A Robust Digital Signature Scheme for Secure and Efficient Communication for the Internet of Things

V. Antony suresh[1], M. Abinaya[2], A. Swetha[2], P. Vaitheswari[2]
Assistant Professor[1], Final Year[2]
Department of Information Technology
St. Joseph College of Engineering, Sriperumbudur, Chennai.

Abstract

Internet of Things (IoT) is a network of all devices that can be accessed through the internet. These devices can be remotely accessed and controlled using existing network infrastructure, thus allowing a direct integration of computing systems with the physical world. This also reduces human involvement along with improving accuracy, efficiency and resulting in economic benefit. IoT applications need to retrieve sensing data from the cloud for analysis and decision-making purposes. However, it is challenging to guarantee the security for the correctness and safety of IoT applications. Ensuring the authenticity and integrity of the sensing data is essential for the correctness and safety of IoT applications. More specifically, the sensing data are authenticated by two signature schemes: dynamic tree chaining and geometric star chaining that provide efficient and secure communication for the Internet of Things. Extensive simulations and prototype emulation experiments driven by real IoT data show that the proposed system is more efficient than alternative solutions in terms of time and space.

I.INTRODUCTION

The Internet of Things is the expansion of the current Internet services so as to accommodate each and every object which exists in this world or likely to exist in the coming future. This article discusses the perspectives, challenges and opportunities behind a future Internet that fully supports the "things", as well as how the things can help in the design of a more synergistic future Internet. Things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental, and user contexts [1-3]. There is several fuzziness about the concept of Internet of Things such as IoT can be broken in two parts Internet and Things. The worldwide network of interconnected computer networks based on a standard communication protocol, the Internet suite (TCP/IP) while a thing is an object not precisely identifiable [2-4]. The world around us is full of objects, smart objects and the existing service provider known as Internet. The convergence of the sensors like smart objects, RFID based sensor networks and Internet gives rise to the Internet of Things. With increased usage of sensors the raw data as well as distributed data is increasing. Smart devices are now connected to Internet using their communication protocol and continuously collecting and processing the data [5]. Ubiquitous computing which was thought as a difficult task has now become a reality due to advances in the field of Automatic Identification, wireless communications, distributed computation process and fast speed of Internet [6]. From just a data perspective the amount of data generated, stored and processed will be

enormous. The Internet of Things (IoT) developed rapidly in all fields. It has been applied to every corner of the society. Its security problems will certainly affect all aspects ofthe humankind.

To protect IoT, existing security management methods for IoT use static security defence strategies and deploy security technologies to IoT nodes. It's a traditional security defence solution for IoT. However, along with the spreading of IoT applications, the security environment of IoT maybe changes constantly. Absolutely safe IoT doesn't exist. Only static security defence strategies and security technologies for IoT are difficult to adapt to the security situation of IoT. A theoretical model which can fit the change of the IoT environment and scientifically handle security threats against IoT is urgently needed to defend IoT. The IoT is changing at fast pace and is in the process of constructing current static Internet into a fully integrated future Internet. This revolution will change the way people work, think and live life. Imagine each of the vital objects in day to day life connected to each other. For any individual his wallet and watch will themselves present an alert to the user keeping them safe. The individual will be able to keep track of his belongings from anywhere and anytime and from any network.

Since the sensing data are stored in a third-party cloud, data authenticity and integrity, which guarantee that data are from these sensing devices and have not been modified, are important for trustworthy IoT applications [14]. However the data could be corrupted by outside attackers malicious cloud employees, transmission failures, or storage loss [18]. Without data authenticity and integrity, IoT applications may make wrong decisions and cause economic and human-life losses. Authenticity and integrity should be verifiable by data applications.

The goal of this paper is to examine the security issues and challenges of IoT, and present an overview of up-to- date security solutions. This paper also review IoT data communication involving the three key entities: dynamic treechaining, geometric star chaining and onion encryption that provide efficient and secure communication for IOT

I. RELATED WORKS

Digital signature is widely used to ensure data authenticity and integrity. However, none of existing signature schemes are appropriate for the IoT scenario described in below.

First, the Sign-each method causes expensive computational cost on both signer/verifier sides owe to excessive public-key encryption and decryption operations. It is known that public-key encryption and decryption is much slower and more energy-consuming than symmetric-key encryption/decryption and cryptographic hashing. For example signing one short message using RSA with a 1024-bit k' consumes approximately 360mWs and takes about 12 seconds on one popular wireless sensor network platform, while computing SHA-1 of the same message consumesless than 1mWs [10]. Furthermore, the Sign-each method may not be able to detect data loss.

The concatenate signature scheme can amortize the signing and verification cost to multiple messages, but it is not suitable for sensing devices which may be lack of buffer space to accommodate all messages. In addition, it does not support partial data retrieval.

Hash chaining [9] reduces the buffer space complexity from O to 1 for both the signer and verifier, where m isthe number of messages buffered in the sensing device to be jointly signed. In hash chaining signature scheme, only the first message is signed and each message carries the one-time signature for the succeeding message. However, hash chaining fails when some events are dropped due

to sampling or partial data retrieval.

II. DESIGN GOALS

A. System Model

In this work, consider a cloud-based data service system composed of three entities as shown in Fig. 1, i.e., the IoT devices, cloud server, the sensing data, and the data applications. IoT devices are resource-constraint devices that generate sensing data. IoT devices are usually limited in computation, memory, and power resources. Cloud Server offers the data storage to the clients and data access to the data applications. It is a third party cloud provider who has rich resources and expertise in operating cloud computing services. Data applications are software systems that may request to retrieve the sensing data for analysis purposes. Different data applications may have varied data granularity requirements. An application may fetch all or a fraction of data from the cloud of an epoch to conduct post-processing based on their requirements.

In our system model, the IOT devices and the data applications are trusted entities. All the IOT devices are ableto sense and upload data to the cloud server through coordinator. To enable authenticity and information hiding in

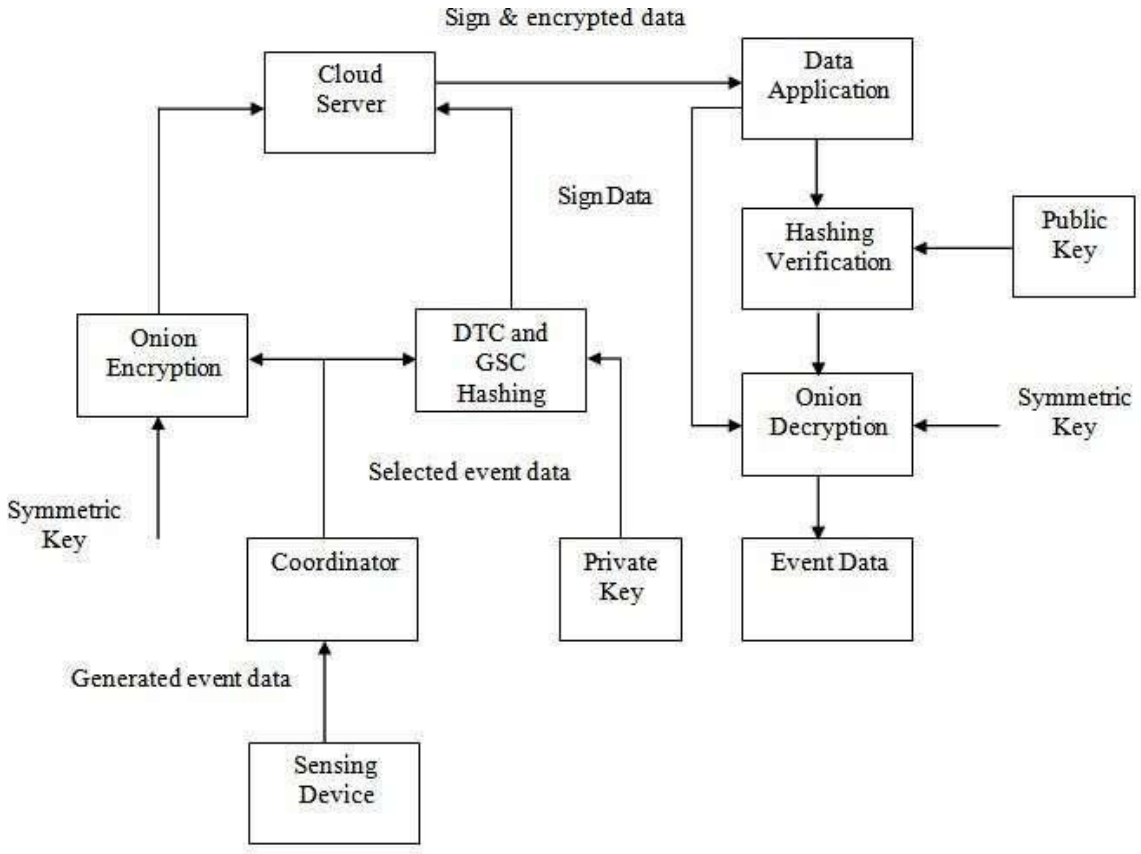


Fig 1 Flow diagram of proposed system

Remote data integrity, the generated sensing data using IoT devices is signed and encrypted over the outsourced data. To prevent the data could be corrupted by outside attackers, malicious cloud employees, transmission failures, or storage loss. Authenticity and information hiding in remote data integrity should be verifiable by data applications.

B. Data Model

IoT sensing data can be classified into two types: time series data and event data [23]. Time series data are generated by each device for every fixed time period, such as 1 second. They are used to conduct continuous monitoring tasks such as temperature reports. Event data are generated whenevera certain type of events occurs, such as a vehicle appearing in a smart camera. They are used to monitor discrete events. Here, assume IoT devices transmit sensing data to the cloud at a fixed time interval called epoch. IoT devices do not require perfect synchronization but we do assume one synchronization protocol available to loosely synchronize clocks on different IoT devices with bounded drift

C. Threat Model and Security Definitions

In our system model, the cloud server stores and manages the outsourced data and data applications can access the outsourced data using onion function. However, cloud is not trusted, which may return incorrect query results to the data applications. The main motivation of this work is to provide a verifiable evaluation scheme by which the data applications are able to validate whether the cloud server has correctly outsourced the correct data.

In this work, we consider the following potential security threatens in this work.

Data corruption. In this type of security threat, the outsourced data is corrupted. The corrupted data used as the input of delegated computation can lead to a wrong result. The adversaries could be the outside attackers or the cloud server

Incorrect result. The cloud server may not fully perform the delegated computations over the entire inputs or

Randomly output a result to save the computation resources for the monetar reasons

C. Design Objectives

The goal of this paper is to allow IoT applications to have the capability to verify the data authenticity and information hiding in remote data integrity of the stored sensing data.

Each IoT device hosts and uses its own private key in case of device compromises. We assume that there is a well-functioning PKI which manages the distribution of the public keys. We also assume that no special hardware is leveraged to use physical-layer information to boost secure communication [24]

III. VERIFIABLE COMPUTATION FOR PROPOSED MODEL

With ever-growing volume of IoT data, storing all raw IoT data in the cloud poses a on the users. Here incorporate the budget limit solution to address the issue of budget limit is compatible withDTC, GSC and Onion encryption.

A. Sampling Protocol Design

In sampling protocol introduces a new entity, called coordinator, in the network model. One coordinator is a software working as a sampler which sits between the sensing devices and the

cloud. A coordinator can be installed on an IoT hub or a server at the edge of the Internet. It maintains communications with all sensing devices on behalf of the cloud and temporarily buffers IoT data sample

The straightforward solution for heavy economic burden is to buffer all the events in the coordinator and uniformly sample them based on the budget limit. This protocol involves two algorithms at the sensing device and the coordinator respectively

Algorithm 1: SP at Sensing Device *k* in Round *j*

for each event e **do**

$i \leftarrow \min\{x \in \mathbb{N} : h(e) \geq 2^{-x-1}\};$

$l_i^k \leftarrow l_i^k + 1;$

if $i \geq j$ **then**

Forward *e* to the coordinator;

else

Discard

***e*;end**

end

1) Sensing Device:

On receiving a new event e, the sensing device first computes which numeric interval in {Si} that h(e) falls in, and updates the local counter associated with this set, where h(·) is a uniform random hashing function and ∀x : 0 ≤ h(x) ≤ 1. Let l^k_i be the local counter for S_i at device k. Each sensing device and the coordinator maintain their own local counters. The local counters at devicesare used for auditing the coordinator. Suppose h(e) ∈ S_i. If i ≥ j, which implies h (e) ≤ 2^{-j} (sample rate), the device instantly forwards event e to the coordinator; otherwise, the event is discarded locally. At the end of each epoch, the sensing device signs both sampled events and all counters it maintains. Note that none events are buffered at the device in any case. Algorithm 1 is the pseudo- code for the sensing device part of this sampling protocol.

2) Coordinator: The coordinator maintains queues $\{Q^k_i\}$, each of which corresponds to one numerical interval in {Si} of each sensing device. Upon receiving an event e, the coordinator first computes i, such that h (e) ∈ Si, followed by comparing the value of i and j. In the case of i< j , event e is discarded; otherwise, it is buffered at queue Q^k_i (suppose the event is from k_{th} sensing device) followed by updating both the counter associated with numerical interval Si and the global counter g, which records the total number of events buffered at the coordinator. At this moment, as long as the value of the global counter g exceeds the budget limit B, all event queues associated with Si are discarded, the global counter is updated accordingly and the sampling protocol advances to the next

round (i.e. j ← j + 1). The coordinator then signals all sensing devices to promote to the newest round
j. It is evident that coordinator buffers at most B + 1 events all the time. Hash chaining cannot coexist with the sampling protocol, because the coordinator is allowed to discard events that are essential for the verifier to validate the received data. DTC and GSC, on the other hand, do not bear the same problem. Algorithm 2 is the pseudo-code for the coordinator part of this sampling protocol.

B. Signature Schemes

Digital signature is widely used to ensure data authenticity and integrity. However, none of existing signature schemes is appropriate for the IoT setting. In this paper presents two new signature schemes.

Algorithm 2: SP at the Coordinator in Round *j*

for each *event e* **do**

$i \leftarrow \min\{x \in \mathbb{N} : h(e) \geq 2^{-x-1}\};$

$k \leftarrow e.source;$

if $i \geq j$ **then**

$Q_i^k.add(e);$

$l_i' \leftarrow l_i' + 1;$

$g \leftarrow g + 1;$

while $g > B$ **do**

Discard queues $\left\{\forall \hat{k}, Q_j^{\hat{k}}\right\};$

$g \leftarrow g - l_j';$

$j \leftarrow j + 1;$

Broadcast *j* to all sensing devices;

e

nd

else

Discard *e*;

e

nd

end

Dynamic Tree Chaining (DTC)

Here, start from the Tree chaining designed by Wong and Lam [22], one variation of Merkle tree[21]. The digest of each event report is one leaf node in binary authentication tree presented in Fig. 2. The value of the internal node is computed as the hashing of the concatenation of its two children. Take the authentication tree in Fig. 2 as an example. D12 is the parent of D1 and D2 and D12 = H (D1‖D2), where H (·) is the message digest function, such as SHA-1 [34] or MD5 [35], used for tree chaining. Likewise, D14 = H(D12‖D34) and D18 = H(D14‖D58). As a result, the root summarizes all the leaf nodes. The root node is regarded as the block digest. The block digest is appended with epochID and then signed by the private key to create the block signature. EpochID is used to identify which epoch the data are generated; otherwise, the cloud returns events from other epochs without being detected.

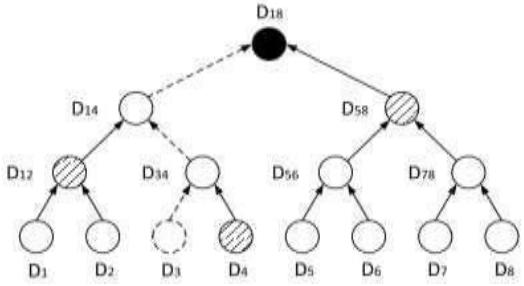


Fig. 2. Design of Tree chaining

The verification process is on a per-event basis. In order to verify the integrity/authenticity of an event e, the verifier requires the block signature, the position of event e in the authentication tree and the sibling nodes in the path to the root, which are all appended to event e. As a result, the overhead totransmit this metadata is O (log n), where n denotes the number of events.

Basically, the verification algorithm is to replay the process to build the authentication tree and to verify the nodes in the path to the root. Imagine the receiver begins to verify event e_3 which is represented as the dashed circle in Fig. 2.

First, the receiver computes $D'_3 = H(e_3)$ and then its ancestors in order: $D'_{34} = H(D'_3‖D_4)$, $D'_{14} = H(D_{12}‖D'_{34})$, $D'_{18} = H(D'_{14}‖H_{48})$. Event e_3 is verified if the decrypted block signature equal D'_{18}, thatis to say $\{D_{18}\}_{pk-1}$ pk = D 18, where $\{\cdot\}$pk−1 denotes singing using private key whereas $\{\cdot\}$pk is the function to decrypt signature with public key. In this case, all the nodes in the path, as well as their siblings, are verified and they could be cached to accelerate the verification process. Suppose the 4th event e4 arrives after e3 has been verified. Event e4 is verified directly if H(e4) = D4.

Geometric Star Chaining (GSC)

Here, propose a more efficient and secure data communication in this paper, called Geometric Star Chaining (GSC). The basic idea of GSC is inspired by one observation that any arbitrary fraction value can be represented or closely approximated by a few number of binary digits.

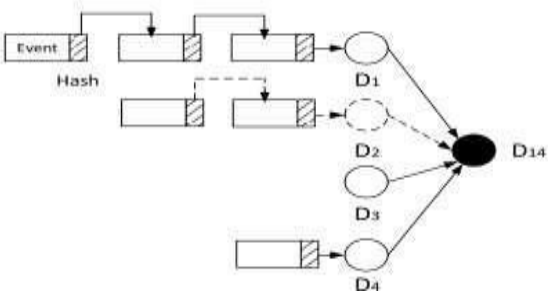


Fig. 3. Design of GSC.

The events included in the sample blocks are in geometric distribution. Each sample block should draw events uniformly from the IoT data stream. In order to ease the presentation of how sample blocks form, we define a set of successive numerical intervals {Si} where Si {$x \in R: 2-i-1 < x \leq 2-i , i \in N$}, which are visually represented as rectangles in Fig. 3. On receiving a new event e, the sensing device computes which numeric interval in {Si} that h(e) falls in and event e is inserted into the corresponding sample block, where h(·) is a non-cryptographic uniform random hashing function and $\forall x : 0 \leq h(x) \leq 1$.

Note that events in the same data block are either completely retrieved or not retrieved at all. Thus we can view each of such data block as an atomic "giant event". GSC computes one message digest for every block and concatenates these digests to a single digest for digital signature, as is depicted in Fig. 3. The digest of one sample block is computed in an online fashion. One variable Di is allocated to each sample block to capture the newest value of message digest. Suppose a new event e observed at the device which belongs to the i^{th} sample block. The message digest updates as Di = h (h (e) ‖Di), which is also referred as Merkle-Damgård Construction [20]. This online updating proceeds until the end of the epoch. At this time, concatenate approach is applied to all the message digests {Di}. The result summarizes all events generated in one epoch. Note the value i, which indicates the sampling rate of each block, should also be stored and hashed with the block. In this way, the application that receives the block can verify the sampling rate.

C) Onion Encryption and Decryption

An onion is the data structure formed by wrapping a message with successive layers of encryption to be decrypted only by data applications. Encryption holds promise in addressing all theseavenues of attack. Onion encryption protects and hides sensing data from IoT device to cloud storage. These onions have different layers each encrypted by using same key to reduce computational cost. Atthe end of each epoch, the encryption of sampled events is computed in a layered way with the symmetric key of Ks.

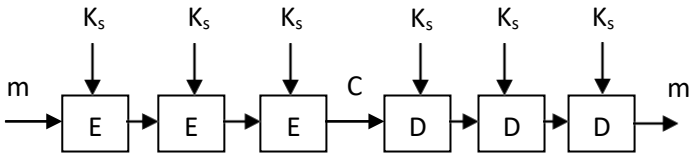
The encrypted onion part is generated

$$bym = O_{Ks} (e)$$

The encrypted onion part is updated by

$$\text{Cipher text, } C = O_{Ks} (m)$$

For decryption, removes one by onelayer on the top of the encrypted onion by using same symmetric key of Ks, and continues the full data reached.



It shows iteratively encrypts a message to get a cipher text that, when iteratively decrypted, yieldsthe original message.

D) Data Retrieval

A sampled fraction of sensing data is usually sufficient for most IoT applications [19]. In thesystem model presented in Sec. III-A, an application requests for a certain fraction of events

observed at a particular sensing device from the cloud. GSC provides verifiable authenticity, integrity, anduniformity for partial data retrieval with an arbitrary sampling rate.

Based on the application requirement, a data application first determines the maximum numberof events of each sensing device for an epoch it wants to receive, called a portion number. It then sends all portion numbers to the cloud. For each portion number, the cloud converts it to a sampling rate and constructs the binary expression. Then the cloud sends the corresponding sample blocks to the application. For the received sample blocks, the application first computes their digests as the final digest used for the signature. It then compares the final digest and the decrypted signature. This step verifies the following properties. 1) The received blocks were not modified or partially dropped and
2) The data were indeed uniformly sampled based on the given sampling rates and the uniform random hash function

The sampling protocol is compatible with DTC and GSC. It is natural for this budget-based sampling mechanism to be compatible with GSC since the sampling algorithm discarding the events half at each round which is essentially removing the existing largest GSC sampling block. As a result, the remaining buffered sample blocks correspond to successive numerical intervals. The data application can still fetch any fraction of data that is stored in the cloud. DTC requires a minor modification to support verifiable uniformity when the sampling protocol is performed. The sampling algorithm may discard the events specified by the random permutation.

IV. PERFORMANCE ANALYSIS

In this simulation, fix the budget limit to 100 events in this micro-scale experiment. The two linesin Fig. 5 represent the number of events sent to the coordinator by all the 7 sensing devices and monitored at all sensing devices, respectively. The two lines vary against time in one day. Initially, the number of events is the same for the two lines. It is worth mentioning that the total number of events sent to the coordinator grows slower with the time, which is a desirable property since the communication cost stays low even if much more events are monitored. This simulation experiment,to some extent, validates the theoretical analysis on the communication cost in which the communication cost only grows logarithmically.

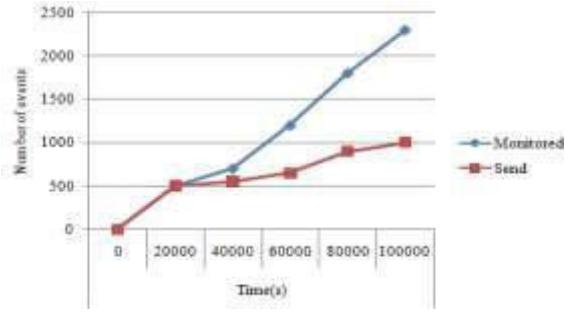


Fig. 5. One-day micro-scale exp

Next, investigate how different values of budget limit impact the number of events eventually savedto the cloud. Here present the number of events saved at the cloud with different values of budget limit in Fig. 6. Fig. 6 shows that this sampling protocol utilizes approximately 75% of the

budget on average for different budget values. In Fig. 6, also demonstrate that this sampling protocol works correctly in the presence of drastic changes, as the number of events monitored soars at the end.

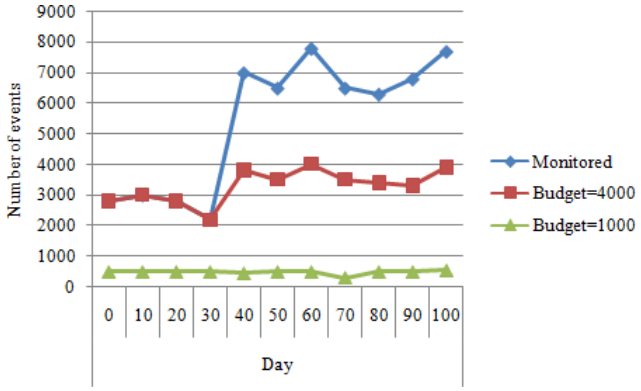


Fig. 6. Events saved in the cloud

Fig.7 illustrates the signing/verifying performance comparison between GSC and DTC under varied space available at the signer. It is obvious that both signing and verifying performance of DTC are capped by available memory at the signer, whereas GSC runs at full speed all the time

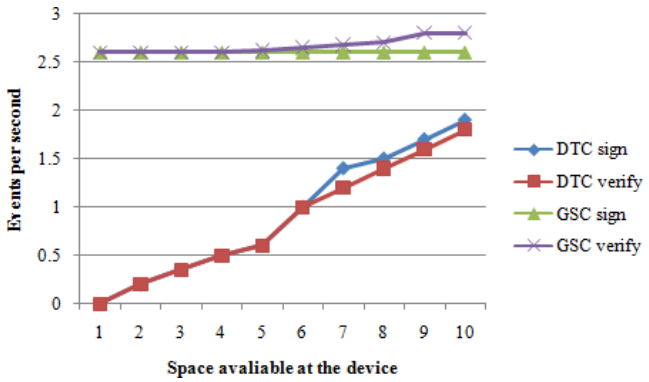


Fig. 7. Throughput comparison

V. CONCLUSION

In this paper, examined the practical problem of outsourcing evaluation over the outsourced cloud data. Due to the possible misbehaviours of the cloud server, result verification for the outsourced computation is a must. To fulfil the requirements, new challenges of the IoT data communication withauthenticity and data integrity an argue that existing solutions cannot be easily adopted. Besides, the information hiding in remote data integrity is still able to be efficiently executed using onion encryption. In addition, our scheme also meets the requirements to uniformly sample data from sensing devices and then securely store the data in the cloud while respecting resource budget constraint. The performance evaluations show that the proposed scheme is secure and efficient.

VI. REFERENCES

1. D. Giusto, A. Iera, G. Morabito, L. Atzori (Eds.), "The Internet of Things," Springer, 2010. ISBN: 978-1-4419-1673-0.J.
2. L. Atzori, A. Iera and G. Morabito, "The Internet of Things: A survey," Computer Networks (2010), doi: 10.1016/ j.comnet.2010.05.010
3. J. A. Stankovic, "Research Directions for the Internet of Things," IEEE Internet of Things Journal, vol. 1, no. 1, pp. 3-9, 2014.
4. D. Giusto, A. Iera, G. Morabito, L. Atzori (Eds.), The Internet of Things, Springer, 2010.
5. G. Tripathi, D. Singh, "EOI: Entity of Interest Based Network Fusion for Future Internet Services", ICHIT2011, September 23-25, 2011, Daejeon, Korea. © Springer-Verlag Berlin Heidelberg, CCIS, vol. 206, pp. 39–45, 2011.
6. Hall, D. L., Llinas, J., "Handbook of Multisensor Data Fusion," CRC Press, (2001).
7. R. Gennaro and P. Rohatgi. 1997. How to sign digital streams. In Crypto.