Quantum Aware Distributed Ledger Technology for Blockchain-Based IoT Network



Koustav Kumar Mondal and Deepsubhra Guha Roy

Abstract A digital ledger which is based on distributed technology will help to address cybersecurity and secrecy problems in the Internet of Things Architecture, but combining these two technologies poses some difficulties. In order to preserve a tradition of financial transactions, cryptocurrencies introduced DL technology. The DL size in a cryptocurrency addresses hundreds of GBs, while the storage of IoT nodes is limited. Similarly, cryptocurrencies implement costly processes of consensus, while IoT nodes in calculation and energy are restricted. Moreover, classic distributed ledger technology (Bitcoin) are not based on quantum. The chapter aims a distributed ledger security based on quantum technology, specifically distributed ledger for Internet of Things, for IoT architectures. One cornerstone of the chapter is a new signature creation scheme which is named as Single time signature based on time(STS), Blockchain-STS, a compact scheme. Compared with the famous Winternitz-OTS+ system Blockchain-STS offers a 75% reduction in signature size and a 76% reduction in signature generation time.

Keywords Quantum cryptography · Distributed Ledger (DL) technology · Internet of Things (IoT) · Post quantum digital signature · Cryptocurrency · Cloud computing

1 Introduction

IoT uses embedded technology for connecting and communicating material things. As a massive amount of material things are interconnected, a large amount of data is therefore generated. Maintenance of information protection and privacy remains a

K. K. Mondal
School of Computational Science, Department of IT, Maulana Abul Kalam Azad University of Technology, Simhat, Haringhata, Nadia, Kolkata, West Bengal 741249, India

D. Guha Roy (✉)
Mobile and Cloud Lab, Institute of Computer Science, University of Tartu, Ulikooli 17-324, 50090 Tartu, Estonia

© The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd. 2022

D. De and S. Bhattacharyya (eds.), *Blockchain based Internet of Things*,
Lecture Notes on Data Engineering and Communications Technologies 112,
https://doi.org/10.1007/978-981-16-9260-4_11

tough job for IoT Architecture [1]. The "Things-Network" framework signifies a four-stage design, Sensor Layer, Networking Layer, Service (Blockchain or Cloud) Layer, and an Interface Layer (applicability programs) [2]. The Sensing Layer incorporates devices (actuators and sensors) integrated into the physical objects for communication. The sensor data are transmitted to S3 data storage of cloud through the Layer of Network. S3 Storage stores information and enables apps to start their processing the information in order to make human and living being more manageable. Figure 1 demonstrates layered IoT-based systems architecture [3]. In general, IoT devices lack storage, energy (battery life), and computing power. In general, sensor-formed data is collected in Clouds. Nevertheless, stability, privacy, and data reliability challenges arise in IoT systems [4, 5]. The transfer of large volumes of information to the Cloud causes an overload of network sources. Likewise, a malicious Cloud can damage data protection and privacy [6].

The corresponding IoT systems could be pushed out of operation to an idle status [7]. Distributed ledgers (DLs) present an appropriate cloud service alternative for IoT systems [8]. Instead of collecting information on a primary server, matches themselves store information. Each match controls a database nearby, although modifications to the data are done through mutual agreement between the peers [9]. The model of the information on each match is synchronized at all times. Although a

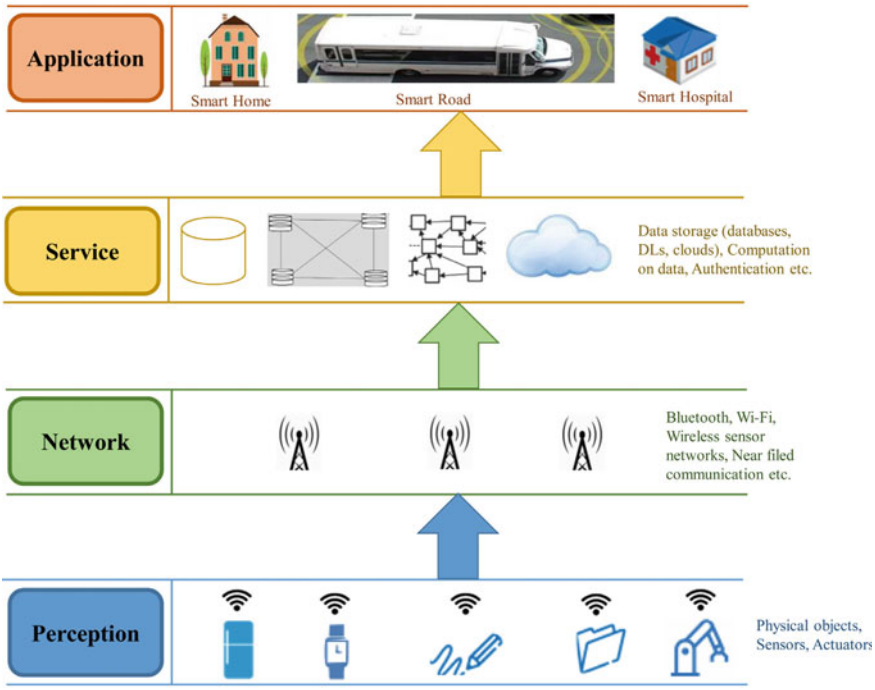


Fig. 1 Four-layered architecture of Internet-of-Things based working system

number of blockchain (DL) IoT systems are available, such as [10, 11], etc., there is still no widely accepted framework [7, 12]. The DL was initially suggested by cryptocurrencies (Bitcoin) to preserve a history of economic transactions. Nevertheless, it creates unusual difficulties by adopting DL technology for IoT-based operations. First of all, transactions are heavily linked to an input/output relationship in a cryptocurrency. Any new transaction would redeem the coins before locked by a traditional transaction. Though the transaction contents differ from the cryptocurrency in an IoT setting. A model of an IoT performance is the data produced by a sensor. The linkage of transactions of IoT to a legitimate string is. Therefore, a problem for IoT-based DLs [4]. Second of all, in a cryptocurrency, the size of a DL usually is quite big (as an example, the current volume of the Bitcoin ledger is around 200 GB), but IoT connections are restricted into stock. Third, DLs implement a computer-based cost-effective consent technique like ethereum's consensus algorithm (Proof-of-Work) to enable new operations or data block to be added to the repository. As IoT devices are energy and computation-limited, they are not suited to PoW-like consent mechanisms [13]. In one end, the Digital signature design applied through the classic DLs (such as Ethereum, Bitcoin) is the elliptic curve, which is not a quantum-safe algorithm [14]. The famous Shor algorithm allows an opponent to break ECDSA by a quantum computer that is sufficiently efficient [15]. DL technology will revolutionize IoT, but the convergence of two technologies presents challenges [7, 16]. Before the combination of the pair technologies, the related problems need to be carefully regarded. These difficulties are due to IoT gadgets that do not have the required calculation, communication, storage, and energy [5]. While blockchain-based IoT applications become suggested, such as [5, 10, 13, 17], it remains, however, an emerging research field without a widely accepted framework [7]. IoT systems cannot hold loads of consensus-like PoW structures. In addition, IoT devices cannot support a constantly increasing leader scale [18]. Finally, the digital signature system applied through the standard DLs is not immune to quantity [14]. For IoT-based systems, we discuss a quantum-secure DL, Blockchain-for-IoT, in this chapter. The DL proposed, recognizes, and discusses various issues related to Distributed Ledger technology adoption for IoT methods. We tackle four trials: Quantum Resilient Signature (QR), Scalability Ledger (LS), Transaction Chaining (TC), and Lightweight Consensus (LWC). This chapter developed the aimed Blockchain for IoT using a new STS scheme, Blockchain-STS, a lightweight, energy-efficient, and fast STS.

The chapter was structured as follows: Part 2 comprises the corresponding work. Part 3 outlines our design frame. Part 4 contains the STS scheme for DL construction. In Part 5, a quantum-safe DL for IoT operations is discussed, and Parts 6 and 7 assess the safety and success of our plans. Lastly, Part 8 ends with valuable guidelines for the future study.

2 Related Work

This portion comprises two paragraphs. Section 2.1 offers a description of DL technology-incorporating IoT applications. Section 2.2 gives a summary of current SHA384 hash-based STS technique's.

2.1 Distributed Ledger-Based Internet of Things Network

This section gives a provisional overview of research integrated with IoT technology by blockchain/DL. The DL technology extends well beyond cryptocurrencies [19]. Besides cryptocurrencies, in the development of intelligent agreements, Access Control Systems [20], healthcare, e-business, applications [21], VANETS [22], smart homes, intelligent cities [23], and so on, DL technology was also used. The Smart Access Control System [20] from Blockchain intends a design for IoT device management. The resourceful link, the management center, on the recommendation of the restricted IoT nodes [24], communicates with the Blockchain (i.e., sensor). An operator node creates the intelligent deal. The research does not have a mechanism for cutting boards and does not recommend any protection against quantum threats. Lightchain [13] aims at multiple synergistic proofs (SMP), a slight consent mechanism, and a UBOF, a ledger pruning mechanism. Lightchain [13] proposes an SMP. However, no ordinance for binding transactions into an internal or external exchange have been suggested or any protection upon quantum attacks. IOTA [4] is an intelligent cryptocurrency that serves the economy of machine-to-machine (M2M). In contrast to IOTEX, IOTA eliminates the costly technique's of consent (i.e., PoW). The Blockchain algorithm that IOTA uses is called the Tangle. A cutting mechanism is implemented to keep the leader size from continuously increasing. IOTA is using a secure signature system based on quantum technology (i.e., Winternitz OTS+). DL's intelligent house [23] concurrently uses two records, a regional immutable (IL) ledger and a public Blockchain [25]. The study does not propose any ledger cutting mechanism or quantity-attack protection. Lee and Lee suggested a Blockchain-based clarification to update embedded IoT device's firmware in a safe manner [26]. Each request for firmware upgrade erases a current block in Blockchain. The aimed leaflet has a computationally costly consensus mechanism and no leaflet cutting device. Blockchain-based VANETs enable VANETs to be increased and safety event messages disseminated efficiently [22]. The Blockchain [22] suggested is a local Blockchain that stores road cases information for the particular geographical area. The introduced Blockchain uses a costly consent system based on PoW and proposes no protection from quantum threats. An IoT-based E-business model was introduced by Zhang et al. using Blockchain technology [21]. The model proposed does not resolve the challenges of combining DL technology and IoT, such as the growing size of the leads and a highly costly consensus process [27, 28]. The latest Blockchain-based IoT purposes are summarized in Table 1.

Table 1 Blockchain-based IoT application's summary

Ref.	IoT Application	Transaction type	Smart contract	Mining	Sig. scheme	QR[a]	IoT-challenges		
							LS[b]	TC[c]	CM[d]
[12]	Access control	Triggered operation	Allowed operations	Miner nodes	ECDSA	✗	✗	✗	PoC
[7]	IIoT	Sensor data	✗	ECS	ECDSA	✗	UBOF	✗	SMP
[8]	Smart home	Tiers communication	✗	CH	Non QR	✗	✗	√	DT[e]
IoTA [3]	Cryptocurrency	Financial transaction	✗	All peers	WOTS	√	√	√	CC[f]
[13]	Firmware update	Update request	✗	IoT nodes	ECDSA	✗	✗	✗	PoW
[9]	VANETS	Safety events	✗	Vehicles	Non QR	✗	CB[g]	✗	PoW
[16]	E-business	Financial transaction	Trade deals	Miners	ECDSA	✗	✗	√	PoW

[a] Quantum resilient.
[b] Ledger scalability.
[c] Transaction chaining.
[d] Consensus mechanism.
[e] Distributed trust.
[f] Central coordination.
[g] Country-based.

2.2 SHA384 Hash-Based STS Technique

A well-known researcher Lamport Diffie suggested the signature sizes and comprehensive key for the first STS system. The Winternitz-OTS scheme (WOTS) provided reductions in main and signature sizes of 87% and 75% compared with the Lamport scheme. WOTS PRF and WOTS+ [11] are two lightweight WOTS alternatives that give a reduction in the signature size of approximately 33% in comparison with WOTS. HORS is an effective OTS system with minimal signature size. The major scale of HORS is, however, virtually bigger. Tree HORS (HORST) [10] compromise key size execution speed. PORS/PORST [29] are safer HORS/HORST variants.

2.3 IoT-Related Emerging Trends In Research

Apart from IoT DLs, several of the emerging IoT analysis trends contain adopting Green IoT communications designs using the 5G networks [30], Industrial surveillance applying large Data Analysis [17, 31], Big Data Analytics design of small businesses [32], modeling and optimizing social Internet selection features [33] defining the use of bi-based human behavior [34].

3 Discussion on System Model

The framework template proposed includes four layers (Fig. 2), Layer of Perception, Layer of Communication peer to peer, Network Layer based P2P technology, and Blockchain layer. The layer of perception repose of driving/sensing gadget correlated to everyday household appliances(such as fridge, smart clock, bicycle), which enable them to interact. The layer of perception tools have minimal calculation and storage capacities [35]. The P2P layer is made up of peers that are capable of computer connections (like laptops or PCs). The pairs remain linked to each other and

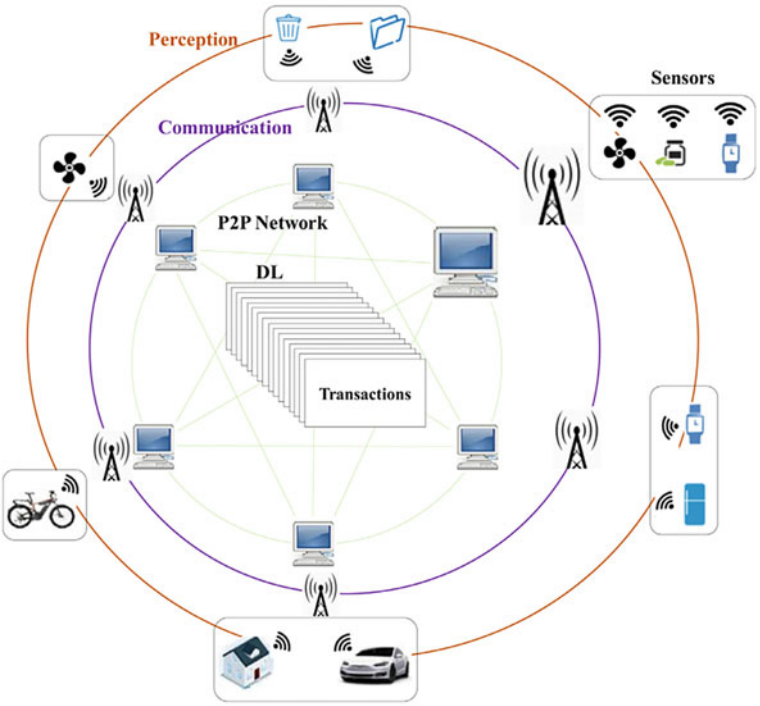


Fig. 2 System architecture model for Blockchain-based Internet of Things

thus form the P2P network. Individual peer has sufficient storage and computing capability to control a Distributed Directory (DD). The layer of communication provides communication among sensor peers and devices. Lastly, the records storage facilities are provided in the DL Layer. The matches preserve information on behalf of actuators and sensors in the aimed Blockchain-based IoT system [36]. Several actuators and sensors are attached to each peer. When information from one of the relevant sensors is received, a match creates a new deal and conveys it across the P2P network. Different peers validate the performance and collect the new deal in the ledger after reasonable verification (DL). We suggest a method that will assure that a new transaction can be stored in DL only by trusted peers. For a DL integrated into an IoT context, we have developed new transaction verification rules. The aimed transaction confirmation laws allow the DL to connect input/output deals. Therefore, each transaction has precisely one predecessor and one successor, "zero or one." To succeed in an actual transaction a, the Signs of b must validate a key earlier collected in a. The transaction owner b (Owner B) may only sign the transaction later than the transaction owner a (Owner A). In this process, Owner A serves as Owner B's sponsor. The consent given through the owner of an earlier collected transaction for each new transaction removes the need for a traditional (PoW) process. Therefore, we need not encapsulate single transactions within blocks, so our introduced DL

is a transaction store, not a block set. Specific transaction collects a slipping key so that the transaction owner can delete the activity until it is no longer necessary. The cutting key enables matches to periodically cut the record by removing wasteful activities following the agreement of respective admins. Matches are capable data nodes, and individual peer which connected to node locally collects the booklet. Nevertheless, a peer with unusual storage ability is not allowed to store the booklet locally. Since our modeling offers a board cutting mechanism, which enables pairs to eliminate unnecessary transactions safely, the board is therefore compact enough to store a pair with a standard storage capacity when needed.

3.1 Basic Elements in System Model

In our device model, there are three essential elements:

1. Physical items and embedded sensors (such as vehicles, refrigerators, portals, coffee makers).
2. The connectivity medium for transferring information amongst physical items and Computing Nodes.
3. The Network which have peer to peer technology capable of Computing Nodes, Hereabouts we describe the working function of every entity individually.

- **Embedded Sensors/Actuators Physical Things:** The layer of perception of the IoT architecture is clear. The built-in sensors transform physical environments into appropriate digital types. Digital signals are transformed by actuators into physical action. For example, an in-car sensor shows a digital atmospheric temperature and the actuator drives the coffee maker when a computer gets Digital Signals. Simply put, the actuators and sensors enable material objects to feel, interact, and carry out activities to help people live.
- **Communication Medium:** The communication medium serves the IoT architecture network layer. Method of communication includes devices that enable data transmission between computing nodes and sensors/actuators. Communication equipment can contain BLE, Wireless Fidelity, NFC, and WSN.
- **P2P Network:** a Peer-to-Peer network is a system of capable computer data nodes that implement two-tier Internet of Things architectural services, i.e., the Service and Application Layers. Matches (computer connections) obtain and method data from physical objects and take adequate measures to meet user conditions. The pairs manage a DL to collect the information transmitted through the sensors. Every peer keeps a model of the ledger. The safety and authenticity of the ledger are guaranteed by a new STS, i.e., Blockchain-STS. The sensor data is collected as transactions in the ledger. The effective peer generates a new activity and conveys it to another peer when a sensor transmits data. All other pairs check transactions (using the introduced STS scheme) and later collect them in the leader. When a current transaction shows that the machinery part should be started, the analogous peer acts his role and instates the relevant machine.

4 STS Scheme

Our suggested Blockchain-for-IoT is built on a hash-based single time signature structure. This segment describes in detail the introduced STS system. Our stated SS design has been denominated Blockchain-STS, the compactest and most successful scheme compared to all present STS plans. This sector will be restricted to the Blockchain-STS characteristics, while in Sect. 7, we will compare Blockchain-STS with current standard STS systems. Blockchain-STS is a fourfold version (key generation, signature, check, and key-compression). Key generation allows an input protection parameter (n) and delivers a key combination (sk, pk). Sign acquires the information (m) as the information furthermore delivers m signatures, i.e., σ^m with the private key(sk) Verify accepts input (m), $m(\sigma^m)$ signatures and the public key (pk), returning one of both outputs; unless prosperous or abandoned. Keycompress cryptographically compresses the public key.

4.1 Key Generation

This paragraph discusses the process of key creation toward our introduced STS system, Blockchain-STS. Within hash-based STS programs, several values are the key and the signature (denoted as l). The key and the signature at Blockchain-STS is a total of 17 values, 16 of which match the possible alphabets of a message hash to signify contracted, i.e., 0, 1, 2 and 17 for a checksum (quite similar WOTS including its alternatives utilize checksum). In addition, in STS systems, several iterations of hash (w) require the transition of an individual private key (sk) into a specific public key (pk). Each sk-value (except for the 17th value) of Blockchain-STS is 47 times (we decided 47 based on our analyses in the design of Blockchain-STS) for the corresponding pk-value to be produced. For "l om w" times (we refer to it as w c), the 17th sk-value (it is utilized for MD5 Checksum) hashed toward the identical pk-value. Blockchain-STS recommends that each sk imports are produced from one fundamental importance called a handshake. We suggest using SHA384 for an increasing the security using post-quantum technology. In the entire post, we utilize H to relate to the SHA384 hash purpose. Table 2 illustrates the various symbols in this section and the following pages.

$$\sum_{i=1}^{l} sk_{i-1} = H^i(seed)$$

$$\left[\sum_{i=0}^{l-2} pk_i = H^w(sk_i)\right] \cup [pk_{l-1} = H^{w_c}(sk_{l-1})]$$

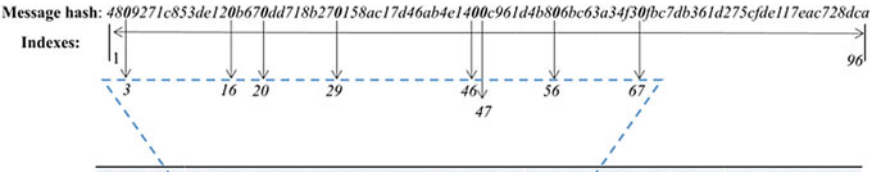
Table 2 Mathematical symbols and their description

Symbol	Description
l	No. of values in key/signatures
n	Bit-length of an individual key/signature value
h_m	Hash of the message to be signed
l_k	No. of values in key
w	No. of hash iterations used to transform an sk-value to the corresponding pk-value
w_c	Number of hash iterations used to transform the last sk-value (used for checksum) to the corresponding pk-value
sk_i	An individual value in private key
pk_i	An individual value in public key
PK	The compressed public key
H	Hash function SHA384
$H^n(m)$	SHA384 applied on m for n times
σ^m	Signatures created on message m
vk	Verification key computed by the verifier from the signatures (to be compared with pk)
trx_i	i^{th} transaction stored in the ledger
$prnKey$	Pruning key (a PK to authorize a peer to remove an un-necessary transaction from DL)
$succID$	The ID of the transaction trx_i stored in the transaction trx_{i-1}
$\mathcal{F}$	A forger who accepts the challenge to break DL-OTS
$\mathcal{A}$	An adversary who accepts the challenge to break oneway-ness of H
$m^{\mathcal{Q}}$	Message queried by the $\mathcal{F}_{DL-OTS}$ during the attack process
$m^{\mathcal{F}}$	Message returned by the $\mathcal{F}_{DL-OTS}$ at the end of the attack process

4.2 Signature Creation

Every method of development of signatures starts with a hash of the signed message (i.e. $h_m = H(msg)$). In its hexadecimal representation, we process message-hash. Since we use SHA384, h_m is comprised of 96 hexadecimal symbols (Eq. 4.2). The symbols are indexed as one to ninety-six. Then we group the records according to their symbols, as follows: The index set containing symbol 0; the index set containing symbol 1; up to the index set containing symbol F (Algorithm 1 steps 1–3). Then, in each of the 16 sets, we calculate a total of 16 integer values (steps 4–5). We utilize module operators to ensure all 17 values range from 1 to w (step 6). We calculate a check-sum for the above sixteen values in the next level: we extract each of the 17 w values and summarize all those differences (steps 7–8). Subsequently, we produce message signatures by computing post-images with the initial value "l-1", numbers of terms the identical integer value (step 9). However, the ultimate sign component is created by calculating the post-image number of times the check-sum of the lth sk-element (step 10). Figure 3 illustrates the process of creating signatures with an example.

$$h_m = \sum_{i=1}^{96} m_i = \{m_1, m_2, ..., m_{96}\}$$



Symbol (m)	Set of Indexes containing symbol "m" (ind_m)	$Sum(ind_m)\%48+1$ ($value_m$)	Signatures (σ_m)
0	316202946475667	21	$hash^{21}(sk_m)$
1	7142430354451768586	36	$hash^{36}(sk_m)$
2	515277892	47	$hash^{47}(sk_m)$
3	1161636674	42	$hash^{42}(sk_m)$
4	13842455364	46	$hash^{46}(sk_m)$
5	103180	14	$hash^{14}(sk_m)$
6	183950576075	9	$hash^{9}(sk_m)$
7	61923283671798791	42	$hash^{42}(sk_m)$
8	2925325593	46	$hash^{46}(sk_m)$
9	449	18	$hash^{18}(sk_m)$
a	3340628996	3	$hash^{3}(sk_m)$
b	17264154586973	21	$hash^{21}(sk_m)$
c	834485970819095	33	$hash^{33}(sk_m)$
d	122122375272778394	27	$hash^{27}(sk_m)$
e	13438488	40	$hash^{40}(sk_m)$
f	656882	36	$hash^{36}(sk_m)$
16		$\Sigma(48 - value_m) = 481$	$hash^{481}(sk_m)$

Fig. 3 Creation of the Blockchain-STS signature for an example 'Good Message 707070'

Algorithm 1: Blockchain-STS. sign().

Input: $h_m, [sk_0, sk_1, ..., sk_{l-1}]$
Output: $[\sigma_0, \sigma_1, ..., \sigma_{l-1}]$

1: $\sum_{i=0}^{l-2} \left[ind_i \leftarrow \phi \right]$
2: $j \leftarrow 1$
3: $\forall x \in h_m \bullet \left[\left[ind_x \leftarrow ind_x \cup j \right] \wedge \left[j \leftarrow j+1 \right] \right]$
4: $\sum_{i=0}^{l-2} \left[value_i \leftarrow 0 \right]$
5: $\sum_{i=0}^{l-2} \left[\forall d \in ind_i \bullet (value_i \leftarrow value_i + d) \right]$
6: $\sum_{i=0}^{l-2} \left[value_i \leftarrow (value_i \% w + 1) \right]$
7: $checksum \leftarrow 0$
8: $\sum_{i=0}^{l-2} \left[checksum \leftarrow checksum + (w - value_i) \right]$
9: $\sum_{i=0}^{l-2} \left[\sigma_i \leftarrow H^{value_i}(sk_i) \right]$
10: $\sigma_{l-1} \leftarrow H^{checksum}(sk_{l-1})$

4.3 Verification of Signature

This verifier shall also take Algorithm first steps (1–8) in the signature verification to calculate the 16 conditions and the checksum. Subsequent the verifier can calculate the quantity of "w times" (each identical content) after each of the first "$l-1$" $\sigma - elements$. Verifier will also retrieve the post image number of times for the $lth - element, w_c - checksum$. The checker thus calculates total l values; we mark this collection of conditions as the check key [vk] (Eq. (4.3)). Ultimately, several calculated values will be compared to each identical pk-value. If the total conditions are the same, signatures (Eq. (4.3)) shall then be accepted by a verifier; otherwise, the signatures would be refused and null.

$$\left[\sum_{i=0}^{l-2} H^{w-value_i}(\sigma)\right] \cup [vk_{l-1} = H^{w_c-checksum}(\sigma_{l-1})]$$

$$\left[\sum_{i=0}^{l-1} vk_i == pk_i\right] \Rightarrow \text{“}Accepted\text{”}$$

4.4 Key Compression

The simple public key comprises l conditions, all 384-bit in length. We utilize the algorithm of Merkle hash tree to compact the available key to simply one long 384 -bit value. Figure 4 explains the compression tree structure, while Algorithm 2 explains in-depth the primary compression process. The simple pk consists of a total of seventeen hash values. Blockchain-STS utilizes a pure hash tree from Merkle to compress a single pk. Step 1 determines array N of height 4. Here, N is 31. In a perfect binary tree, there are 16 leaf nodes (of length 4). Step 2 reserves the values of definite pk in every N leaf node. The definite pk is formed by seventeen conditions, 16 of which remain consolidated under the N leaf nodes (the 17th value is integrate and hashed, including the node N which is a root node). At step 3 starts with a new variable j to run the outward circle (steps 4–9). The outward circle emphasize to the height of every tree for many cycles. Individually, renewal of the outer circle produces a novel overhead tree level. The internal loop (steps 6–8) is equivalent to the corresponding stage's node for several cycles. Each inner loop iteration calculates a new tree node. We concatenate the two descendant nodes and calculate their hash to calculate a new parent node (step 7). Ultimately, in step 10, the tree root (N) is linked together among individual ending pk-values; furthermore, its hash is calculated. The value calculated within step 10 is the pk (indicated as PK).

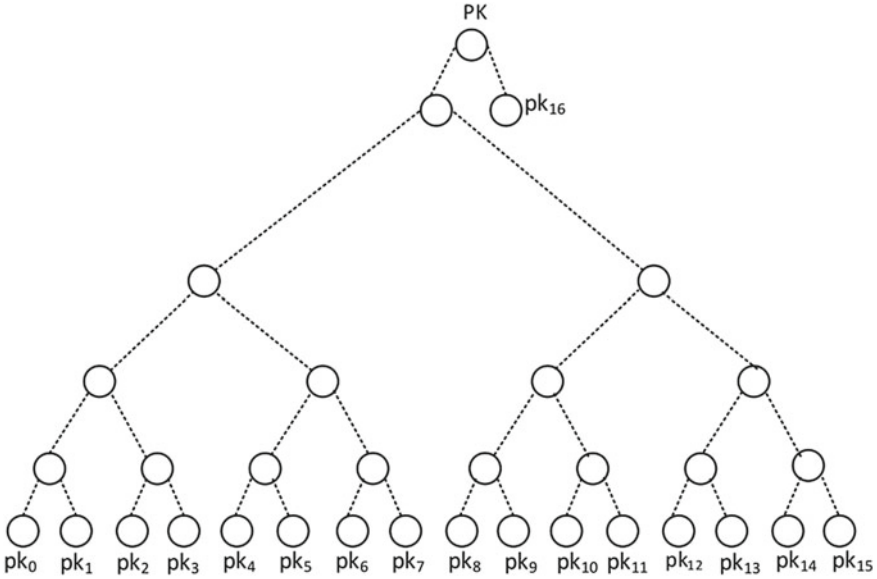


Fig. 4 Key-compression of blockchain-STS

Algorithm 2: Blockchain-STS.keyCompress().

Input: $[pk_0, pk_1, \cdots, pk_{l-1}]$
Output: PK

1: define $N[(l-1)*2-1]$ ▷ Declares a perfect binary tree N of height 4
2: $\sum_{i=0}^{l-2} N[i+l-2] \leftarrow pk_i$ ▷ Stores pk_0 to pk_{15} in leaf nodes of N
3: define $j \leftarrow (l-1)*2-1$
4: **while** $j > 1$ **do** ▷ Loop iterates 4 times, and generates an upper level of N in each repetition
5: define $k \leftarrow (j/2)$
6: **while** $k < j$ **do** ▷ Computes a new node of N in each of the repetition
7: $N[k/2] \leftarrow H(N[k] + N[k+1])$
8: $k \leftarrow k+2$
9: $j \leftarrow j/2$
10: $PK \leftarrow H(N[0] + pk_{l-1})$ ▷ Computes compressed pk from root of N and 17^{th} pk-vale

5 Discussion on Post-quantum Distributed Ledger for IoT

This portion of the chapter explains in detail our introduced Distributed Ledger-STS. The introduced Distributed Ledger includes several activities. An event contains the subsequent information objects: execution time, sensors (Internet-of-Things Informations), docking key(prnkey), digital signatures(σ), and the transaction's successor ID (succID). We will use trx i(n) to describe the transaction I data item (0 to 4). SuccID and docking key are both the public keys provided by our intended STS system (Sect. 4). The initial event (i.e., the creation of genesis block) is a succID. The owner creates the event of genesis. Different events receive the ID of their antecedent,

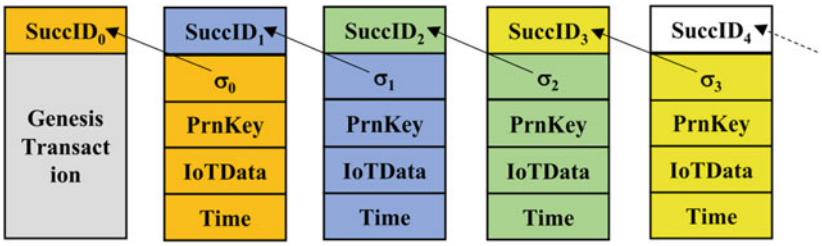


Fig. 5 Transaction structure and chaining

which signifies only a reasonable later agreement of each event keeper before it. When making the genesis block transaction(trx 0), the proprietor generates a public key utilizing our recommended STS scheme (i.e., Blockchain-STS) and save the public key in the genesis block as succID (Algorithm 3). The private key allows trx 0 owners to enable the generation of a new transaction by another node. The event proprietor of the second transaction block(trx 1) must signature the block transaction with a private key that matches the succID in trx 0. The measurement connected peers will be obey trx 1 sign to produce the public key equivalent to the succID collected in trx 0. Similarly, trx 1's succID gave permission to trx 1 owner to gave permission to different nodes to create a new event(trx 2); trx 2 keeper has to sign its business utilizing each private key corresponding to the succID collected in trx 1. Figure 5 illustrates transaction structure and chaining, and Algorithm 4 demonstrates how new transactions are created.

Algorithm 3 DL-for-IoT.*createGenesis( ).*

Input: ϕ
Output: Genesis Transaction (trx_0)
1: $trx_0[0] \leftarrow trx_0[1] \leftarrow trx_0[2] \leftarrow trx_0[3] \leftarrow \phi$
2: $(sk, pk) \leftarrow$ *DL-OTS.keyGen()*
3: $trx_0[4] \leftarrow$ *DL-OTS.keyCompress(pk)*

Algorithm 4 DL-for-IoT.*newTransaction().*

Input: sk_σ (*sk* corresponding to $trx_{i-1}[4]$), *IoT data*
Output: trx_i
1: $trx_i[0] \leftarrow$ *currentTime*
2: $trx_i[1] \leftarrow$ *IoT data*
3: $(sk_{prn}, pk_{prn}) \leftarrow$ *DL-OTS.keyGen()*
4: $trx_i[2] \leftarrow$ *DL-OTS.keyCompress*(pk_{prn})
5: $trx_i[3] \leftarrow$ DL-OTS.sign($trx_i[0]+trx_i[1]+trx_i[2]$, sk_σ)
6: $(sk_{succ}, pk_{succ}) \leftarrow$ *DL-OTS.keyGen()*
7: $trx_i[4] \leftarrow$ *DL-OTS.keyCompress*(pk_{succ})

5.1 Accepting and Verification of Transaction

This paragraph discusses the rules on transaction confirmation embraced by our suggested DL, namely Blockchain-for-IoT. The companion can attach a new transaction in our proposed model only after the consent of another peer who has confidence and has his or her transactions previously collected in the Blockchain. Several distinct transactions receive their ID from their previous activity, which can only be accomplished after the agreement of the previous transaction holders. Once a peer has

submitted a new transaction, all other partners check the novel activity (each identical algorithm five methods) and subsequently add it to the ledger. Solely trusted peers can store transactions on the ledger in our system model. A colleague with a stake in the directory is trusted (inside each class of a transaction) furthermore may concede a different colleague to supplement a new activity. The new peer is still trusted because the old trustworthy peer has approved it. The season for a wicked companion is insignificant in our suggested model. Furthermore, if a peer has a wicked deed of a remarkable kind, the subsequent peer and his supporter may be excluded by the trusted peers.

Algorithm 5 DL-for-IoT.*verifyTransaction()*.

Input: trx_i
Output: *Verified/Failed*
1: $vk \leftarrow$ *DL-OTS.verificationKey*($trx_i[0]$+$trx_i[1]$+$trx_i[2]$, $trx_i[3]$)
2: *vkCompressed* $\leftarrow$ *DL-OTS.keyCompress*(*vk*)
3: **if** *vkCompressed* = $trx_{i-1}[4]$ **then**
4: *Verified*
5: **else**
6: *Failed*

5.2 Differentiate Between Trusted and Malicious Peers

This portion of the chapter discusses how our discussed example distinguishes among a trusting plus a hostile companion. A companion that can collect a trade effectively inside the catalog is trustworthy. The transaction proprietor obligation provides the valid signatures of his trade to store a new event in the header (otherwise, the event is unverified and thus rejected). The owner signs his transaction using our suggested "DL- STS" system. The signatures must match an ID previously registered in the record. The transaction previously collected with each identical ID would be the precursor of the current transaction. The owner shall provide the proprietor of that antecedent event with a seed to allow him to sign a new transaction. The seeding from an established owner to the new owner ensures that each current proprietor is a reliable user/companion. Since distinct companions take the identical method to collect the event in the record, all companions deposited their events in the leader are trustworthy. A pair that fails to implement signatures corresponding to a cached ID is marked as a wicked combination, including its action is dismissed.

5.3 Ledger Pruning

The DL-with-IoT offers a higher scalability for the elimination of unnecessary purchases. An owner can delete his transaction until it is no longer necessary. To delete a transaction (say trx_i), trx_i owner must sign it with a private key (prnkey) already registered in trx_i to delete a transaction. Other peers would then validate the signatures and then delete the transaction concerned. For the removal of trx_i, the succID contained in it is easily copied to trx_{i-1} transaction (Algorithm 6). The succID

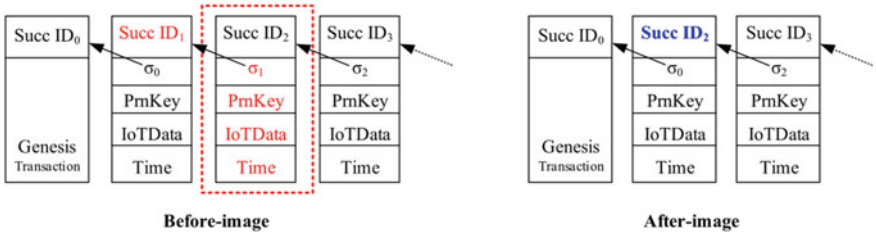


Fig. 6 Ledger-pruning technique

contained in trx I is essentially the transaction's ID, trx_{i+1}. When this succID is copied to trx_{i-1}, trx_{i+1} will be positioned as trx_i, while the old trx_i will exit the chain (ledger). Figure 6 displays DL photos before and after a transaction has been removed.

Algorithm 6 DL-for-IoT.*removeTransaction()*.

Input: σ_{trx_i} (signatures on transaction to be removed, created using the corresponding sk_{pm})
Output: DL with trx_i removed from it

```
vk ← DL-OTS.verificationKey(trx_i, σ_trx_i)
vkCompressed ← DL-OTS.keyCompress(vk)
if vkCompressed = trx_i[2] then
    trx_{i-1}[4] ← trx_i[4]
```

5.4 Working Architecture of Blockchain-for-IoT

Us comprehend some knowledges' architectures [4, 5]. In [5], the planned design includes miners, full nodes (additionally recognized as factors), including Internet-of-Things data nodes. Internet-of-Things sensor nodes create information plus stock information from miners into the box. On behalf of IoT nodes, agents commute with miners. Makhdoom et al. suggested in [4] an interface into which IoT sensor and actuator communicate explicitly with the DL nodes. In this system, both a miner and an agent share a blockchain server and Internet-of-Things nodes interact explicitly with connected peers. Peers are resourceful enough to collect and save sensor information in their blockchain ledger. Due to reasonably small transaction verification and consensus processes, we may exclude an agent's position. Our design assumes that IoT devices do not produce keys, but a connected peer conducts these action in the name of an IoT node community. Peers often retain a DL in order to save information from the related Internet-of-Things sensor [37]. When a part of information is saved in the blocks of blockchain, a peer produces two STS major pairs. Each pair keeps their keys locally. Store a key-pair means just store two SHA384 hash data, private and public key compressed; (each 384-bit). Therefore, the cumulative key size on disc is just 0.097KB. We summarise the following points in our suggested architecture:

1. Two kinds of nodes, IoT nodes and peers are available
2. Data generation of IoT nodes (i.e. sensors)

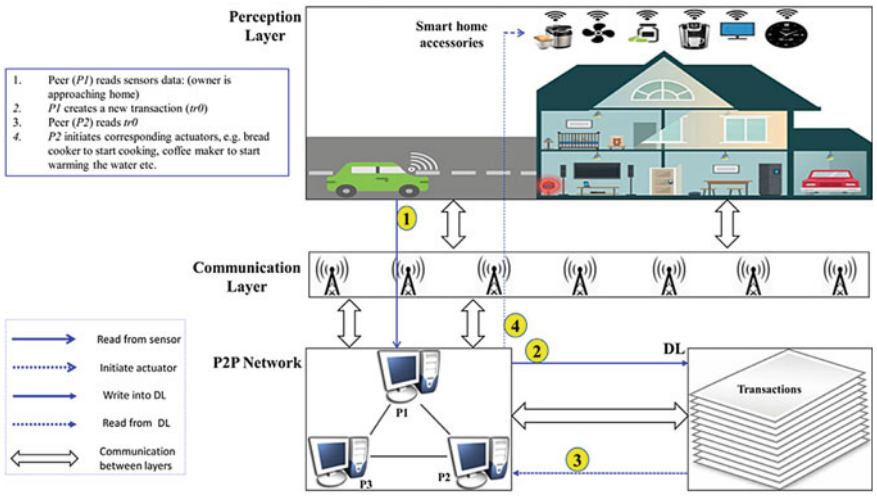


Fig. 7 A smart home architecture incorporating with blockchain-for-IoT

3. A peer has a double role; one is for a number of nodes connected in IoT system, and 2nd is that people keep the Blockchain-DL.
4. Information of IoT system is saved in the distributed ledger by connected peers. A part of information which is generated by a sensor is deposited in the Blockchain-DL by the pair operating as a sensor thing.
5. Peers build and store keys to store data for the sensors
6. Each pair keeps their keys locally

Figure 7 explains the design of our proposed Blockchain-for-IoT in a clever house.

6 Security Analysis for Distributed Ledger STS

For safe execution, Blockchain-STS is a SHA384 hash-based technique that involves a single-way hash feature. In this part, we'll see how the security of Blockchain-STS is based on the single-way of its underlying hash algorithm being reduced. The Blockchain-STS is made up of 4 parts. Key-generation accepts n as a protection variable and gives a key-pair $(\sum_{i=0}^{l-1}(sk_i, pk_i))$ A information-hash (h_m) and a private-key $(\sum_{i=0}^{l-1} sk_i)$ are needed to sign then its return message signature of $m(\sigma^m)$.

Table 3 Key and signature sizes of STS schemes

Scheme	Parameters			Sizes (KB)	
	h_m	n	l_k	key	σ
Lamport	384-**bit**	384-bit	768	36.9	18.4
WOTS		384-bit	99	4.8	4.8
WOTS^{PRF}		256-bit	99	3.2	3.2
WOTS^{+}		256-bit	115	3.7	3.2
HORS/PORS		384-bit	65536	3.1MB	1.2
DL-OTS		384-bit	17	0.8	0.8

7 Blockchain-STS Performance Analysis

This segment assesses the neatness and responsiveness of the Blockchain-STS, which is the foundation of our crafted Blockchain-for-IoT. In Sect. 7.1, the main and signature sizes of Blockchain-STS are compared to those of current STS schemes.

7.1 Blockchain-STS Compactness Assessment

Blockchain-STS has the minor key plus sign dimensions of each present single-time signature (STS) system. The primary moreover sign dimensions of Distributed Ledger-STS are compared to those of current typical STS schemes in Table 3. The extent of each information-hash to be signed (h_m), the extent of a natural key/component(n), also this whole quantity of components required to instantiate a scheme all influence the key and signature sizes (w). Those parameters enable the user to change the scheme's assurance level(the security level is determined by the parameters h_m, and n), as well as to create a trade-off between signature value size plus sign production time. To manage nearly 128-bit post-quantum protection, we chose values for h_m, and n. We have also selected the value of parameter w that recommends [11]. Compared to the most common STS system, WOTS, the findings show that Blockchain-STS provides an 84% reduction in both primary and sign dimensions. In addition, Blockchain-STS provides substantial principal and signature size reductions as opposed to other systems.

The original WOTS lightweight versions WOTS + is a standard compact WOTS version that is around 33% smaller than WOTS +. Blockchain-STS is 75% smaller than WOTS +, but Blockchain-STS is 78% smaller than WOTS + when it comes to core dimension.

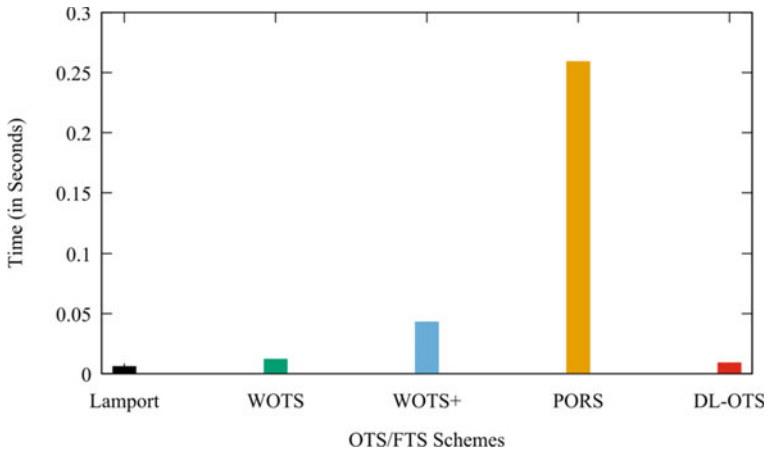


Fig. 8 Key generation time of STS

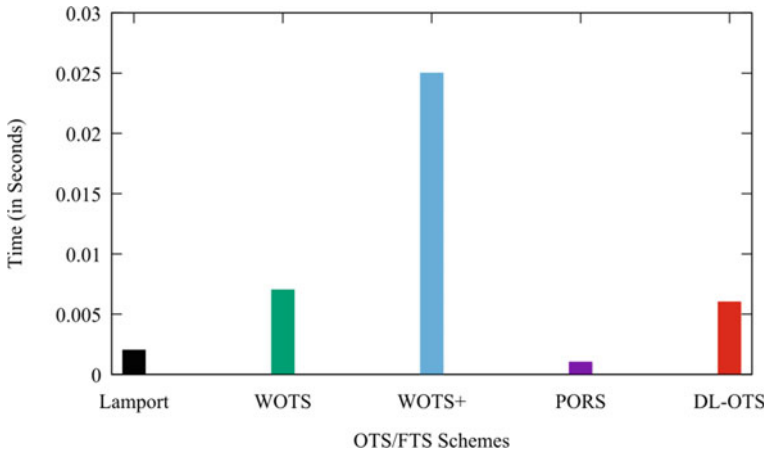


Fig. 9 Signature creation time of STS

7.2 Power and Energy Efficiency of Blockchain STS

This sub-section contrasts the Blockchain-STS implementation period and energy usage to current STS schemes. Distributed Ledger-STS is not just a lightweight but additionally an active and energy-efficient STS system that provides the least time for crucial creation, information signing, signature, and key compression [38]. The charts in Figs. 8, 9, 10 and 11 allow the execution period of Blockchain-STS to be compared with current standard STS systems. Both these systems have been introduced in the "JetBrains PyCharm Community Edition 2018.3.3" environment utilizing the Python language [39]. The findings were obtained from a laptop includ-

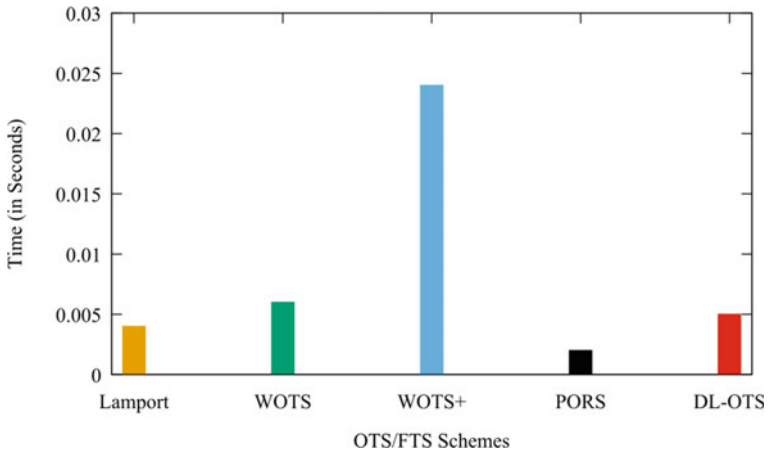


Fig. 10 Signature verification time of STS

ing an Intel Core i3 CPU (2.0 GHz) and 8GBRAM with a 64-bit version of Windows 10. All these findings show that our proposed scheme is effective. The results show that the Distributed Ledger STS production, signature formation, and main compression times are 25%, 14%, and 55% lower than the WOTS scheme. In addition, Blockchain-STS provides 79%, 76%, and 54% reductions in the primary generation, signature creation/verification, and key compression times, sequentially, associated with WOTS+. The graph in Fig. 12 relates Blockchain-STS power usage to other standard STS systems. The findings show that Distributed Ledger STS Blockchain-STS saves 47.9% of energy compared to the popular WOTS system. We also observed the study of Damasevicius et al. in calculating energy consumption [40]. SciPhone i+++ Cell Computer belongs to the testbed of the energy-related tests [41].

8 Conclusion

For IoT-based applications, Blockchain-for-IoT is a fully secured system based on the quantum technology. The Blockchain-for-IoT discussed the problems of combining the two distributing and IoT technologies. Blockchain-for-IoT is characterized by transaction-chaining, book cutting, low-weight agreement. Blockchain-for-IoT is a distributed, quantum-based DL that customizes rules for the transaction chain (as per standard specifications of IoT systems), develops a ledger pruning mechanism, and implements a lightweight consensus mechanism. The Blockchain-for-IoT building block is a new STS system, namely Blockchain-STS. Comparing Blockchain-STS among typical STS schemes, Blockchain-STS is a lightweight, fast, furthermore energy-efficient signature pattern. Compared with the famous Winternitz OTS+ schema, Blockchain-STS offers a 75% reduction in signature dimension

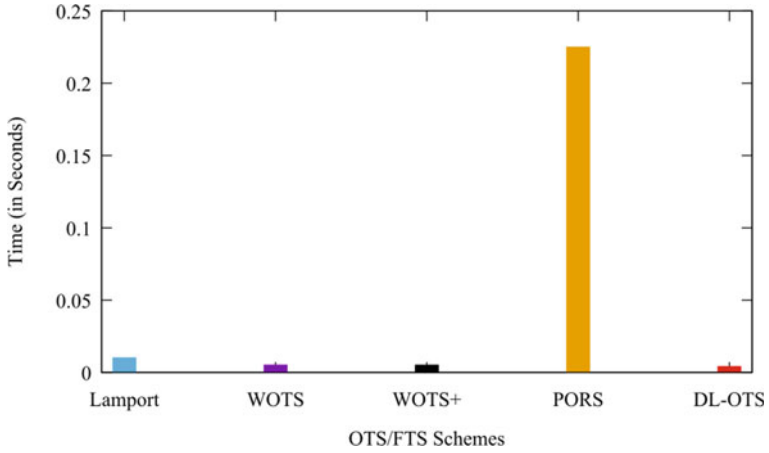


Fig. 11 Key compression time of STS

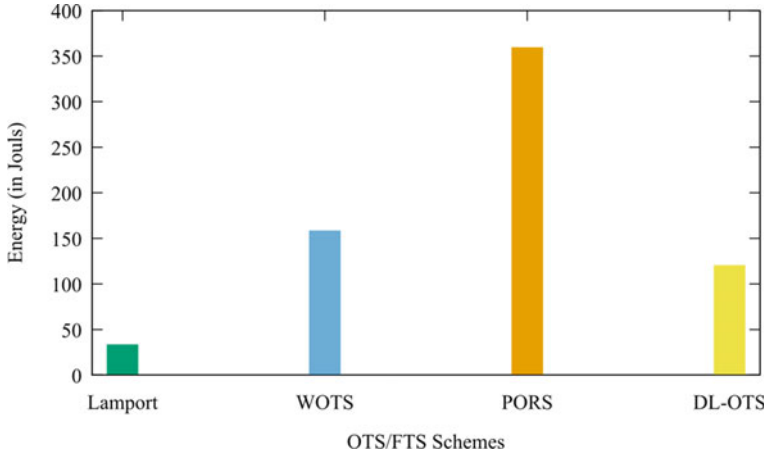


Fig. 12 Energy consumption (in Joules) of STS schemes

and 77% reduction in signature formation period. In addition, Blockchain-STS saves 47.9% more power than Winternitz-OTS +. The suggested Blockchain-for-IoT is a general architecture that can be used after customizing miners for various kinds of IoT utilization (such as smart houses, clever towns, smart grids, or VANETs, etc.). The future study emphasizes the protocols to ensure safe data transfer among IoT nodes and the P2P network. In addition, the proposed architecture can be improved in the future to accommodate complex situations such as complex index modification as several transactions occur or concurrently prune.

References

1. Khan MA, Salah K (2018) Iot security: Review, blockchain solutions, and open challenges. Futur Gener Comput Syst 82:395–411
2. Qiu T, Chen N, Li K, Atiquzzaman M, Zhao W (2018) How can heterogeneous internet of things build our future: a survey. IEEE Commun Surv Tutor 20(3):2011–2027
3. Roy DG, Mahato B, De D, Buyya R (2018) Application-aware end-to-end delay and message loss estimation in internet of things (iot)-mqtt-sn protocols. Futur Gener Comput Syst 89:300–316
4. Makhdoom I, Abolhasan M, Abbas H, Ni W (2019) Blockchain's adoption in iot: the challenges, and a way forward. J Netw Comput Appl 125:251–279
5. Qureshi KN, Din S, Jeon G, Piccialli F (2020) Link quality and energy utilization based preferable next hop selection routing for wireless body area networks. Comput Commun 149:382–392
6. Cheng C, Lu R, Petzoldt A, Takagi T (2017) Securing the internet of things in a quantum world. IEEE Commun Mag 55(2):116–120
7. Yu Y, Li Y, Tian J, Liu J (2018) Blockchain-based solutions to security and privacy issues in the internet of things. IEEE Wirel Commun 25(6):12–18
8. Wu X, Jiang G, Wang X, Xie P, Li X (2019) A multi-level-denoising autoencoder approach for wind turbine fault detection. IEEE Access 7:59376–59387
9. Chaudhary R, Jindal A, Aujla GS, Kumar N, Das AK, Saxena N (2018) Lscsh: lattice-based secure cryptosystem for smart healthcare in smart cities environment. IEEE Commun Mag 56(4):24–32
10. Bernstein DJ, Hopwood D, Hülsing A, Lange T, Niederhagen R, Papachristodoulou L, Schneider M, Schwabe P, Wilcox-O'Hearn Z (2015) Sphincs: practical stateless hash-based signatures. In: Annual international conference on the theory and applications of cryptographic techniques. Springer (2015), pp 368–397
11. Hülsing A (2013) W-ots+–shorter signatures for hash-based signature schemes. In: International conference on cryptology in Africa. Springer (2013), pp 173–188
12. Roy DG, Das P, De D, Buyya R (2019) Qos-aware secure transaction framework for internet of things using blockchain mechanism. J Netw Comput Appl 144:59–78
13. Liu Y, Wang K, Lin Y, Xu W (2019) A lightweight blockchain system for industrial internet of things. IEEE Trans Industr Inf 15(6):3571–3581
14. Aggarwal D, Brennen GK, Lee T, Santha M, Tomamichel M (2017) Quantum attacks on bitcoin, and how to protect against them, arXiv preprint arXiv:1710.10377
15. Costello C, Jao D, Longa P, Naehrig M, Renes J, Urbanik D (2017) Efficient compression of sidh public keys. In: Annual international conference on the theory and applications of cryptographic techniques. Springer, pp 679–706
16. Reyna A, Martín C, Chen J, Soler E, Díaz M (2018) On blockchain and its integration with iot. challenges and opportunities. Future Gener Comput Syst 88:173–190
17. Din S, Paul A, Ahmad A, Gupta BB, Rho S (2018) Service orchestration of optimizing continuous features in industrial surveillance using big data based fog-enabled internet of things. IEEE Access 6:21582–21591
18. Roy DG, De D, Alam MM, Chattopadhyay S (2016) Multi-cloud scenario based qos enhancing virtual resource brokering. In: 2016 3rd international conference on recent advances in information technology (RAIT). IEEE, pp 576–581
19. Underwood S (2016) Blockchain beyond bitcoin. Commun ACM 59(11):15–17
20. Novo O (2018) Blockchain meets iot: an architecture for scalable access management in iot. IEEE Internet Things J 5(2):1184–1195
21. Zhang Y, Wen J (2017) Peer-to-peer network application, vol 10, p 983
22. Shrestha R, Bajracharya R, Shrestha AP, Nam SY (2020) A new type of blockchain for secure message exchange in Vanet. Digital Commun Netw 6(2):177–186
23. Winter E, Forshaw S, Ferrario MA (2018) Measuring human values in software engineering. In: Proceedings of the 12th ACM/IEEE international symposium on empirical software engineering and measurement, pp 1–4

24. Roy DG, Mahato B, Ghosh A, De D (2019) Service aware resource management into cloudlets for data offloading towards iot. Microsyst Technol, 1–15
25. Jiang H, Zhang Z, Ma Z (2019) Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In: International conference on post-quantum cryptography. Springer, pp 227–248
26. Lee B, Lee J-H (2017) Blockchain-based secure firmware update for embedded devices in an internet of things environment. J Supercomput 73(3):1152–1167
27. Güneysu T, Oder T (2017) Towards lightweight identity-based encryption for the post-quantum-secure internet of things. In: 18th international symposium on quality electronic design (ISQED). IEEE, pp 319–324
28. Fernandez-Carames TM, Fraga-Lamas P (2019) A review on the application of blockchain to the next generation of cybersecure industry 4.0 smart factories. IEEE Access 7:45201–45218
29. Smart NP (2018) Topics in cryptology–CT-RSA 2018: the cryptographers' track at the RSA conference 2018, San Francisco, April 16–20, 2018, Proceedings, vol 10808. Springer (2018)
30. Din S, Ahmad A, Paul A, Rho S (2018) Mgr: Multi-parameter green reliable communication for internet of things in 5g network. J Parallel Distrib Comput 118:34–45
31. Mahato B, Roy DG, De D (2021) Distributed bandwidth selection approach for cooperative peer to peer multi-cloud platform. Peer-to-Peer Netw Appl 14(1):177–201
32. Gohar M, Ahmed SH, Khan M, Guizani N, Ahmed A, Rahman AU (2018) A big data analytics architecture for the internet of small things. IEEE Commun Mag 56(2):128–133
33. Ahmad A, Khan M, Paul A, Din S, Rathore MM, Jeon G, Choi GS (2018) Toward modeling and optimization of features selection in big data based social internet of things. Futur Gener Comput Syst 82:715–726
34. Liu Z, Pöppelmann T, Oder T, Seo H, Roy SS, Güneysu T, Großschädl J, Kim H, Verbauwhede I (2017) High-performance ideal lattice-based cryptography on 8-bit avr microcontrollers. ACM Trans Embed Comput Syst (TECS) 16(4):1–24
35. Roy DG, Mahato B, De D (2019) A competitive hedonic consumption estimation for iot service distribution. In: URSI Asia-Pacific radio science conference (AP-RASC). IEEE, pp 1–4
36. Guha Roy D, Srirama SN (2021) A blockchain-based cyber attack detection scheme for decentralized internet of things using software-defined network. In: Software: practice and experience
37. Roy DG, Das M, De D (2018) Cohort assembly: a load balancing grouping approach for traditional wi-fi infrastructure using edge cloud. In: Methodologies and application issues of contemporary computing framework. Springer, pp 93–108
38. Pöppelmann T, Güneysu T (2014) Area optimization of lightweight lattice-based encryption on reconfigurable hardware. In: IEEE international symposium on circuits and systems (ISCAS). IEEE, pp 2796–2799
39. Hernández-Rojas DL, Fernández-Caramés TM, Fraga-Lamas P, Escudero CJ (2018) Design and practical evaluation of a family of lightweight protocols for heterogeneous sensing through ble beacons in iot telemetry applications. Sensors 18(1):57
40. Damaševičius R, Ziberkas G, Štuikys V, Toldinas J (2012) Energy consumption of hash functions. Elektronika ir elektrotechnika 18(10):81–84
41. Roy DG, Ghosh A, Mahato B, De D (2018) Qos-aware task offloading using self-organized distributed cloudlet for mobile cloud computing. In: International conference on computational intelligence, communications, and business analytics. Springer, pp 410–424