Blockchain: Research and Applications 2 (2021) 100036

Contents lists available at ScienceDirect

Blockchain: Research and Applications

journal homepage: www.journals.elsevier.com/blockchain-research-and-applications

Comparative study on hash functions for lightweight blockchain in Internet of Things (IoT)

Aishah Alfrhan [a,*], Tarek Moulahi [a], Abdulatif Alabdulatif [b]

[a] Department of Information Technology, College of Computer, Qassim University, Buraydah, 51013, Saudi Arabia
[b] Department of Computer Science, College of Computer, Qassim University, Buraydah, 51013, Saudi Arabia

ARTICLE INFO

Keywords:
Lightweight blockchain
Internet of Things (IoT)
Hash
Cybersecurity

ABSTRACT

Over the last few years, there has been a great interest in the Internet of Things (IoT). This is mainly because the IoT interacts directly with people's everyday lives in critical applications, such as in smart homes and healthcare applications. IoT devices typically have a resource-constrained architecture, rendering them vulnerable to cyberattacks. Accordingly, smart devices and stored data need to be secured through lightweight and energy-efficient security solutions, which have been identified as the main challenge facing adoption of IoT systems. To address this problem, a disruptive technology (blockchain) has been foreseen by the industry and research community as being able to deliver secure, fast, reliable, and transparent solutions for IoT systems. Hence, this work investigates the adoption of lightweight blockchain technology, mainly as a method of securing IoT systems. Because hashing plays a major role in creating a robust blockchain structure, we select a number of different hash techniques to be executed on a Raspberry Pi device. In summation, this work provides a numerical study to evaluate the performance of well-known hash functions that can be used for lightweight blockchain-based IoT.

1. Introduction

The rapid evolution of current technology has shifted attention to electronic devices connected to the Internet, which form a system called the Internet of Things (IoT). IoT comprises a network of connected devices that can sense and share data autonomously using wireless sensor networks (WSNs) and radio frequency identification (RFID) [1]. The range of IoT applications varies from smart homes and cities to healthcare applications, which interact directly with people's everyday lives. IoT systems have very specific features and a large number of devices generating significant volumes of data; thus, they require high levels of connectivity and power to continue running [1]. Additionally, these devices have limitations in terms of computing, storage, and network capacity, rendering them vulnerable to being hacked or compromised by various types of cyberattacks [2]. Therefore, security and privacy have been identified as the main dilemmas facing IoT systems [3,4].

According to recent reports, the number of IoT-connected devices is forecasted to increase every year, reaching 25 billion devices by 2021 [5]. This anticipated growth in the IoT pushes the need to develop an IoT stack and standardized protocols, and to find appropriate architectures that will provide high-quality security and services to IoT devices [6].

Today, most IoT solutions rely on the centralized server-client model, connecting to cloud servers [6]. Despite this paradigm working correctly now, the forecasted growth suggests a new schema should be developed [6]. Therefore, numerous decentralized architectures have been proposed to establish peer-to-peer (P2P) WSNs [7]. However, these faced several privacy and security issues until blockchain technology emerged. Blockchain has the potential to track, coordinate, send, and store transactions and information from a large number of devices, making it suitable for applications that do not require a centralized cloud [6].

Blockchain technology is a decentralized, shared, and immutable database ledger that stores a registry of assets and transactions across a P2P network [2]. It has a chain of blocks that store time-stamped data that have been validated by miners [2]. To validate a new block, miners must reach a consensus through a predefined consensus algorithm (such as the proof of work (PoW) algorithm), which is usually a resource-intensive task [1]. Overall, blockchain technology relies on cryptographic techniques, with hash functions playing a major role in its robust structure [8,9].

Blockchain systems can be classified into three models: public, private, and consortium [10]. Public blockchain is considered a fully decentralized network where all transactions are visible to the public and

* Corresponding author.
E-mail address: 391200341@qu.edu.sa (A. Alfrhan).

https://doi.org/10.1016/j.bcra.2021.100036
Received 8 June 2021; Received in revised form 31 October 2021; Accepted 11 November 2021

Table 1
Evaluation studies on blockchain–IoT (Internet of Things) integration.

Ref.	Goal	Criteria	Results	Recommendations
[1]	Evaluate the viability of operating blockchain platforms on IoT devices.	• Energy consumption • CPU and memory usage • Virtual memory amount • Bandwidth consumption	• Blockchain can create a great revolution to the IoT. • Light nodes are better fit for IoT.	• The benefits of this integration should be studied carefully. • Research efforts should ensure scalability, storage capacity, security, and privacy.
[15]	Study the blockchain utilization to foster decentralized and private-by-design IoT.	• Blockchain integrity, adaptability, and anonymity.	• Integrity in large blockchain systems is the most secure. • Adaptability limitation is due to scalability issues. • Anonymity is only guaranteed through pseudonymity.	• Develop IoT applications on another existing secure scalable blockchain. • A layered architecture for the IoT.
[16]	Evaluate the Ethereum blockchain to manage IoT devices.	• Transaction time • Storage	• The Ethereum is not fast enough for time-sensitive domains. • Storage issues in Ethereum.	• Using proxy or full nodes (further solutions needed).
[17]	Utilizing blockchain to enhance device authentication and authorization.	• Memory and CPU usage	• The blockchain overhead is acceptable.	–
[18]	Evaluate hosting platforms for blockchain as a service for the IoT.	• Network latency	• Fog computing perform faster than cloud computing.	–
[8]	Analyzing the adoption of lightweight hash functions in blockchain-based IoT.	• Throughput • Security	• The proposed architecture is suited for IIoT applications that require processing within certain times.	• More lightweight hash functions are required.

everyone can participate in the consensus process. Private blockchain is regarded as a centralized network where only those nodes that come from a particular organization can join the consensus process. Finally, consortium blockchain is considered a partially decentralized network where only a set of pre-selected nodes take part in the consensus [11].

Blockchain technology was first presented in the cryptocurrency sector (Bitcoin), although it now offers great potential in other applications such as e-Governance services, financial services, healthcare systems, identity verification, and traceability systems [1]. Nevertheless, according to a systematic review presented in 2016 [12], over 80% of blockchain-related research papers address the Bitcoin system, with the remainder focusing on other blockchain applications.

1.1. Motivation

The integration of blockchain into the IoT has caught the attention of both industry and research communities as a major method of securing IoT systems [2,4,13]. However, this integration is not an easy task, as several serious challenges need to be addressed: (i) high resource requirements in blockchain due to the use of consensus algorithms; (ii) scalability issues caused by the massive amount of data and the need to reach a consensus among miners; and (iii) high delays attributed to consensus algorithms and other mechanisms [14]. Consequently, the IoT needs lightweight, scalable, distributed security solutions and privacy safeguards [4], which can be achieved through lightweight and energy-efficient communication protocols and cryptographic algorithms and hashes. However, it should be noted that despite the robustness of blockchain security systems, they have vulnerabilities that can compromise their overall security [2].

1.2. Contribution

In this context, the aim of this work is to help in choosing the most efficient hash functions for resource-constrained IoT devices. To accomplish this, a numerical study is presented to evaluate the performance of different hash functions, because they play a major role in the blockchain structure.

1.3. Paper organization

The remainder of this paper is organized as follows: Section 2 presents a literature review on studies that have addressed and evaluated the integration of IoT and blockchain; Section 3 provides a summary of the examined hash functions; Section 4 covers the implementation process and its specifications; Section 5 presents the results and discussion; finally, the conclusions are reviewed in Section 6.

2. Literature review

Several studies have proposed system architectures and functions for integrating blockchain into the IoT by examining different critical factors that can affect the feasibility of this integration. In addition, some recommendations for making this integration were introduced. In this section, several evaluation studies on blockchain–IoT integration are reviewed and summarized, as shown in Table 1. In addition, some papers dedicated to evaluating the performance of different hash functions executed on constrained devices are reviewed.

In Ref. [1], the viability of operating blockchain platforms on IoT devices was evaluated by executing different nodes (light and full) from different platforms, including Bitcoin, Litecoin, and Ethereum (light nodes only) on a Raspberry Pi device. The results indicated that light nodes were a better fit for limited-resource IoT devices because they consumed less resources compared to full nodes. Furthermore, the authors stated that blockchain will probably create a great revolution in the IoT sector, meaning the benefits of this integration should be studied carefully and research efforts should ensure the scalability, storage capacity, security, and privacy of such critical technologies.

The current degree of integrity, adaptability, and anonymity of blockchain was evaluated in Ref. [15], in order to determine whether blockchain could be utilized to foster decentralized and private-by-design IoT. They concluded that integrity in large blockchain systems (such as Bitcoin) was the most secure because of its difficulty using hashing protocol PoW. However, scalability issues were also reported, making it less suitable for the IoT. Accordingly, they suggested developing IoT applications on top of another existing secure scalable blockchain. Moreover, they also proposed a layered architecture where the blockchain is separated from the application layer, and IoT devices only store part of the blockchain. Finally, they stated that anonymity was only guaranteed through pseudonymity, which is not sufficient given the possibility of being de-anonymized by several techniques.

Another study [16] suggested using blockchain to build an IoT system by controlling and configuring IoT devices. They used the Ethereum blockchain as a platform to build a key management system for authenticating devices and employed Raspberry Pis to simulate the IoT system with deployed smart contracts equipped with a public key infrastructure. The results demonstrated that Ethereum was too slow for time-sensitive applications. Furthermore, the fact that light nodes are not yet supported on Ethereum creates storage challenges for such

resource-limited devices. Accordingly, they suggested using a proxy or full nodes; however, the former solution can affect system security by adding a third party, while the latter can be too expensive for such small devices. Ultimately, they concluded that further solutions are needed to resolve the aforementioned issues.

In Ref. [17], an out-of-band two-factor authentication system for IoT devices was proposed based on blockchain technology in smart home settings. In this schema, blockchain was utilized to enhance the authentication and authorization process by storing and verifying the relationship information of IoT devices using smart contracts. The authors conducted an experiment to evaluate the performance of the proposed schema (simulated by Eris Blockchain, BeagleBone Black, and Raspberry Pi 3 devices). The results showed slight memory consumption (an average of approximately 29.5 MB) and acceptable CPU usage (an average of 29.55% and 13.35% for BeagleBone Black and Raspberry Pi 3 nodes, respectively).

An evaluation of hosting platforms for blockchain as a service for IoT (including cloud and fog computing) was conducted in Ref. [18]. Several experiments were performed to test the performance of the proposed system, which runs constrained edge devices as blockchain nodes using Intel Edison boards and IBM's Bluemix blockchain. The results indicated that fog computing performs faster than cloud computing with regard to network latency.

A few papers have contained analyses of the adoption of lightweight hash functions in blockchain-based IoT systems. For example, a lightweight hash-based blockchain architecture for Industrial IoT (IIoT) was proposed in Ref. [8]. This architecture consists of “Cell nodes” as miner nodes and “Storage nodes” as full nodes, where cell nodes are also responsible for selecting from a list of lightweight hash functions (called the lightweight hash list) for block mining, based on the network traffic. The authors worked on three lightweight hash functions (QUARK, PHOTON, and SPONGENT), all of which can be implemented in small areas with low power consumption. The simulation results exhibited good throughput and security for the selected lightweight hash functions. The authors stated that the proposed architecture was more suitable for time-limited scenarios such as IoT monitoring and surveillance applications. Finally, they argued that if more lightweight hash functions were developed, the proposed architecture could be employed for various IIoT applications (which demand a latency of <1 s).

Several papers evaluated the performance of different hash functions executed on constrained devices [19–21]. For example, the performance of different hash functions in terms of speed and memory space was evaluated in Ref. [19]. The examined hash functions included (i) SHA256 and SHA3 candidates; (ii) four lightweight hash functions in different digest sizes (including QUARK, PHOTON, SPONGENT, and the lightweight Keccak (Keccak-*f* [200] and Keccak-*f* [400])); and (iii) several blocks cipher-based constructions. The results indicated there was a high cycle count in lightweight algorithms compared to the other hash functions, while lightweight algorithms with sponge-based construction exhibited an extreme reduction in RAM usage.

In conclusion, several studies have confirmed that blockchain can strengthen IoT systems [13,22], because they are immutable, transparent, and redefine trust while offering secure, fast, reliable, and transparent solutions [22]. In addition, several challenges were identified that should be considered carefully when performing this integration. One of the main challenges is meeting the high resource requirements in blockchain with the constrained architecture of IoT devices [14]. This requires further research to develop and evaluate lightweight security mechanisms for this sector.

3. Hash functions

In this section, a brief summary of the examined hash functions is presented, including selected instances of the Secure Hash Algorithm 2 (SHA-2) [23], an SHA-3 candidate called Keccak [24], and a lightweight hash function named PHOTON [25]. The selected hash functions include different features in terms of construction, security, and performance. Keccak and SHA-2 are both popular hash functions in blockchain-based currency [8]. Further, both lightweight versions of Keccak and PHOTON exhibited acceptable performance results compared to other lightweight hash functions according to some evaluation studies [19,20].

3.1. SHA-2

The SHA-2 is a standardized hash family approved by NIST and described in FIPS PUB 180-4 [23]. It consists of various algorithms that are iterative, one-way hash functions that process an input message to create a condensed representation, termed a message digest. Each algorithm goes through two stages: preprocessing and hash computation. First, preprocessing involves padding and parsing a message into m-bit blocks, then setting the initialization values to be used later in the hash computation process. Second, from the padded message, the hash computation process produces a message schedule that is used with functions, constants, and word operations to iteratively produce a series of hash values. The final hash value created by the hash computation process is used to define the message digest. The algorithms vary in terms of blocks sizes, words of data, or message digest sizes; therefore, they offer a wide range of security levels [23].

As debated in Ref. [26], only SHA-256 and SHA-512 can be considered original designs out of all the SHA-2 functions, because the others are variants with different initial hash values and truncated digests. Therefore, in this work, we examine these two variants of the SHA-2 family. Both algorithms are based on the Merkle–Damgård structure with a Davies–Meyer compression function. SHA-256 operates on 32-bit words and processes a message block of 512-bit to produce a digest of 256-bit, providing 128-bit of security against collisions [23]. SHA-512 operates on 64-bit words and processes a message block of 1024-bit to produce a 512-bit digest, providing 256-bit of security against collisions [23]. Both algorithms use eight working variables with lengths of 32- and 64-bit in SHA-256 and SHA-512, respectively [23].

The SHA-256 is a popular hash function in cryptocurrency systems and is the hash used in the Bitcoin system to mine blocks [27]. However, the Bitcoin SHA-256 is considered insufficiently secure for long-term security by NIST [28] because it results in a particular hashing problem, termed the constrained input small output (CISO) problem, which appears in the Bitcoin mining process. This problem affects the speed and cost of running the hash and has been examined in several studies [28,29], and this problem has been thoroughly studied in the submitted SHA-3, or Keccak [28]. Moreover, SHA-512 is expected to be more critical in the future (either in high performance computing or in the IoT field) as a result of the expected emergence of the quantum cryptanalysis threat [26].

3.2. Keccak

Keccak is a family of sponge-based hash functions with seven different Keccak-*f* permutations [30]. The sponge construction in hash functions is an iterative construction for computing a function f with variable input length and an arbitrary output length using a fixed-length permutation p running on a fixed number b of bits. It has four basic parameters: state size b, rate r, capacity c, and digest size n, where $b = r+c$ [31]. The sponge construction process consists of two phases [31]. The first is the absorbing phase, which starts by padding the message and then splitting it into r-bit chunks. These are then XORed into the first r bits of the internal state, and the phase ends by applying the permutation P. The second is the squeezing phase, which iteratively returns the first r bits of the state as output blocks interleaved with permutation P. The security level of the sponge construction depends on both the capacity c and digest size n, having a collision resistance of $\min(2^{n/2}, 2^{c/2})$ and a second preimage resistance of $\min(2^{n}, 2^{c/2})$ [31].

The Keccak-*f* permutations, formulated as Keccak-*f* [b] (where b is the permutation width) include 25, 50, 100, 200, 400, 800, or 1600 bits [24]. These Keccak-*f* permutations are iterated constructions composed of a

sequence of nearly identical rounds, with each round composed of a sequence of invertible steps, with each step operating on the state. This is presented as an array of 5×5 lanes with length $w \in \{1, 2, 4, 8, 16, 32, 64\}$ (b=25w). Moreover, the Keccak parameters include the bitrate r and the capacity c as Keccak[r, c], where r+c=b and b>c. Another parameter is the number of rounds nr, which depends on the permutation width b, as nr=12+2ℓ, where 2^{ℓ}=b/25. For the same permutation, the security level can be increased at the expense of speed by trading bitrate for capacity [24].

Keccak was the winner of the SHA-3 competition in 2012, and Keccak-f [1600] was chosen as the standard instance of SHA-3 with $n \in \{224, 256, 384, 512\}$ [32], nr=24, and security level c=2n to provide an acceptable (second) preimage resistance of 2^n [24]. Moreover, it operates well on 64-bit CPUs because it represents the lane as a 64-bit word [24]. Keccak-f [200] and Keccak-f [400] are considered lightweight alternatives with c=n or c=2n [20].

3.3. PHOTON-n/r/r′

PHOTON is a sponge-based lightweight hash function with fixed-key permutations and a design similar to the advanced encryption standard (AES), with the aim of increasing overall security [25]. As stated previously, the sponge construction of hash functions can be defined by the state size b, rate r, capacity c, and digest size n, where b=r+c [20]. However, the PHOTON hash differs from other sponge-like hashes by having two values of bitrate (one for the input and one for the output) because it shrinks the latter to improve the preimage resistance [25]. Each variant in the PHOTON family is defined by the hash output size n (range 80–256 bit), the bitrate of input r, and the bitrate output r' with a state size b=n+r and capacity c=n. This is written as PHOTON-n/r/r' [25].

PHOTON hash functions consist of five flavors: PHOTON-80/20/16, PHOTON-128/16/16, PHOTON-160/36/36, PHOTON-224/32/32, and PHOTON-256/32/32 with internal permutations of P100, P144, P196, P256, and P288, respectively. Moreover, the PHOTON hash claims a collision resistance of $\min\{2^{n/2}, 2^{c/2}\}$, $\min\{2^n, 2^{c/2}\}$ for the second-preimage resistance, and $\min\{2^n, 2^c, \max\{2^{n-r}, 2^{c/2}\}\}$ for the preimage resistance [25]. Further, it provides different security levels, ranging from 40-bit to 128-bit collision resistance [25].

The internal permutation P in the PHOTON hash is composed of 12 rounds, each consisting of 4 steps: AddConstants, SubCells, ShiftRows, and MixColumnsSerial. The AddConstants step is where round constants are added to the state. The SubCells applies a substitution box (S-box) to the state, either the 4-bit PRESENT S-box or the 8-bit AES S-box, depending on the cell size (4 bits in all flavors except for PHOTON-256/32/32, which uses 8-bit cells). Each row then rotates to the left in the ShiftRows step. Finally, MixColumnsSerial mixes each column of the internal state a number of times [25].

4. Implementation

To examine the feasibility of employing blockchain in IoT systems, we conducted several experiments to evaluate the performance of different hash functions with various structures and sizes. This section presents the implementation algorithm in detail.

4.1. Implementation algorithm

The implementation algorithm shown in Algorithm 1 outlines the procedure of the hash evaluation process for a selected function. It starts with a message input to be hashed by a hash function X and ends with the evaluation step, which is based on the cycle count metric.

4.2. Implementation details

In the implementation process, selected hash functions were executed with hash small messages. Given that this work is testing lightweight cryptography, both 8- and 16-byte messages were tested. The codes were written in the C language and executed on a Raspberry Pi 4 model B device (with a 1.5 GHz ARM processor, 1 GB of RAM memory, and 32 GB of storage) and installed with the 32-bit Raspberry Pi OS. In this experiment, the Raspberry Pi device was used to simulate IoT devices because of its constrained abilities in power and computing. The implementation setup consists of an Ethernet-connected Raspberry Pi and a laptop.

Algorithm 1: Implementation algorithm

```
1: begin
2:     Input Message
3:     Get Message_size
4:     Start Cycle_counter (t0)
5:     Compute Hash_X (Message, Message_size)
6:     End Cycle_counter (t1)
7:     Compute Cycle_counter (t1-t0)
8: end
```

Furthermore, the cycle count of each hash function was computed to obtain the latency of the hash function measured in clock cycles as well as the throughput (i.e., the hash rate) [35]. The cycle count is considered a good relative measure to evaluate the performance of hash algorithms [35]. In this work, the cycle count was computed using the modern retro console (MRC) which makes the performance counters available to measure, this was executed through a loadable kernel module (LKM) on the Raspberry Pi device.

First, for SHA-2 functions, the OpenSSL cryptographic library was used to execute both SHA-256 and SHA-512 hashes, producing digest sizes of 256-bit and 512-bit, respectively. Second, for Keccak, the four SHA-3 instances were examined with an output size of n=c/2 to guarantee the acceptable security of (second) preimage resistance of 2^n, as mentioned in Section 3.2. In addition, the lightweight versions Keccak-f [400] and Keccak-f [200] were tested. The former had a message digest size of 128-bit, a capacity value of 256-bit, and a data rate of 144-bit, while the latter had a message digest size of 64-bit, a capacity value of 128-bit, and a data rate of 72-bit. Two additional Keccak-f [200] instances were examined with a bitrate value of 40-bit and a capacity value of 160-bit, one with a digest size of n=c and another with n=c/2. The Keccak source code was obtained from the GitHub repository [33]. Finally, all five PHOTON instances were tested with message digests of 80, 128, 160, 224, and 256 bits. The PHOTON source code was obtained from Ref. [34].

5. Results and discussion

In this section, the experiment results are analyzed and illustrated. For the evaluation criteria, the cycle count metric was computed to obtain the latency of a certain hash function [35]. The evaluation results are shown in Table 2 and illustrated in Figs. 1–4. The results are categorized into four sections: (i) SHA-2 functions with digest sizes of 256 bits and 512 bits; (ii) SHA-3 candidate Keccak with a state size of 1600; (iii) Keccak-f [b] represents the lightweight Keccak with state sizes of 400 and 200; and (iv) the PHOTON lightweight hash function, each with its corresponding preimage, second preimage, and collision resistances.

For the first category, SHA-256 outperforms SHA-512 because the implementation process includes working in a 32-bit environment. For the sponge-based functions, they exhibited lower performance compared to the SHA-2 functions. This was expected, as several studies have emphasized how sponge-based hash functions can have low performance with regard to cycle count when working with small messages because it may slow down the squeezing process [25]. In contrast, sponge

Table 2
Evaluation of hash functions in cycle count.

Hash	Digest size (bit)	Preimage	2nd Preimage	Collision resistances	Cycle count (8-byte message)	Cycle count (16-byte message)
SHA-2						
SHA-256	256	256	256	128	46,248	49,792
SHA-512	512	512	512	256	55,143	55,273
Keccak-*f* [1600]						
keccak_224 [$r = 1152$, $c = 448$]	224	224	224	112	115,429	116,400
keccak_256 [$r = 1088$, $c = 512$]	256	256	256	128	116,653	117,365
keccak_384 [$r = 832$, $c = 768$]	384	384	384	192	117,811	117,999
keccak_512 [$r = 576$, $c = 1024$]	512	512	512	256	118,590	118,719
Keccak-*f*[*b*]						
Keccak-*f* [400] [$r = 144$, $c = 256$]	128	128	128	64	114,239	115,391
Keccak-*f* [200] [$r = 72$, $c = 128$]	64	64	64	32	96,717	183,572
Keccak-*f* [200] [$r = 40$, $c = 160$]	80	80	80	40	294,397	465,314
Keccak-*f* [200] [$r = 40$, $c = 160$]	160	80	80	80	480,535	652,098
PHOTON-*n*/*r*/*r*′						
PHOTON-80/20/16	80	64	40	40	2,450,356	3,349,360
PHOTON-128/16/16	128	112	64	64	6,098,711	8,090,018
PHOTON-160/36/36	160	124	80	80	4,754,197	6,307,556
PHOTON-224/32/32	224	192	112	112	10,484,827	12,919,409
PHOTON-256/32/32	256	224	128	128	8,779,297	10,643,384

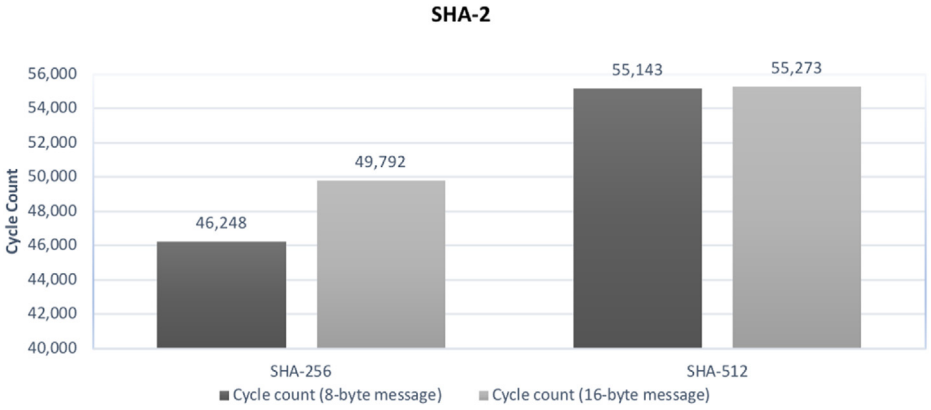

Fig. 1. Evaluation of SHA-2 in cycle count.

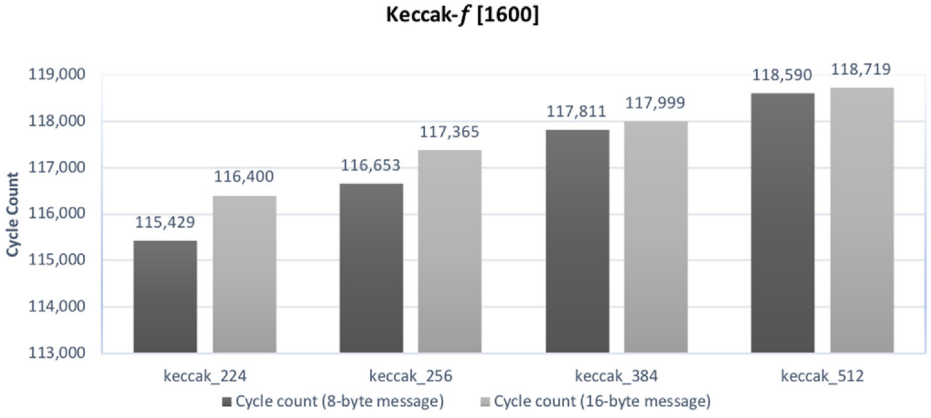

Fig. 2. Evaluation of Keccak-*f* [1600] hash functions in cycle count.

construction can be an efficient choice when the application focus is to minimize internal memory size [19,25].

Overall, all Keccak instances produced lower cycle counts compared to PHOTON. On the one hand, Keccak-*f* [1600] exhibited cycle counts ranging from 115k cycles to 118k cycles, resulting in a hashing rate of approximately 14,428.6 cycles and 14,823.8 cycles per byte (with keccak_224 being the fastest and keccak_512 being the slowest). On the other hand, Keccak-*f* [400] demonstrated slightly faster execution compared to keccak_224, with a hashing rate of approximately 14,279.9 cycles per byte when hashing an 8-byte message. Further, Keccak-*f* [200] (with a bitrate of r=72 and a capacity of c=128) had the lowest clock cycles out of all Keccak instances, producing a hashing rate of approximately 12,089.6 when hashing an 8-byte message. However, the latter has a security level of 64-bit, which is considered too low for lightweight implementations [36].

Additional instances of Keccak-*f* [200] with a bitrate value of 40-bit were tested. The results exhibited low hashing speed when hashing with a variant that had a small value of bitrate r, and offered a higher level of security. The opposite occurred when working with larger bitrate values, such as when r=72-bit. In addition, for Keccak-*f* [200] with a

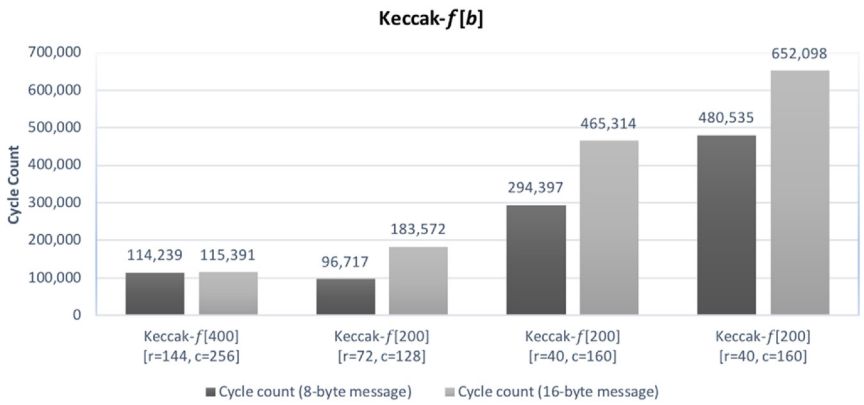

Fig. 3. Evaluation of Keccak-*f* [b] hash functions in cycle count.

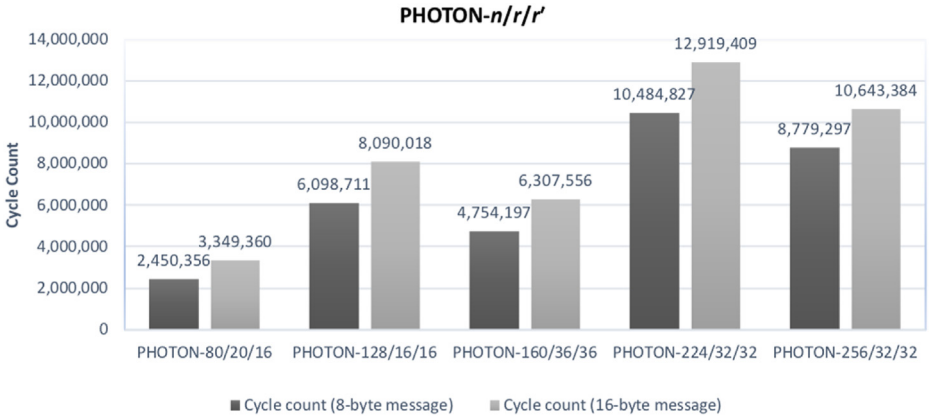

Fig. 4. Evaluation of PHOTON-n/r/r ′hash functions in cycle count.

bitrate of r=40-bit, two digest sizes, n=c/2 and n=c, were examined. The former exhibited a lower cycle count and a lower security level of 80-bit message digest size.

For the final category, the PHOTON hash function exhibited the weakest performance. This was because it had the highest cycle count out of all the examined hash functions, reaching approximately 1.3 M cycles per byte when hashing an 8-byte message by PHOTON-224/32/32. On the contrary, among the PHOTON instances, PHOTON-80/20/16 exhibited the highest performance with a hash rate value of approximately 306k cycles per byte and claimed security levels of 40-bit for collision resistance and 64-bit for preimage resistance, which is the least security level out of all other PHOTON flavors. Finally, PHOTON-256/32/32 exhibited relatively good performance (for the high security level it offers) compared to PHOTON-224/32/32.

With regard to message size, both SHA-2 and Keccak hash functions exhibited slight increases in cycle count when working with larger messages (except for Keccak variants with a state size of 200) because they exhibited a significant increase in cycle count. In addition, PHOTON also exhibited a sharp increase (20%–36%) in clock cycles when hashing a 16-byte message.

In conclusion, SHA-2 functions exhibited lower cycle counts compared to the examined dedicated lightweight hash functions, with SHA-2 providing an optimal security level, this supports the results presented in Refs. [19,21]. However, the Keccak hash function is significantly faster than the lightweight hash function PHOTON, the latter has shown the weakest performance as well in Ref. [20]. Further, for a similar security level, Keccak offers better performance in terms of speed. However, the test of lightweight versions of Keccak proved how the value of bitrate r can severely impact the hashing speed when working with small messages. Overall, the experiments indicated that hash execution can be expensive for constrained devices, with even some dedicated lightweight hash functions achieving a lower performance. Therefore, with regard to a blockchain architecture in IoT, consortium blockchain could be more feasible in this scenario, where high resource devices are responsible for storage (i.e., full nodes) and mining tasks (i.e., hash function operations), and resource-constrained IoT devices are not fully integrated into the blockchain network, working as light nodes that only store transactional data. Further, as stated in Section 2, a layered architecture for IoT is recommended, where IoT end devices (i.e., in the field layer) are kept separate from the blockchain layer.

6. Conclusion

Over recent years, the focus has been shifted to the connected schema of the IoT, resulting in significant efforts to deliver novel and effective security solutions. One promising approach is to integrate blockchain systems with the IoT. Thus, this work presents a numerical study with the aim of evaluating performance on selected hash functions with various constructions and security levels. The experiment results indicate that in terms of hashing speed, SHA-2 functions outperform the sponge-based hash functions, followed by the lightweight versions of Keccak with a high bitrate value, which supports the previous studies that have been highlighted in this work. Further, the presented results confirm the importance of developing more efficient lightweight hash functions, as the NIST's lightweight cryptography project suggested [35], which can be identified as an open issue that needs more investigation. In future work, we plan to implement a bandwidth-based performance evaluation of hash functions on multiple connected devices using the new microcontroller of Raspberry Pi (Pico).

Funding statement

The authors received no specific funding for this study.

Declaration of competing interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Reyna, C. Martín, J. Chen, et al., On blockchain and its integration with IoT. Challenges and opportunities, Future Generat. Comput. Syst. 88 (2018) 173–190.
[2] M.A. Khan, K. Salah, IoT security: review, blockchain solutions, and open challenges, Future Generat. Comput. Syst. 82 (2018) 395–411.
[3] D. Miorandi, S. Sicari, F. De Pellegrini, et al., Internet of things: vision, applications and research challenges, Ad Hoc Netw. 10 (7) (2012) 1497–1516.
[4] A. Dorri, S.S. Kanhere, R. Jurdak, et al., Blockchain for IoT security and privacy: the case study of a smart home, in: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops); 13–17 Mar 2017; Kona, HI, USA, IEEE, Piscataway, NJ, USA, 2017, pp. 618–623.
[5] G. Omale, Gartner identifies top 10 strategic IoT technologies and trends, Gartneryrket (2018). Available Online, https://www.gartner.com/newsroom/id/3812063.
[6] T.M. Fernández-Caramés, P. Fraga-Lamas, A review on the use of blockchain for the Internet of things, IEEE Access 6 (2018).
[7] S. Krco, D. Cleary, D. Parker, P2P mobile sensor networks, in: IEEE 38th Hawaii International Conference on System Sciences; 3–6 Jan 2005; Big Island, HI, USA, IEEE, Piscataway, NJ, USA, 2005, p. 324c.
[8] B. Seok, J. Park, J.H. Park, A lightweight hash-based blockchain architecture for industrial IoT, Appl. Sci. 9 (18) (2019), 3740.
[9] M. Crosby, Nachiappan, P. Pattanayak, et al., Blockchain technology: beyond Bitcoin, Berkley Eng. (2) (2016) 6–19.
[10] V. Buterin, On public and private blockchains, Ethereum Blog (2015). Available Online, https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/.
[11] Z. Zheng, S. Xie, H. Dai, et al., An overview of blockchain technology: architecture, consensus, and future trends, in: Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress; 25–30 Jun 2017; Honolulu, HI, USA, IEEE, Piscataway, NJ, USA, 2017, pp. 557–564.
[12] J. Yli-Huumo, D. Ko, S. Choi, et al., Where is current research on Blockchain technology? - a systematic review, PLoS One 11 (10) (2016), e0163477.
[13] N. Kshetri, Can Blockchain Strengthen the Internet of Things? IT Prof. 19 (4) (2017) 68–72.
[14] A. Dorri, S.S. Kanhere, R. Jurdak, Towards an optimized blockchain for IoT, in: Proceedings - 2017 IEEE/ACM 2nd International Conference on Internet-Of-Things Design and Implementation, IoTDI 2017 (Part of CPS Week); 18–21 Apr 2017; Pittsburgh, PA, USA, IEEE, Piscataway, NJ, USA, 2017, pp. 173–178.
[15] M. Conoscenti, A. Vetrò, J.C. De Martin, Blockchain for the Internet of Things: a systematic literature review, in: Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA; 29 Nov–2 Dec 2016; Agadir, Morocco, IEEE, Piscataway, NJ, USA, 2016, pp. 1–6.
[16] S. Huh, S. Cho, S. Kim, Managing IoT devices using blockchain platform, in: 19th International Conference on Advanced Communication Technology, ICACT; 19–22 Feb 2017; PyeongChang, Republic of Korea, IEEE, Piscataway, NJ, USA, 2017, pp. 464–467.
[17] L. Wu, X. Du, W. Wang, et al., An out-of-band Authentication scheme for Internet of things using blockchain technology, in: 2018 International Conference on Computing, Networking and Communications, ICNC; 5–8 Mar 2018; Maui, HI, USA, IEEE, Piscataway, NJ, USA, 2018, pp. 769–773.
[18] M. Samaniego, U. Jamsrandorj, R. Deters, Blockchain as a service for IoT, in: Proceedings - 2016 IEEE International Conference on Internet of Things; IEEE Green Computing and Communications; IEEE Cyber, Physical, and Social Computing; IEEE Smart Data, iThings-GreenCom-CPSCom-Smart Data; 15–18 Dec 2016; Chengdu, China, IEEE, Piscataway, NJ, USA, 2016, pp. 433–436.
[19] J. Balasch, B. Ege, T. Eisenbarth, et al., Compact implementation and performance evaluation of hash functions in attiny devices, in: S. Mangard (Ed.), CARDIS 2012: Smart Card Research and Advanced Applications, Springer, Berlin, Heidelberg, Germany, 2012, pp. 158–172.
[20] B. Jungk, L.R. Lima, M. Hiller, A systematic study of lightweight hash functions on FPGAs, in: 2014 International Conference on Reconfigurable Computing and FPGAs (ReConFig14); 8–10 Dec 2014; Cancun, Mexico, IEEE, Piscataway, NJ, USA, 2014, pp. 1–6.
[21] J.-P. Kaps, P. Yalla, K.K. Surapathi, et al., Lightweight implementations of SHA-3 finalists on FPGAs, in: D.J. Bernstein, S. Chatterjee (Eds.), Progress in Cryptology – INDOCRYPT 2011 vol. 60, Springer, Berlin, Heidelberg, Germany, 2012, pp. 1–17.
[22] S. Underwood, Blockchain beyond bitcoin, Commun. ACM 59 (11) (2016) 15–17.
[23] Q.H. Dang, Secure Hash Standard (SHS), Federal Information Processing Standards (FIPS-180-4), National Institute of Standards and Technology, Gaithersburg, MD, USA, 2015, https://doi.org/10.6028/NIST.FIPS.180-4.
[24] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche, Keccak sponge function family main document 3 (30) (2009) 320–337. Submiss. to NIST (Round 2).
[25] J. Guo, T. Peyrin, A. Poschmann, The PHOTON family of lightweight hash functions, in: P. Rogaway (Ed.), Advances in Cryptology—CRYPTO 2011, Springer, Berlin, Heidelberg, Germany, 2011, pp. 222–239.
[26] H. Cheng, D. Dinu, J. Großschädl, Efficient implementation of the SHA-512 hash function for 8-bit AVR microcontrollers, in: J.L. Lanet, C. Toma (Eds.), SECITC 2018: Innovative Security Solutions for Information Technology and Communications, Springer, Cham, Switzerland, 2018, pp. 273–287.
[27] X. Wang, X. Zha, W. Ni, et al., Survey on blockchain for Internet of things, Comput. Commun. 136 (2019) 10–29.
[28] N.T. Courtois, M. Grajek, R. Naik, Optimizing SHA256 in Bitcoin mining, in: Z. Kotulski, B. Księżopolski, K. Mazur (Eds.), CSS 2014: Cryptography and Security Systems 448, Springer, Berlin, Heidelberg, Germany, 2014, pp. 131–144.
[29] L.V.T. Duong, N.T.T. Thuy, L.D. Khai, A fast approach for bitcoin blockchain cryptocurrency mining system, Integration 74 (2020) 107–114.
[30] G. Bertoni, J. Daemen, M. Peeters, et al., Keccak, advances in cryptology—eurocrypt 2013, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques; May 2013; Athens, Greece, Springer, Cham, Switzerland, 2013, pp. 313–314.
[31] B. Guido, D. Joan, P. Michaël, V.A. Gilles, Cryptographic sponge functions [Online]. Available, http://sponge.noekeon.org/, 2011.
[32] M.J. Dworkin, SHA-3 standard: permutation-based hash and extendable-output functions, National Institute of Standards and Technology FIPS-202 (2015).
[33] D. Steinsland, keccak, GitHub repository (2015). Available Online, https://github.com/davidsteinsland/keccak.
[34] J. Guo, The PHOTON family of lightweight hash functions [Online]. Available, https://sites.google.com/site/photonhashfunction/downloads, 2013.
[35] K.A. McKay, L. Bassham, M.S. Turan, et al., Report on lightweight cryptography, NIST DRAFT NISTIR 8114 (2016).
[36] P.M. Mukundan, S. Manayankath, C. Srinivasan, et al., Hash-One: a lightweight cryptographic hash function, IET Inf. Secur. 10 (5) (2016) 225–231.