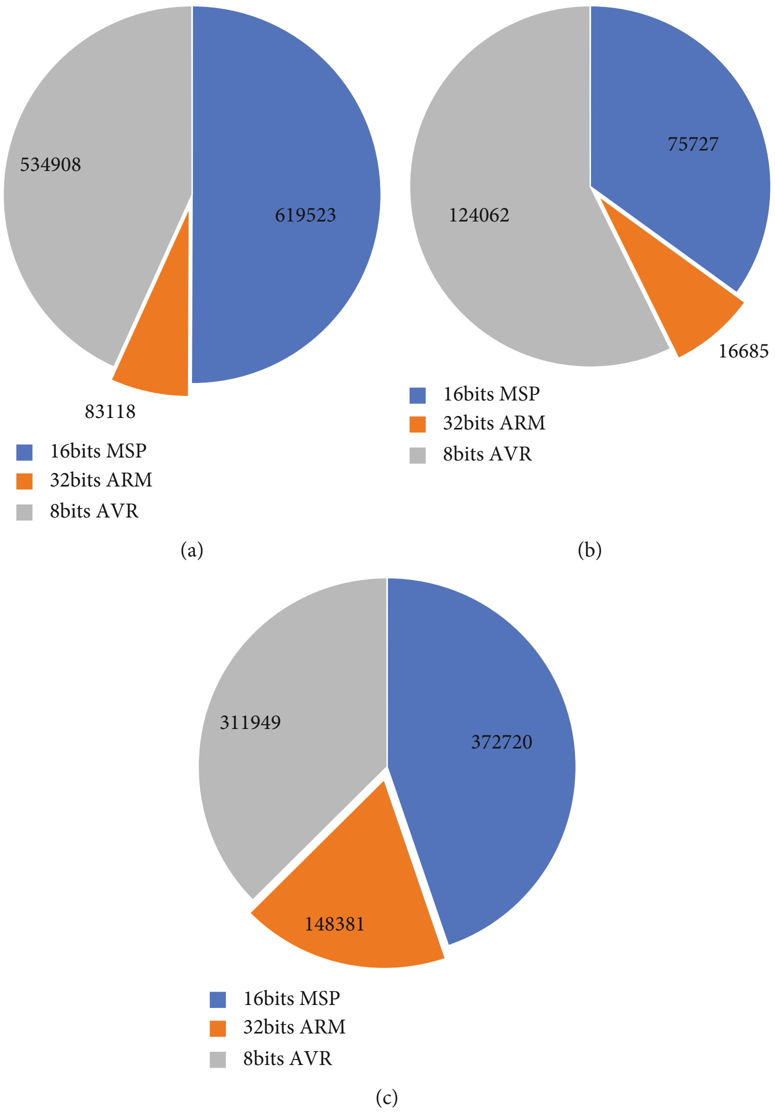


FIGURE 12: PBAE-based algorithms cycle count on 8 bits AVR, 16 bits MSP, and 32 bits ARM, (a) cycle count of ASCON_v1.2 in software platform, (b) cycle count of NORX in software platforms, and (c) cycle count of Ketje in software platforms.

diverse features, constraints, limitations, as well as providing efficient performance [176]. However, to ensure these characteristics while maintaining design security constitute a trade-off challenge. A scheme is able to resist attacks threatening IoT devices when placed in an uncontrolled environment (e.g., nonce misuse, quantum attacks, and side channel attacks) and still having to retain reasonable cost and performance efficiency, continues to be a problem.

9. Conclusion

The growth of IoT devices exposes data confidentiality and integrity breaches. IoT data encryption and checking its integrity are prerequisites for preventing information disclosure and detecting adversary data manipulation. As a result, there is a demand for a light computation scheme that can maintain a trade-off with up-to-date design security level, performance efficiency, and reasonable cost. LAE is an effective scheme compared to other lightweight cryptography primitives. It encrypts and authenticates the information as well the packet header and proposes a future direction that addresses limited device requirements. Recent studies have shown a lack of algorithm design being taken into account, leading to weak algorithms being discussed in the IoT literature or conventional cryptographic solutions that are not suitable for limited resource platforms. We presented a state-of-the-art LAE with security, design characteristics, and performance comparisons. Major problems regarding the establishment of LAE for IoT devices are highlighted here. Future research on this topic should propose a lightweight scheme for IoT devices as long as the main focus is kept on up-to-date security against attacks, with a trade-off between performance efficiency and cost of resources.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The first author would like to acknowledge financial support from the Sultanate of Oman Government. We appreciate Maliha Omar for her assistance. Finally, we thank the anonymous referees whose suggestions improved the presentation of the paper.

References

[1] S. Ghanavati, J. H. Abawajy, D. Izadi, and A. A. Alelaiwi, "Cloud-assisted IoT-based health status monitoring framework," *Cluster Computing*, vol. 20, no. 2, pp. 1843–1853, 2017.

[2] Statista, "IoT: number of connected devices worldwide 2012-2025," 2019, https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/files/172/iot-number-of-connected-devices-worldwide.html.

[3] M. Lombardi, F. Pascale, and D. Santaniello, "Internet of things: a general overview between architectures, protocols and applications," *Information*, vol. 12, no. 2, p. 87, 2021.

[4] L. D. Xu, W. He, and S. Li, "Internet of Things in industries: a Survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[5] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[6] D. Izadi, J. Abawajy, and S. Ghanavati, "An alternative node deployment scheme for WSNs," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 667–675, 2015.

[7] J. H. Abawajy and M. M. Hassan, "Federated Internet of Things and cloud computing pervasive patient health monitoring system," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 48–53, 2017.

[8] M. Maryska, P. Doucek, P. Sladek, and L. Nedomova, "Economic efficiency of the Internet of Things solution in the energy industry: a very high voltage frosting case study," *Energies*, vol. 12, no. 4, p. 585, 2019.

[9] M. Michael, "Attack landscape H1 2019: IoT, SMB traffic abound," *Threats and Research*, vol. 1, 2019https://blog.f-secure.com/attack-landscape-h1-2019-iot-smb-traffic-abound/.

[10] B. R. Ray, M. U. Chowdhury, and J. H. Abawajy, "Secure object tracking protocol for the Internet of Things," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 544–553, 2016.

[11] D. Mauro, W. Rodrigues, K. Gama, J. A. Suruagy, and P. A. D. S. Gonçalves, "Towards a multilayer strategy against attacks on IoT environments," in *2019 IEEE/ACM 1st International Workshop on Software Engineering Research & Practices for the Internet of Things (SERP4IoT)*, pp. 17–20, Montreal, QC, Canada, May 2019.

[12] E. Bertino and N. Islam, "Botnets and Internet of Things security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[13] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: an exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations," *IEEE Communication Surveys and Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019.

[14] R. Chetan and R. Shahabadkar, "A comprehensive survey on exiting solution approaches towards security and privacy requirements of IoT," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 4, p. 2319, 2018.

[15] M. Agrawal, J. Zhou, and D. Chang, "A survey on lightweight authenticated encryption and challenges for securing industrial IoT," in *Security and Privacy Trends in the Industrial Internet of Things*, C. Alcaraz, Ed., pp. 71–94, Springer International Publishing, Cham, 2019.

[16] J. Kaur, A. Kumar, and M. Bansal, "Lightweight cipher algorithms for smart cards security: a survey and open challenges," in *2017 4th International Conference on Signal Processing, Computing and Control (ISPCC)*, pp. 541–546, Solan, India, September 2017.

[17] S. Ghosh, R. Misoczki, L. Zhao, and M. R. Sastry, "Lightweight block cipher circuits for automotive and IoT sensor devices," in *Proceedings of the Hardware and Architectural Support for Security and Privacy*, Toronto, ON, Canada, 2017.

[18] B. J. Mohd, T. Hayajneh, and A. V. Vasilakos, "A survey on lightweight block ciphers for low-resource devices: comparative study and open issues," *Journal of Network and Computer Applications*, vol. 58, pp. 73–93, 2015.

[19] D. A. Saraiva, V. R. Q. Leithardt, D. de Paula, A. Sales Mendes, G. V. González, and P. Crocker, "PRISEC: comparison of symmetric key algorithms for IoT devices," *Sensors*, vol. 19, no. 19, p. 4312, 2019.

[20] S. Koteshwara and A. Das, "Comparative study of authenticated encryption targeting lightweight IoT applications," *IEEE Design & Test*, vol. 34, no. 4, pp. 26–33, 2017.

[21] O. G. Abood, M. A. Elsadd, and S. K. Guirguis, "Investigation of cryptography algorithms used for security and privacy protection in smart grid," in *2017 Nineteenth International Middle East Power Systems Conference (MEPCON)*, pp. 644–649, Cairo, Egypt, December 2017.

[22] M. Rana, Q. Mamun, and R. Islam, "Lightweight cryptography in IoT networks: a survey," *Future Generation Computer Systems*, vol. 129, pp. 77–89, 2022.

[23] E. Jintcharadze and M. Iavich, "Hybrid implementation of Twofish, AES, ElGamal and RSA cryptosystems," in *2020 IEEE East-West Design & Test Symposium (EWDTS)*, pp. 1–5, Varna, Bulgaria, September 2020.

[24] L. M. Shamala, G. Zayaraz, K. Vivekanandan, and V. Vijayalakshmi, "Lightweight cryptography algorithms for Internet of Things enabled networks: an overview," *Journal of Physics: Conference Series*, vol. 1717, no. 1, p. 012072, 2021.

[25] N. Samir, A. S. Hussein, M. Khaled et al., "ASIC and FPGA comparative study for IoT lightweight hardware security algorithms," *Journal of Circuits, Systems and Computers*, vol. 28, no. 12, p. 1930009, 2019.

[26] M. Nandi, "Forging attacks on two authenticated encryptions COBRA and POET," *IACR Cryptology ePrint Archive*, vol. 8873, p. 363, 2014.

[27] S. Roy, U. Rawat, and J. Karjee, "A lightweight cellular automata based encryption technique for IoT applications," *IEEE Access*, vol. 7, pp. 39782–39793, 2019.

[28] Y. M. Khattabi, M. M. Matalgah, and M. M. Olama, "Revisiting lightweight encryption for IoT applications: error performance and throughput in wireless fading channels with and without coding," *IEEE Access*, vol. 8, pp. 13429–13443, 2020.

[29] R. Yugha and S. Chithra, "A survey on technologies and security protocols: reference for future generation IoT," *Journal of Network and Computer Applications*, vol. 169, p. 102763, 2020.

[30] I. Ali and F. Li, "An efficient conditional privacy-preserving authentication scheme for vehicle-to-infrastructure communication in VANETs," *Vehicular Communications*, vol. 22, p. 100228, 2020.

[31] S.-Y. Tan, K.-W. Yeow, and S. O. Hwang, "Enhancement of a lightweight attribute-based encryption scheme for the Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6384–6395, 2019.

[32] U. Chatterjee, R. S. Chakraborty, and D. Mukhopadhyay, "A PUF-based secure communication protocol for IoT," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 3, p. 67, 2017.

[33] L. Guo, M. Dong, K. Ota et al., "A secure mechanism for big data collection in large scale Internet of vehicle," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 601–610, 2017.

[34] A. A. Alamr, F. Kausar, J. Kim, and C. Seo, "A secure ECC-based RFID mutual authentication protocol for Internet of Things," *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4281–4294, 2018.

[35] S. Huda, S. Miah, J. Yearwood, S. Alyahya, H. Al-Dossari, and R. Doss, "A malicious threat detection model for cloud assisted internet of things (CoT) based industrial control system (ICS) networks using deep belief network," *Journal of Parallel and Distributed Computing*, vol. 120, pp. 23–31, 2018.

[36] R. H. Randhawa, A. Hameed, and A. N. Mian, "Energy efficient cross-layer approach for object security of CoAP for IoT devices," *Ad Hoc Networks*, vol. 92, p. 101761, 2019.

[37] R. A. Rahman and B. Shah, "Security analysis of IoT protocols: a focus in CoAP," in *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, pp. 1–7, Muscat, Oman, March 2016.

[38] Y. Meng, W. Zhang, H. Zhu, and X. S. Shen, "Securing consumer IoT in the smart home: architecture, challenges, and countermeasures," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 53–59, 2018.

[39] K. Mohajerani, R. Haeussler, R. Nagpal et al., "Hardware benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 164–169, Grenoble, France, February 2021.

[40] T. S. Sector, "Overview of the Internet of Things," *International Telecomunication Union*, vol. 1, 2020https://www.itu.int/rec/T-REC-Y.2060/en.

[41] A. M. Yousuf, E. M. Rochester, and M. Ghaderi, "A low-cost LoRaWAN testbed for IoT: implementation and measurements," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pp. 361–366, Singapore, February 2018.

[42] L. Catarinucci, D. De Donno, L. Mainetti et al., "An IoT-aware architecture for smart healthcare systems," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, 2015.

[43] S. Li, "Chapter 1 - introduction: securing the Internet of Things," in *Securing the Internet of Things*, S. Li and L. D. Xu, Eds., pp. 1–25, Syngress, Boston, 2017.

[44] P. V. Paul and R. Saraswathi, "The Internet of Things — a comprehensive survey," in *2017 International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC)*, pp. 421–426, Melmaruvathur, India, 2017.

[45] S. Li, "Chapter 3 - security and vulnerability in the Internet of Things," in *Securing the Internet of Things*, S. Li and L. D. Xu, Eds., pp. 49–68, Syngress, Boston, 2017.

[46] H. Q. A. Mahri, L. Simpson, H. Bartlett, E. Dawson, and K. K.-H. Wong, "Forgery attacks on ++AE authenticated encryption mode," in *presented at the Proceedings of the Australasian Computer Science Week Multiconference*, Canberra, Australia, 2016.

[47] M. Wazid, A. K. Das, S. Shetty, J. J. P. C. Rodrigues, and Y. Park, "LDAKM-EIoT: lightweight device authentication and key management mechanism for edge-based IoT deployment," *Sensors*, vol. 19, no. 24, p. 5539, 2019.

[48] F. Dalipi and S. Y. Yayilgan, "Security and privacy considerations for IoT application on smart grids: survey and research challenges," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 63–68, Vienna, Austria, 2016.

[49] A. Mosenia and N. K. Jha, "A comprehensive study of security of Internet-of-Things," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 4, pp. 586–602, 2017.

[50] D. N. Le, C. Bhatt, and M. Madhukar, *Security Designs for the Cloud, IoT, and Social Networking*, Wiley, 2019.

[51] D. Dolev, C. Dwork, and M. Naor, "Nonmalleable cryptography," *SIAM Review*, vol. 45, no. 4, pp. 727–784, 2003.

[52] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412, pp. 223–241, 2017.

[53] X. Li, M. Wang, H. Wang, Y. Yu, and C. Qian, "Toward secure and efficient communication for the Internet of Things," *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 621–634, 2019.

[54] S. Dziembowski, K. Pietrzak, and D. Wichs, "Non-malleable codes," *Journal of the ACM*, vol. 65, no. 4, pp. 1–32, 2018.

[55] M. S. Turan, K. McKay, D. Chang et al., "Status report on the second round of the NIST lightweight cryptography standardization process," *National Institute of Standards and Technology Internal Report*, vol. 8369, no. 10, p. 6028, 2021.

[56] M. Bellare and C. Namprempre, "Authenticated encryption: relations among notions and analysis of the generic composition paradigm," *Journal of Cryptology*, vol. 21, no. 4, pp. 469–491, 2008.

[57] S. Vaudenay, "Security flaws induced by CBC padding—applications to SSL, IPSEC, WTLS," in *Advances in Cryptology — EUROCRYPT 2002. EUROCRYPT 2002. Lecture Notes in Computer Science*, vol. 2332, Springer, Berlin, Heidelberg.

[58] P. Rogaway and T. Shrimpton, "A provable-security treatment of the key-wrap problem," in *Advances in Cryptology - EUROCRYPT 2006. EUROCRYPT 2006. Lecture Notes in Computer Science*, vol. 4004, Springer, Berlin, Heidelberg.

[59] A. W. Atamli and A. Martin, "Threat-based security analysis for the Internet of Things," in *2014 International Workshop on Secure Internet of Things (SIoT)*, pp. 35–43, Wroclaw, Poland, 2014.

[60] S. Kumar, J. Haj-Yahya, M. Khairallah, M. A. Elmohr, and A. Chattopadhyay, "A comprehensive performance analysis of hardware implementations of CAESAR candidates," *Cryptology ePrint Archive*, vol. 2017, p. 1261, 2017, 2019/12/16/03 : 27 : 58. http://eprint.iacr.org/2017/1261.

[61] "Inside the smart home: IoT device threats and attack scenarios - security news - trend micro AU," 2019, https://www.trendmicro.com/vinfo/au/security/news/internet-of-things/

inside-the-smart-home-iot-device-threats-and-attack-scenarios.

[62] N. A. Moldovyan, A. A.-M. Nashwan, D. T. Nguyen, N. H. Nguyen, and H. M. Nguyen, "Deniability of symmetric encryption based on computational indistinguishability from probabilistic ciphering," in *Information Systems Design and Intelligent Applications*, pp. 209–218, Springer, 2018.

[63] R. Cheng, J. Yan, C. Guan, F. Zhang, and K. Ren, "Verifiable searchable symmetric encryption from indistinguishability obfuscation," in *Proceedings of the 10th ACM symposium on information, computer and communications security*, pp. 621–626, Singapore Republic of Singapore, 2015.

[64] C. Guo, X. Fu, Y. Mao, G. Wu, F. Li, and T. Wu, "Multi-user searchable symmetric encryption with dynamic updates for cloud computing," *Information*, vol. 9, no. 10, p. 242, 2018.

[65] V. Koppula and B. Waters, "Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption," in *Advances in Cryptology – CRYPTO 2019. CRYPTO 2019. Lecture Notes in Computer Science*, vol. 11693, Springer, Cham.

[66] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai, "Zero-knowledge proofs on secret-shared data via fully linear PCPs," in *Advances in Cryptology – CRYPTO 2019. CRYPTO 2019. Lecture Notes in Computer Science*, vol. 11694, Springer, Cham.

[67] J. Katz and M. Yung, "Unforgeable encryption and chosen ciphertext secure modes of operation," in *Fast Software Encryption*, G. Goos, J. Hartmanis, J. Leeuwen, and B. Schneier, Eds., vol. 1978, pp. 284–299, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[68] K. Manandhar, X. Cao, F. Hu, and Y. Liu, "Detection of faults and attacks including false data injection attack in smart grid using Kalman filter," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 4, pp. 370–379, 2014.

[69] C. Dobraunig, *On the Security and Design of Authenticated Encryption, [Ph.D. thesis]*, Graz University of Technology, 2017.

[70] C. Dobraunig, M. Eichlseder, T. Korak, V. Lomné, and F. Mendel, "Statistical fault attacks on nonce-based authenticated encryption schemes," in *Advances in Cryptology – ASIACRYPT 2016*, J. H. Cheon and T. Takagi, Eds., vol. 10031, pp. 369–395, Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[71] J. Balasch, B. Gierlichs, and I. Verbauwhede, "An in-depth and black-box characterization of the effects of clock glitches on 8-bit MCUs," in *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography*, pp. 105–114, Nara, Japan, 2011.

[72] "CMS responds to data breach affecting 75,000 in federal ACA portal | Healthcare Finance News," 2018, https://www.healthcarefinancenews.com/news/cms-responds-data-breach-affecting-75000-federal-aca-portal.

[73] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[74] S. Wu, H. Wu, T. Huang, M. Wang, and W. Wu, "Leaked-state-forgery attack against the authenticated encryption algorithm ALE," in *Advances in Cryptology - ASIACRYPT 2013. ASIACRYPT 2013. Lecture Notes in Computer Science*, vol. 8269, Springer, Berlin, Heidelberg.

[75] Y. Liu, Y. Sasaki, L. Song, and G. Wang, "Cryptanalysis of reduced sliscp permutation in sponge-hash and duplex-AE modes," in *Selected Areas in Cryptography – SAC 2018. SAC 2018. Lecture Notes in Computer Science*, vol. 11349, Springer, Cham.

[76] J. Lu, "Almost universal forgery attacks on the COPA and marble authenticated encryption algorithms," in *ASIA CCS '17: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 789–799, Abu Dhabi, United Arab Emirates, 2017.

[77] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: real-time intrusion detection in the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[78] N. Saxena, S. Grijalva, and N. S. Chaudhari, "Authentication protocol for an IoT-Enabled LTE network," *ACM Transactions on Internet Technology*, vol. 16, no. 4, pp. 1–20, 2016.

[79] W. Diehl, A. Abdulgadir, F. Farahmand, J.-P. Kaps, and K. Gaj, "Comparison of cost of protection against differential power analysis of selected authenticated ciphers," *Cryptography*, vol. 2, no. 3, p. 26, 2018.

[80] L. Li, G. Xu, L. Jiao et al., "A secure random key distribution scheme against node replication attacks in industrial wireless sensor systems," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2091–2101, 2020.

[81] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182–8201, 2019.

[82] T. Maitra, M. S. Obaidat, D. Giri, S. Dutta, and K. Dahal, "ElGamal cryptosystem-based secure authentication system for cloud-based IoT applications," *IET Networks*, vol. 8, no. 5, pp. 289–298, 2019.

[83] J. Hermans, R. Peeters, and B. Preneel, "Proper RFID privacy: model and protocols," *IEEE Transactions on Mobile Computing*, vol. 13, no. 12, pp. 2888–2902, 2014.

[84] J. Zhang, Y. Yu, S. Fan, Z. Zhang, and K. Yang, "Tweaking the asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes," in *Public-Key Cryptography – PKC 2020. PKC 2020. Lecture Notes in Computer Science*, vol. 12111, Springer, Cham.

[85] E. Andreeva, G. Barwell, R. Bhaumik, M. Nandi, D. Page, and M. Stam, "Turning online ciphers off," *IACR Transactions on Symmetric Cryptology*, vol. 2017, no. 2, pp. 105–142, 2017.

[86] V. T. Hoang, T. Krovetz, and P. Rogaway, "Robust authenticated-encryption AEZ and the problem that it solves," in *Advances in Cryptology – EUROCRYPT 2015. EUROCRYPT 2015. Lecture Notes in Computer Science*, vol. 9056, Springer, Berlin, Heidelberg.

[87] J. Mattsson and M. Westerlund, "Authentication key recovery on Galois/counter mode (GCM)," in *Progress in Cryptology – AFRICACRYPT 2016. AFRICACRYPT 2016. Lecture Notes in Computer Science*, vol. 9646, Springer, Cham.

[88] M. Bellare and B. Tackmann, "The multi-user security of authenticated encryption: AES-GCM in TLS 1.3," in *Advances in Cryptology – CRYPTO 2016. CRYPTO 2016. Lecture Notes in Computer Science*, vol. 9814, Springer, Berlin, Heidelberg.

[89] Y. Naito and T. Sugawara, "Lightweight authenticated encryption mode of operation for tweakable block ciphers," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, pp. 66–94, 2019.

[90] P. Grubbs, J. Lu, and T. Ristenpart, "Message franking via committing authenticated encryption," in *Advances in*

Cryptology – CRYPTO 2017. CRYPTO 2017. Lecture Notes in Computer Science, vol. 10403, Springer, Cham.

[91] P. Rogaway, “Authenticated-encryption with associated-data,” in *Proceedings of the 9th ACM conference on Computer and communications security*, pp. 98–107, Washington, DC USA, 2002.

[92] D. Chang, A. K. Chauhan, N. Gupta, A. Jati, and S. K. Sanadhya, “Exploiting the leakage: analysis of some authenticated encryption schemes,” in *Security, Privacy, and Applied Cryptography Engineering. SPACE 2016. Lecture Notes in Computer Science*, vol. 10076, Springer, Cham.

[93] E. Andreeva, V. Lallemand, A. Purnal, R. Reyhanitabar, A. Roy, and D. Vizár, “Forkcipher: a new primitive for authenticated encryption of very short messages,” in *Advances in Cryptology – ASIACRYPT 2019. ASIACRYPT 2019. Lecture Notes in Computer Science*, vol. 11922, Springer, Cham.

[94] T. Kohno, A. Palacio, and J. Black, “Building secure cryptographic transforms, or how to encrypt and MAC,” Cryptology ePrint Archive, 2003.

[95] M. Bellare and P. Rogaway, “Encode-then-encipher encryption: how to exploit nonces or redundancy in plaintexts for efficient cryptography,” in *Advances in Cryptology — ASIACRYPT 2000. ASIACRYPT 2000. Lecture Notes in Computer Science*, vol. 1976, Springer, Berlin, Heidelberg.

[96] D. McGrew and J. Viega, “The Galois/counter mode of operation (GCM),” *submission to NIST Modes of Operation Process*, vol. 20, 2004.

[97] T. Kohno, J. Viega, and D. Whiting, “The CWC-AES dual-use mode,” *Submission to NIST Modes of Operation Process*, vol. 1, 2003.

[98] S. Arvind and V. A. Narayanan, “An overview of security in CoAP: attack and analysis,” in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, pp. 655–660, Coimbatore, India, 2019.

[99] H. Almahri, “Analysis of selected block cipher modes for authenticated encryption, [Ph.D. thesis],” Queensland University of Technology, Australia, 2018.

[100] C. S. Jutla, “Encryption modes with almost free message integrity,” in *Advances in Cryptology — EUROCRYPT 2001. EUROCRYPT 2001. Lecture Notes in Computer Science*, vol. 2045, Springer, Berlin, Heidelberg.

[101] A. Chakraborti, T. Iwata, K. Minematsu, and M. Nandi, “Blockcipher-based authenticated encryption: how small can we go?,” *Journal of Cryptology*, vol. 33, no. 3, pp. 703–741, 2020.

[102] P. Rogaway, M. Bellare, and J. Black, “OCB: a block-cipher mode of operation for efficient authenticated encryption,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 6, no. 3, pp. 365–403, 2003.

[103] A. Zúquete and P. Guedes, “Efficient error-propagating block chaining,” in *Crytography and Coding. Cryptography and Coding 1997. Lecture Notes in Computer Science*, vol. 1355, Springer, Berlin, Heidelberg.

[104] K. Minematsu, “Parallelizable rate-1 authenticated encryption from pseudorandom functions,” in *Advances in Cryptology – EUROCRYPT 2014. EUROCRYPT 2014. Lecture Notes in Computer Science*, vol. 8441, Springer, Berlin, Heidelberg.

[105] T. Krovetz and P. Rogaway, “The software performance of authenticated-encryption modes,” in *Fast Software Encryption. FSE 2011. Lecture Notes in Computer Science*, vol. 6733, Springer, Berlin, Heidelberg.

[106] M. Liskov, R. L. Rivest, and D. Wagner, “Tweakable block ciphers,” in *Advances in Cryptology — CRYPTO 2002. CRYPTO 2002. Lecture Notes in Computer Science*, vol. 2442, Springer, Berlin, Heidelberg.

[107] H. Delfs, H. Knebl, and H. Knebl, *Introduction to Cryptography*, Springer, 2002.

[108] D. Whiting, R. Housley, and N. Ferguson, *Counter with cbc-mac (ccm)*, RFC3610, 2003.

[109] M. Bellare, P. Rogaway, and D. Wagner, “The EAX mode of operation,” in *Fast Software Encryption. FSE 2004. Lecture Notes in Computer Science*, vol. 3017, Springer, Berlin, Heidelberg.

[110] T. Kohno, J. Viega, and D. Whiting, “CWC: A high-performance conventional authenticated encryption mode,” in *Fast Software Encryption. FSE 2004. Lecture Notes in Computer Science*, vol. 3017, Springer, Berlin, Heidelberg.

[111] Y. Naito, M. Matsui, T. Sugawara, and D. Suzuki, “SAEB: a lightweight blockcipher-based AEAD mode of operation,” Cryptology ePrint Archive, 2019.

[112] T. Iwata, K. Minematsu, J. Guo, and S. Morioka, “CLOC: authenticated encryption for short input,” in *Fast Software Encryption. FSE 2014. Lecture Notes in Computer Science*, vol. 8540, Springer, Berlin, Heidelberg.

[113] T. H. Hongjun Wu, “The JAMBU lightweight authentication encryption mode (v2. 1),” *CAESAR Competition Proposal*, vol. 1, 2016.

[114] S. R. Nagpaul and S. K. Jain, *Topics in Applied Abstract Algebra*, Thomson Brooks/Cole, 2005.

[115] M. R. Adhikari and A. Adhikari, *Basic Modern Algebra with Applications*, Springer India, 2013.

[116] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, “Permutation-based encryption, authentication and authenticated encryption,” Directions in Authenticated Ciphers, 2012.

[117] I. Kuon and J. Rose, “Measuring the gap between FPGAs and ASICs,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203–215, 2007.

[118] A. Ehliar and D. Liu, “An ASIC perspective on FPGA optimizations,” in *2009 International Conference on Field Programmable Logic and Applications (FPL)*, pp. 218–223, Prague, Czech Republic, 2009.

[119] M. D. Aagaard, M. Sattarov, and N. Zidaric, “Hardware design and analysis of the ACE and WAGE ciphers,” 2019, http://arxiv.org/abs/1909.12338.

[120] R. Ankele and R. Ankele, “Software benchmarking of the 2nd round CAESAR candidates,” *IACR Cryptology ePrint Archive*, vol. 2016, p. 740, 2016, https://api.semanticscholar.org/db00fe6cffa7ab78e80e9a61fc4ec15fb2481c16?_ga=2.231023199.1524844265.1576644624-134252252.1576552945.

[121] W. Trappe, R. Howard, and R. S. Moore, “Low-energy security: limits and opportunities in the Internet of Things,” *IEEE Security and Privacy*, vol. 13, no. 1, pp. 14–21, 2015.

[122] D.-D. Dinu, A. Biryukov, J. Groszschaedl, D. Khovratovich, Y. L. Corre, and L. Perrin, “FELICS - fair evaluation of lightweight cryptographic systems,” p. 2015, 2015, https://api.semanticscholar.org/980fef72a502338685c6b4ad2aedf424b7560691.

[123] X. Jia, Q. Feng, T. Fan, and Q. Lei, “RFID technology and its applications in Internet of Things (IoT),” in *2012 2nd*

International Conference on Consumer Electronics, Communications and Networks (CECNet), pp. 1282–1285, Yichang, China, 2012.

[124] J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based IoT: challenges," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 26–33, 2017.

[125] J. Salowey, A. Choudhury, and D. McGrew, "AES Galois counter mode (GCM) cipher suites for TLS," *Request for Comments*, vol. 5288, 2008.

[126] K. Igoe and J. Solinas, "AES Galois counter mode for the secure shell transport layer protocol," *IETF Request for Comments*, vol. 5647, 2009.

[127] L. Law and J. Solinas, "Suite B cryptographic suites for IPsec," *IETF Request for Comments*, vol. 4869, 2007.

[128] S. Gueron, A. Langley, and Y. Lindell, "AES-GCM-SIV: specification and analysis," *IACR Cryptology ePrint Archive*, vol. 2017, p. 168, 2017.

[129] H. Q. Al Mahri, L. Simpson, H. Bartlett, E. Dawson, and K. K.-H. Wong, "A fault-based attack on AEZ v4.2," in *2017 IEEE Trustcom/BigDataSE/ICESS*, pp. 634–641, Sydney, NSW, Australia, 2017.

[130] W. Diehl, F. Farahmand, A. Abdulgadir, J.-P. Kaps, and K. Gaj, "Face-off between the CAESAR lightweight finalists: ACORN vs. Ascon," in *2018 International Conference on Field-Programmable Technology (FPT)*, p. 184, Naha, Japan, December 2018.

[131] M. Liu and D. Lin, "Cryptanalysis of lightweight authenticated cipher ACORN," in *Posed on the Crypto-Competition Mailing List*, Crypto Competition, 2014.

[132] X. Zhang, X. Feng, and D. Lin, "Fault attack on ACORN v3," *The Computer Journal*, vol. 61, no. 8, pp. 1166–1179, 2018.

[133] A. Adomnicai, J. J. Fournier, and L. Masson, "Masking the lightweight authenticated ciphers ACORN and Ascon in software," *IACR Cryptology ePrint Archive*, vol. 2018, p. 708, 2018.

[134] H. Groß, E. Wenger, C. Dobraunig, and C. Ehrenhöfer, "Suit up!–made-to-measure hardware implementations of ASCON," in *2015 Euromicro Conference on Digital System Design*, pp. 645–652, Madeira, Portugal, 2015.

[135] H. Gross and S. Mangard, "Reconciling $d+1$ masking in hardware and software," in *Cryptographic Hardware and Embedded Systems – CHES 2017. CHES 2017. Lecture Notes in Computer Science*, W. Fischer and N. Homma, Eds., vol. 10529, Springer, Cham, 2017.

[136] R. AlTawy, R. Rohit, M. He, K. Mandal, G. Yang, and G. Gong, "sLiSCP: Simeck-based permutations for lightweight sponge cryptographic primitives," in *Selected Areas in Cryptography – SAC 2017. SAC 2017. Lecture Notes in Computer Science*, vol. 10719, Springer, Cham.

[137] M. Aagaard, R. AlTawy, G. Gong, K. Mandal, R. Rohit, and N. Zidaric, "WAGE: An Authenticated Cipher," *IACR Transactions on Symmetric Cryptology*, pp. 132–159, 2020.

[138] M. Aagaard, R. AlTawy, G. Gong, K. Mandal, and R. Rohit, *ACE: An Authenticated Encryption and Hash Algorithm*, Submission to NIST-LWC, 2019.

[139] C. Dobraunig and B. Mennink, *Elephant v1*, NIST, 2019.

[140] S. Gueron, A. Jha, and M. Nandi, *COMET: counter mode encryption with authentication tag*, NIST, 2019.

[141] M. Khairallah, "Weak keys in the rekeying paradigm: application to COMET and mixFeed," *IACR Transactions on Symmetric Cryptology*, vol. 2019, pp. 272–289, 2020.

[142] D. A. McGrew and J. Viega, "The security and performance of the Galois/counter mode (GCM) of operation," in *Progress in Cryptology - INDOCRYPT 2004. INDOCRYPT 2004. Lecture Notes in Computer Science*, vol. 3348, Springer, Berlin, Heidelberg.

[143] B. Lapid and A. Wool, "Cache-attacks on the ARM TrustZone implementations of AES-256 and AES-256-GCM via GPU-based analysis," in *Selected Areas in Cryptography – SAC 2018. SAC 2018. Lecture Notes in Computer Science*, vol. 11349, Springer, Cham.

[144] H. Wu, "ACORN: a lightweight authenticated cipher (v3)," *Candidate for the CAESAR Competition*, vol. 2016, 2016https://competitions.cr.yp.to/round3/acornv3.pdf.

[145] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2: Lightweight authenticated encryption and hashing," *Journal of Cryptology*, vol. 34, no. 3, pp. 1–42, 2021.

[146] D. B. Roy, A. Chakraborti, D. Chang, S. D. Kumar, D. Mukhopadhyay, and M. Nandi, "Fault based almost universal forgeries on CLOC and SILC," in *Security, Privacy, and Applied Cryptography Engineering. SPACE 2016. Lecture Notes in Computer Science*, vol. 10076, Springer, Cham.

[147] J. Jean, I. Nikolić, and T. Peyrin, "Joltik v1. 3," *CAESAR Round*, vol. 2, 2015.

[148] E. Andreeva, B. Bilgin, A. Bogdanov et al., "PRIMATEs v1," Submission to CAESAR, 2014.

[149] V. Grosso, G. Leurent, F. X. Standaert et al., "SCREAM & iSCREAM side-channel resistant authenticated encryption with masking," Submission to CAESAR, 2014.

[150] T. Iwata, K. Minematsu, J. Guo, S. Morioka, and E. Kobayashi, "SILC: simple lightweight CFB," CAESAR submission, 2014.

[151] H. Wu and B. Preneel, "AEGIS: a fast authenticated encryption algorithm," in *Selected Areas in Cryptography – SAC 2013. SAC 2013. Lecture Notes in Computer Science*, vol. 8282, Springer, Berlin, Heidelberg.

[152] E. Andreeva, A. Bogdanov, N. Datta et al., *COLM v1*, Submission to the CAESAR Competition, CASEAR, 2016.

[153] J. Jean, I. Nikolic, T. Peyrin, and Y. Seurin, *Deoxys v1. 41*, Submitted to CAESAR, 2016.

[154] J.-P. Aumasson, P. Jovanovic, and S. Neves, "NORX: parallel and scalable AEAD," in *Computer Security - ESORICS 2014. ESORICS 2014. Lecture Notes in Computer Science*, vol. 8713, Springer, Cham.

[155] H. Wu and T. Huang, "JAMBU lightweight authenticated encryption mode and AES-JAMBU," *CAESAR Competition Proposal*, vol. 1, 2014.

[156] I. Nikolic, "Tiaoxin-346," *Submission to the CAESAR Competition*, vol. 1, 2014.

[157] I. Salam, H. Q. A. Mahri, L. Simpson, H. Bartlett, E. Dawson, and K. K.-H. Wong, "Fault attacks on Tiaoxin-346," in *Proceedings of the Australasian Computer Science Week Multiconference*, pp. 1–9, Brisband Queensland Australia, 2018.

[158] V. T. Hoang, T. Krovetz, and P. Rogaway, "AEZ v1: authenticated-encryption by enciphering," CAESAR 1st Round, Competitions, 2014, cr. yp. to/round1/aezv1.pdfl.

[159] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche, and R. Van Keer, "CAESAR submission: Ketje v2," CAESAR First Round Submission, 2014.

[160] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche, and R. Van Keer, *Keyak v2*, CAESAR Submission, 2015.

[161] H. Wu and T. Huang, "The authenticated cipher MORUS (v2)," CAESAR submission, 2014.

[162] L. C. dos Santos, J. Großschadl, and A. Biryukov, "FELICS-AEAD: benchmarking of lightweight authenticated encryption algorithms," in *Smart Card Research and Advanced Applications. CARDIS 2019. Lecture Notes in Computer Science*, vol. 11833, Springer, Cham.

[163] M. Khairallah, A. Chattopadhyay, and T. Peyrin, "Looting the LUTs: FPGA optimization of AES and AES-like ciphers for authenticated encryption," in *Progress in Cryptology – INDOCRYPT 2017*, A. Patra and N. P. Smart, Eds., vol. 10698, pp. 282–301, Springer International Publishing, Cham, 2017.

[164] F. Farahmand, W. Diehl, A. Abdulgadir, J.-P. Kaps, and K. Gaj, "Improved lightweight implementations of CAESAR authenticated ciphers," in *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 29–36, Boulder, CO, USA, 2018.

[165] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, and M. Rossi, "Secure communication for smart IoT objects: protocol stacks, use cases and practical examples," in *2012 IEEE international symposium on a world of wireless, mobile and multimedia networks (WoWMoM)*, pp. 1–7, San Francisco, CA, USA, 2012.

[166] M.-J. O. Saarinen, "Cycling attacks on GCM, GHASH and other polynomial MACs and hashes," in *Fast Software Encryption. FSE 2012. Lecture Notes in Computer Science*, vol. 7549, Springer, Berlin, Heidelberg.

[167] M. Tempelmeier, G. Sigl, and J.-P. Kaps, "Experimental power and performance evaluation of CAESAR hardware finalists," in *2018 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pp. 1–6, Cancun, Mexico, 2018.

[168] C. Wenzel-Benner and J. Gräf, "XBX: eXternal benchmarking eXtension for the SUPERCOP crypto benchmarking framework," in *Cryptographic Hardware and Embedded Systems, CHES 2010. CHES 2010. Lecture Notes in Computer Science*, vol. 6225, Springer, Berlin, Heidelberg.

[169] X. Feng, F. Yan, and X. Liu, "Study of wireless communication technologies on Internet of Things for precision agriculture," *Wireless Personal Communications*, vol. 108, no. 3, pp. 1785–1802, 2019.

[170] C. Sobin, "A survey on architecture, protocols and challenges in IoT," *Wireless Personal Communications*, vol. 112, no. 3, pp. 1383–1429, 2020.

[171] A. Ostad-Sharif, H. Arshad, M. Nikooghadam, and D. Abbasinezhad-Mood, "Three party secure data transmission in IoT networks through design of a lightweight authenticated key agreement scheme," *Future Generation Computer Systems*, vol. 100, pp. 882–892, 2019.

[172] I. Grønbæk, "Architecture for the Internet of Things (IoT): API and interconnect," in *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, pp. 802–807, Cap Esterel, France, 2008.

[173] C. Dobraunig, S. Mangard, F. Mendel, and R. Primas, "Fault attacks on nonce-based authenticated encryption: application to keyak and ketje," in *Selected Areas in Cryptography – SAC 2018. SAC 2018. Lecture Notes in Computer Science*, vol. 11349, Springer, Cham.

[174] Y. Xu, W. Liu, and W. Yu, "Quantum forgery attacks on COPA, AES-COPA and marble authenticated encryption algorithms," *Quantum Information Processing*, vol. 20, no. 4, pp. 1–21, 2021.

[175] M. Hosseini, D. Pratas, and A. J. Pinho, "Cryfa: a secure encryption tool for genomic data," *Bioinformatics*, vol. 35, no. 1, pp. 146–148, 2019.

[176] X. Luo, L. Yin, C. Li et al., "A lightweight privacy-preserving communication protocol for heterogeneous IoT environment," *IEEE Access*, vol. 8, pp. 67192–67204, 2020.