Efficient and Lightweight Data Streaming Authentication in Industrial Control and Automation Systems

Jian Xu, Qingyu Meng, Jun Wu, Member, IEEE, James Xi Zheng, Member, IEEE, Xuyun Zhang, Suraj Sharma

Abstract—The industrial control and automation systems have played an increasing important role in critical manufacturing processes. In such systems, many Internet of Things (IoT) devices continuously collect large number of streaming data for real-time processing. Verifiable data streaming (VDS) addresses such authenticity issue for streaming data, but most VDS schemes are not efficient and lightweight, do not support range querying, and cannot be used in practice. To improve the efficiency and achieve a verifiable range query in data streaming, we here present a new primitive, namely, a chameleon authentication tree with prefixes (PCAT), which is extended from the PBTree and CAT. Our scheme is not only lightweight but also supports dynamic expansion and verifiable range query in data streaming, making it more suitable for resource-constrained devices. We separate the PCAT’s algorithms into the following phases: initialization, data appending, query, and verification. Our analyses prove that the PCAT satisfies all the security requirements of VDS. Moreover, an efficiency analysis and performance evaluation demonstrate that our scheme not only supports lightweight data streaming authentication but also has high efficiency, which means that the PCAT is easier to apply in the industrial control and automation systems.

Index Terms—Internet of Things, Integrity, Chameleon Authentication Tree, Resource-constrained Devices, Verifiable Data Streaming

I. INTRODUCTION

TODAY, through the fourth industrial revolution, traditional manufacturing has transformed into a digital ecosystem [1]. As critical innovations, the Industrial Control and Automation Systems have been utilized in various programs all over the world [2]–[4]. These systems are highly interconnected and mutually dependent systems which support a variety of critical infrastructure management services using different kinds of IoT devices for sensing and actuation. IoT initiates a fundamental change within Industrial Control and Automation Systems making them increasingly connected and allowing them to exploit data to optimize their production processes.

However, most IoT devices are weak clients, meaning that they are usually resource-constrained [5]. Therefore, IoT clients cannot keep the data they sense in their local memory, and thus have to stream a massive number of data to a very powerful server, such as cloud storage [6]. Therefore, data integrity has become a critical issue, that is, how does the client ensure that the server has not changed its data? Furthermore, the client also needs to make sure that the server stores the current version of the data, and does not modify it or switch back to a previous version [7], [8].

The modeling of data management in industrial control and automation systems is shown in Fig. 1. There are three parts in this model, which include virtual devices, physical devices, and the cloud platform. Each of the physical devices, such as various monitoring components and other sensors embedded in the physical systems, can be considered as opportunistic data harvesters or as continuously producing new data streams, accumulating within a vast data repository amenable to subsequent analysis. Most physical devices they have limited storage and processing capacity. Thus, they have to stream their data to the more powerful cloud platforms and need to have the abilities to verify the data stream [9].

In industrial control and automation systems, most streaming data involve a large amount of important critical man-

FIG. 1. The modeling of data management in industrial control and automation systems.
ufacturing information. If these essential streaming data are tampered with, serious consequences will result. Therefore, it is crucial to construct protocols such that a client (such as a physical device) can be assured that a server (such as a cloud platform) is unable to add dummy data or making additional changes without the client’s approval, and that the stored data can be neither modified nor reordered.

Streaming data in industrial control and automation systems context have a unique characteristic of continuity, namely, the client can “see” only the current element; it cannot store the previous elements, and future elements cannot be predicted. This characteristic of streaming data makes data integrity verification a more difficult and challenging process.

Accordingly, verifiable data streaming (VDS) [9], [10], designed to guarantee the integrity of streaming data, has become an important streaming data integrity verification method. VDS is a protocol designed for clients with weak computing power. The client can verify its outsourced data through a streaming manner, and query the specified data from the server and update them. A large number of researchers have put significant effort into VDS and proposed some improved VDS schemes [9]–[19]. However, these schemes are still difficult to apply in practice. The main reasons for this are as followed: the number of outsourcing data is bounded, making it unsuitable for an increasing data stream; the storage cost of the client is always closely related to the number of data, however, in the streaming environment, the client needs to add or update elements, which will cause expensive computation overhead; and some schemes do not support a verifiable range query in streaming settings.

Focusing on the problems of the existing VDS schemes, we conducted a study on the integrity verification of data streaming. Accordingly, we propose a chameleon authentication tree with prefixes (PCAT) scheme, which combines the CAT [9], [10] and PBTree [20] to achieve range query and verification of data streaming, as well as to overcome the problems that affect most VDS schemes. Compared with CAT, our scheme supports a verifiable range query, while still maintaining good performance. We have designed and implemented the critical algorithms of PCAT, including initialization, data appending, query, and verification. We proved that the PCAT is sufficiently secure. Furthermore, we conducted a theoretical analysis of PCAT and compared it with other VDS schemes. Our results show that the PCAT achieves a better performance in the range query operation and prove its significant improvement.

We compare the PCAT with other schemes in Table I. And an overview of these related works is provided in Section II.

**TABLE I**

<table>
<thead>
<tr>
<th></th>
<th>Unbounded</th>
<th>Range Query</th>
<th>Range Verify</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT [9], [10]</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>CVCs [11]</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>SADS [16], [17]</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FHMT [13], [15]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PCAT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

II. RELATED WORK

Streaming data is an important form of data in the field of industrial control and automation [21]. In the industrial control and automation environment, most of the sources have limited storage space [22], which means that they have to outsource these data to the server for later use. Since the user must retrieve the correct answer from the server, the data streaming integrity verification problem (also referred to as VDS) [9]–[19], [23] has become very popular in academic circles. Although there are a large number of verifiable computation schemes, most of them are not suitable in the streaming environment. For example, some of them require the client to get all the data in advance and perform some pre-processing before outsourcing the data to the server. Therefore, these schemes are too computationally expensive in an industrial control and automation system streaming setting.

To resolve the abovementioned problems, Schröder et al. [9] first introduced the notion of VDS. VDS is similar to the verifiable database (VDB) [22], [24]: the main difference between the two is that no data need to be used during the setup phase in the VDS, and the other difference is that the data in the VDS is usually ordered, while the VDB usually does not consider the order of the data. Based on the concept of VDS, many schemes have been proposed. These schemes can be split into three categories: schemes based on chameleon hash function, schemes based on homomorphic encryption, and other schemes.

A. Schemes based on chameleon hash function

Schröder et al. [9] proposed a chameleon authentication tree (CAT) scheme, in which the leaves are not fixed in advance, any user with a public key can use the authentication path to verify the specified data, without the need to recompute all other leaves. However, although the CAT supports data adding, it fixes the size of data streaming, which means that it does not meet the requirements of an industrial control and automation system. To solve this issue, Schröder et al. [10] proposed an improved scheme that appends an unbounded data stream. Unfortunately, this scheme cannot perform verifiable range query efficiently.

Krupp et al. [11] presented two constructions for VDS: one is based on the chameleon vector commitment (CVC) by adding a chameleon hash function to every position in the vector commitment, while the other is based on shifting the workload to the server by combining signature schemes with cryptographic accumulators. Wu et al. [12] proposed a secure data stream outsourcing scheme, which is based on CVC to ensure the data integrity. However, these schemes introduce many additional random parameters, which resulting in expensive storage costs. At the same time, these schemes can verify only the signal element integrity and do not support range querying.

B. Schemes based on homomorphic encryption

Yu et al. [13], [14] proposed a series of streaming authenticated data structures, namely, a fully homomorphic
encryption-based Merkle tree (FHMT) [13] and a partially homomorphic encryption-based arithmetic Merkle tree (PAMT) [14], and integrated them with fully and partially homomorphic encryption for VDS. Xu et al. [15] proposed a dynamic fully homomorphic encryption-based Merkle tree, which is a more suitable option for weak clients. However, due to the low efficiency of homomorphic encryption, these schemes are infeasible in current industrial automation and control systems. Meanwhile, these schemes do not consider the range query operation.

C. Other schemes

Papamanthou et al. [16] proposed SADSs, a model that can perform efficient verifiable hash query in the streaming setting, which is a generalized hash tree that is extended from the original Merkle tree. The SADS supports a variety of efficient operations, such as range query and frequency query. However, this scheme is too complicated, making it difficult to design and implement in the high-level applications. Based on the SADS, Qian et al. [17] proposed an abstract SADS construction, any hash function that meets the author’s definition can be used. However, since their scheme still use a lot of computationally complex functions, it still has the problem of inefficiency. Moreover, this scheme has a problem in common with the SADS: the upper bound of data streaming is still predefined.

Zhang et al. [18] defined and constructed a dimension-increasing vector commitment (DIVC), which is a new primitive that is extended from vector commitment. And these authors used their purposed DIVC to construct their novel, namely, constant verifiable data streaming schemes. Tsai et al. [19] proposed the VENUS scheme, which is built using a new structure called the HPBTree. VENUS adds hash functions and weights to the PBTree to achieve efficient range querying and verification; however, the upper bound of data still needs to be determined in advance. Guo et al. [25] proposed a trusted and distributed authentication system, aiming to promote the efficiency of the authentication for IoT terminals. But this scheme is not suitable for streaming data environment.

Obviously, these works have significant meanings for integrity preservation in the industrial automation and control system environment; nevertheless, these schemes are also impacted by some shortcomings, such as the efficiency of the schemes being too low to use in practice and data updates not being supported. Compared with the above schemes, our scheme supports unbounded data and an efficient verifiable range query, which is more suitable for industrial control and automation system environments.

III. PCAT Scheme

A. Definitions

Before defining the PCAT, we first list the symbols used later in this work in Table II. The definitions related to prefix can be found in [20], thus will not be described here.

**Definition 1** The PCAT can be described as a tuple of probabilistic polynomial-time algorithms as follows:

\[ \text{PCAT} = (\text{Initial}, \text{Append}, \text{rangeQuery}, \text{rangeVerify}) \]

**Initial**($\lambda$): Initialization algorithm. Input $\lambda$ as the security parameter. Generate the public/private key pair of the chameleon hash function $(cpk, csk) \leftarrow \text{chGen}(\lambda)$ and the public/private key pair of the user $(upk, usk)$, return $(cpk, upk)$ as the PCAT public key $pk$, and return $(csk, usk, \text{pcatstruct})$ as the PCAT private key $sk$, where $\text{pcatstruct}$ denotes the structure of the PCAT and is empty in the initialization.

**Append**$(sk, data_i, i)$: Data appending algorithm. Append $data_i$ to the PCAT. Generate or update the prefix sequence according to the node’s position, and then update the hash value or the random number of the corresponding node after appending the data. If the PCAT is extended when appending data, sign the new root node with user’s private key $usk$.

**rangeQuery**$(v, i, j)$: Query the elements from the $i^{th}$ to the $j^{th}$ data. Return all the data $data_{i,j}$ in $[i, j]$ and the corresponding authentication path $auth_{i,j}$ if the query is successful, and output 1; otherwise, output 0.

**rangeVerify**$(pk, i, j, data_{i,j}, auth_{i,j})$: Use $pk$, $auth_{i,j}$ to verify whether $data_{i,j}$ are elements from the $i^{th}$ to the $j^{th}$ data. Output 1 if the verification is successful; otherwise, output 0.

**Definition 2** The correctness of the PCAT.

For a PCAT, if for all $q = \text{poly}(X)$, call $\text{Initial}(\lambda)$ once; for all $(data_1, \ldots, data_q)$, call $\text{Append}(sk, data_q, q)$; and for all $i \leq q, j \leq q, i \leq j$, there is the following:

If $(data_{i,j}, auth_{i,j}) \leftarrow \text{rangeQuery}(v, i, j)$, then

1 $\leftarrow \text{rangeVerify}(pk, i, j, data_{i,j}, auth_{i,j})$ holds.

Then the PCAT is correct.

**Definition 3** The security of the PCAT.

If a PCAT meets the following properties:

The adversary cannot replace or modify the nodes in the PCAT and cannot add data to the PCAT by itself. In short, if the query submitted by the verifier returns an incorrect response, the verification cannot be passed.

then the PCAT can be considered as secure.
The security of the PCAT is described below in the form of an interactive game between a challenger and an adversary $A$.

The challenger generates a public/private key pair $(pk, sk)$, then sends the public key $pk$ to the adversary $A$. The adversary $A$ can adaptively send $poly(\lambda)$ data $(data_1, data_2, \ldots, data_{poly(\lambda)})$ to the challenger. The challenger then inserts the data into the PCAT and returns all authentication paths $(auth_{1,2}, \ldots, auth_{i,poly(\lambda)}; \ldots, auth_{poly(\lambda)−1,poly(\lambda)})$. Next, the adversary $A$ attempts to modify, replace, or add nodes in the PCAT, and outputs an authentication path which is not exist in the PCAT.

The formal description of the interactive game is as follows:

**Setup:** The challenger executes $Initial(1^\lambda)$ to generate the public/private key pair $(pk, sk)$ and sends the public key $pk$ to the adversary $A$.

**Update:** The adversary $A$ can select $poly(\lambda)$ data $(data_1, data_2, \ldots, data_{poly(\lambda)})$ and sends the data to the challenger, after which the challenger executes $Append$ to insert the data into the PCAT and then executes $auth_{i,j} \leftarrow rangeQuery(v, i, j)$ to generate all authentication paths $auth_{range} = (auth_{1,2}, \ldots, auth_{i,poly(\lambda)}, \ldots, auth_{poly(\lambda)−1,poly(\lambda)})$. The response sequence obtained by the adversary $A$ is denoted as $Q_{range} := \{(d_{1,2,1,2}, auth_{1,2}), \ldots, (d_{i,poly(\lambda),i,poly(\lambda)}, auth_{i,poly(\lambda)}), \ldots, (d_{poly(\lambda)−1,poly(\lambda)}, poly(\lambda)−1,poly(\lambda)), auth_{poly(\lambda)−1,poly(\lambda)})\}$. The non-dummy leaf node stores a prefix family $F(i)$, which corresponds to the serial number of the element. The challenge family stored by the dummy node is uniformly represented by $F(0)$, as the data in the PCAT starts from the No. 1 and has no data No. 0. The union of prefix families $U(v)$ stored by the parent node is pregenerated and covers the range of possible data serial numbers of all child nodes, as specified in Section IV.

**Output:** Finally, the adversary $A$ outputs $(d_{i+j, i+j}, auth_{i+j})$. If $(d_{i+j, i+j}, auth_{i+j}) \notin Q_{range}$ and $rangeVerify(pk, i+j, data_{i+j}, auth_{i+j}) = 1$, then the adversary wins.

Define $Pr_A$ as the probability of the adversary $A$ winning the game. If the probability $Pr_A$ is negligible according to any polynomial-time algorithm, then the PCAT is secure.

B. Data structure

The PCAT’s structure is presented in Fig. 2. All right child nodes (blue nodes) are chameleon hash nodes that use a chameleon hash function to compute the hash value. All child nodes and the root node (white nodes and red nodes) are normal hash nodes that use a normal collision-resistant hash function to compute the hash value. The node at the top left (red node) contains a digital signature of the hash value, which can be used as the root node for verification at a certain stage. For example, in the state depicted in Fig. 2, only $\rho3$ is the valid verification root node.

In addition to storing the hash value in each node and the random number in the chameleon hash node, the PCAT needs to store the corresponding prefix sequence in each node. The non-dummy leaf node stores a prefix family $F(i)$, which corresponds to the serial number of the element. The prefix family stored by the dummy node is uniformly represented by $F(0)$, as the data in the PCAT starts from the No. 1 and has no data No. 0. The union of prefix families $U(v)$ stored by the parent node is pregenerated and covers the range of possible data serial numbers of all child nodes, as specified in Section IV.

IV. PCAT Algorithms

A. Initialization

Since the PCAT is dynamically extended, it is not necessary to determine the tree’s depth during initialization, the tree’s structure is empty. The PCAT only needs to generate the public/private key pair of the chameleon hash function and the public/private user key pair during initialization. The PCAT is an empty tree after initialization.

**Algorithm 1** The initialization algorithm $Initial(1^\lambda)$

1: **begin**
2: $(cpk, csk) \leftarrow chGen(1^\lambda)$
3: **root** $\leftarrow$ **NULL**
4: size $\leftarrow 0$, capacity $\leftarrow 0$
5: depth $\leftarrow 0$
6: pcatstruct $\leftarrow$ **NULL**
7: **sk** $\leftarrow$ $(csk, usk, pcatstruct)$
8: **pk** $\leftarrow$ $(cpk, upk)$
9: **return** $(sk, pk)$

B. Data appending

The data appending process consists of three parts: generating nodes on the appending path, generating or updating the node’s prefix sequence, and updating the corresponding node’s hash value after appending data. The generating node and the updating hash value parts are the same as in the CAT appending algorithm and will therefore not be described here. The prefix sequence part of the data appending process is described below.

Let the bottom of the tree have a depth of 0, and the $i^{th}$ data is the data that will be inserted. The node $v$ in the appending path (except for the root node, dummy node, and the leaf node storing data) stores $U(v) = F(i) \cup F(i+1) \cup \ldots \cup F(i+2^{depth−1})$ at the time of generation. In addition, the prefix family of the leaf node storing data is $F(i)$, and the prefix family of the dummy node is uniformly represented by $F(0)$. When the extension is required, we first compute the prefix sequences of the nodes below the new root node $root$ and then compute $U(root) = U(root \rightarrow leftchild) \cup U(root \rightarrow rightchild)$.

An example of the PCAT appending algorithm is presented as Fig. 3. At this point, four data points have been inserted.
Algorithm 2 Data appending algorithm

```plaintext
1: begin
2: if (size == capacity) 
3:     Generate new root 
4:   from root to leaf
5:     Generate new node
6:   Generate or update node prefix sequence
7: Update node hash value until the chameleon hash node
8:   if(size == capacity) 
9:     Update root node’s prefix sequence
10:   root.sign ← size
11: size ← size + 1 
```

C. Query

The query algorithm consists of two parts: querying data and generating the authentication path.

In terms of querying data, for a given range \([i, j]\), first, compute \(S([i, j])\), and then check whether \(U(\text{root}) \cap S([i, j]) = \phi\).

If the set is empty, no data fall into the range, and the query terminates; otherwise, if at least one data point is in \([i, j]\), we then recursively check whether \(U(\text{root}) \cap S([i, j]) = \phi\) in the left and right subtrees. If the set is not empty, continue to compute downward until reach the leaf node.

Moreover, when generating the authentication path, if the query result is empty, that is, \(U(\text{root}) \cap S([i, j]) = \phi\), then the authentication path is also empty; otherwise, under the root node, if the sibling node of current query node \(v\) is not in the query range (that is, if \(U(v.brother) \cap S([i, j]) = \phi\), then add node \(v.brother\) to the authentication path. In the same way as for the CAT, the chameleon node’s random number in the path is also added to the authentication path.

In particular, when performing a single data query, i.e., the query range \(i = j\), the PCAT uses the CAT query algorithm. In addition, the method used to generate an authentication path for a single data query and the method for verifying a single data point are the same as for the CAT and thus are no longer described here.

Algorithm 3 Range query algorithm rangeQuery(v, i, j)
```
1: begin 
2: if (U(\text{root}) \cap S([i, j]) = \phi) 
3:     return NULL 
4: if (U(v) \cap S([i, j]) \neq \phi) & & v.type == Cham 
5:     auth.addRand(v.random) 
6: if (v is leaf & & U(v) \cap S([i, j]) \neq \phi) 
7:     queryResult.push_back(v) 
8: return 
9: if (U(v) \cap S([i, j]) = \phi) 
10:   auth.addNode(v) 
11: return 
12: else 
13:     rangeQuery(v.leftchild, i, j) 
14:     rangeQuery(v.rightchild, i, j) 
```

D. Verification

The verification algorithm progresses as follows: first, verify the digital signature of the root node; next, use the query result data\(_{i,j}\) to compute the leaf nodes’ hash value; then, compute the parent nodes’ hash value from left to right on each layer until root node’s hash value \(h^*_p\) is recomputed; and finally, compare \(h^*_p\) to the locally stored \(h_p\) to determine whether the verification was successful.

For example, as shown in Fig. 4, five data points have been inserted into the PCAT, data from No. 3 to No. 5 have been queried, the query path and the passing nodes are marked in green, and the corresponding authentication path is \(\text{auth}_{3,5} = (p_4, p_2, r_{34}, r_4, r_{500}, 00, 0)\). First, verify that the digital signature of the root node’s hash value is correct, and then compute the hash value \(h_3, h_4, h_5\) of data\(_{3,5}\) corresponding to the leaf nodes. Next, according to the authentication path, compute \(h_{34} \leftarrow \text{ch}(\text{cpk}, h_3||h_4, r_{34})\) and \(h_{50} \leftarrow H(h_5||h_0)\) in order. On the second layer, compute \(h_{30} \leftarrow H(h_{34}, h_{50})\) from left to right in the same way. Finally, the root node’s
hash value $h'_{p4}$ is recomputed and then compared with $h_{p4}$ to determine whether the verification was successful.

Algorithm 4 Range verification algorithm

\[
\text{rangeVerify}(pk, i, j, data_{i,j}, auth_{i,j})
\]

1: \hspace{1em} begin
2: \hspace{1em} if \text{root.signature.verify}() \hspace{1em} then
3: \hspace{2em} return 0
4: \hspace{1em} end if
5: \hspace{1em} \text{for } n = 0 \text{ to } j - i \\hspace{1em} do
6: \hspace{2em} compute \text{H(queryResult}[n].data)
7: \hspace{1em} \text{end for}
8: \hspace{1em} return \text{h}_{p} = \text{h}_{p}

V. SECURITY AND PERFORMANCE

A. Security Analysis

First, select any $\lambda \in \mathbb{N}$; the client executes $\text{Initial}(1^\lambda)$ to generate the public/private key pair $(pk, sk)$. Then, the client executes $\text{Append}$ to insert $q = \text{poly}(\lambda)$ data. Every time the $\text{Append}$ algorithm is called, the prefix sequence and the hash value or a random number of the corresponding node is correctly updated.

In terms of query and verification, for a query range $[i, j]$, the server returns the correct query result, $data_{i,j}$ and authentication path $auth_{i,j} = \{\text{root}, v.bro, r\}$. Here, $\text{root}$ is the root node, $v.bro$ is the set of sibling nodes of $v$, and $r$ is the set of random numbers of the chameleon nodes on the query path. The verifier executes $\text{rangeVerify}$ for verification. First, verify the digital signature of the root node, and then use $data_{i,j}$ to compute the hash value $h_{i}, \ldots, h_{j}$ of the leaf nodes in the query range. Next, compute the parent nodes’ hash value from left to right on each layer until the root node’s hash value is recomputed. The computation process can be derived from either formula (1) or formula (2):

\[
h_{\text{parent}} = H(h_{\text{leftchild}}||h_{\text{rightchild}})
\]

\[
h_{\text{parent}} = ch(cpk, h_{\text{leftchild}}||h_{\text{rightchild}}, r_{\text{parent}})
\]

Both $h$ and $r$ can be obtained from $data_{i,j}$, $auth_{i,j}$, or the computation results from the previous step.

Assuming that both the client and server execute the algorithms by following the correct steps, and the generated result are not tampered with, the verifier can then use the returned results and the authentication path to recompute the root node. Therefore, a comparison of this node with the root node stored locally is always carried out to verify. When $i = j$, that is, when performing a single data query and verification, the core idea is the same, although the process of querying and the process of generating the authentication path are different. Therefore, the analysis process is also the same and will not be described here.

Therefore, the PCAT is correct.

Assuming that the adversary $A$ has sufficient computational ability to append $\text{poly}(\lambda)$ data into the PCAT. For the adversary $A$, the structure of the PCAT $(\text{pcaststruct})$ is kept as a private key, and the adversary is unable to obtain it. If the adversary wants to win, it needs to output a tuple $(d_{i,j}, i*, j*, auth_{i*,j*}) \notin Q_{\text{range}}$, and $\text{rangeVerify}(pk, i*, j*, data_{i*,j*}, auth_{i*,j*}) = 1$. The following presents a detailed proof of the security of the PCAT:

Let the set of authentication paths for all single data queries of the PCAT be $\text{auth}_{\text{single}} = (\text{auth}_{1}, \text{auth}_{2}, \ldots, \text{auth}_{\text{poly}(\lambda)})$ and the single data verification algorithm be $\text{Verify}(pk, i, data_{i}, auth_{i})$. Then, the following holds:

**Theorem 1.** If $\exists auth_{i*,j*} \neq auth_{\text{range}}$ and $\text{rangeVerify}(pk, i*, j*, data_{i*,j*}, auth_{i*,j*}) = 1$, then $\exists n \in [i*, j*]$, $auth_{n} \notin \text{auth}_{\text{single}}$ and $\text{Verify}(pk, n, data_{n}, auth_{n}) = 1$.

**Proof.** First, prove $\exists n \in [i*, j*]$, $auth_{n} \notin \text{auth}_{\text{single}}$. If it does not exist, that is, $\forall n \in [i*, j*]$, then there is $auth_{n} \in \text{auth}_{\text{single}}$. Since $auth_{i*,j*} \subseteq \{\text{auth}_{i}, \ldots \cup \text{auth}_{j}\}$ and $\{\text{auth}_{i} \cup \ldots \cup \text{auth}_{j}\} \subseteq \text{auth}_{\text{single}} = \text{auth}_{\text{range}}$, there are $auth_{i*,j*} \in \text{auth}_{\text{range}}$, which contradicts $auth_{i*,j*} \notin \text{auth}_{\text{range}}$.

Then, prove that $\text{Verify}(pk, n, data_{n}, auth_{n}) = 1$. If $\text{Verify}(pk, n, data_{n}, auth_{n}) \neq 1$, this indicates that the verification has failed. Since single verification is a particular case of range verification, the computation range of range verification includes the computation range of single verification; thus, the range verification will fail, which contradicts the assumption.

**Theorem 2.** If the CAT is secure, then the PCAT is secure.

**Proof.** It can be seen from Theorem 1 that if the adversary wants to win, this is equivalent to finding $n \in [i*, j*]$; hence, $auth_{n} \notin \text{auth}_{\text{single}}$ and $\text{Verify}(pk, n, data_{n}, auth_{n}) = 1$, which contradicts the security of the CAT. The security of the CAT has been proven in [9]; therefore, the CAT is secure, and thus the PCAT is secure.

B. Simulation Results and Analysis

This paper mainly tests the time cost performance of the PCAT in terms of the data appending, query, and verification tasks. The configuration of our experiments is Windows 10 Professional, 8GB memory, and 4.0GHz Intel® Core™ i7-6700k processor.

We first compare the average time of the CAT, PCAT, PBTree, and FHMT when appending each data point under different number of data. The results of this comparison are...
The core concept behind the PCAT and FHMT is the same, and it can be seen that the PCAT is slightly better than the FHMT.

6. The range and position of the verification are the same. It under different size of data is presented in Fig. 5. It can be seen that the appending time of the four algorithms do not grow as the number of data grows significantly. Meanwhile, PCAT achieves a slightly lower appending time than that of the PBTree and CAT. This outcome occurs because the PCAT computes the union of the prefix sequence of the node on the path when appending data, while the nodes in the PBTree store the set of prefix families. Since the CAT does not involve range querying and verification, there is no need to compute the prefix sequence when appending data. Therefore, the appending time of the PCAT is lower than that of the CAT and PBTree, as is expected. The appending time of the PCAT exhibits an obvious advantage over the FHMT, which is mainly because the fully homomorphic encryption algorithm used by the FHMT is less efficient at present.

<table>
<thead>
<tr>
<th>TABLE III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparison of the average time between four algorithms when appending each data point under different number of data (UNIT: s)</td>
</tr>
<tr>
<td>Data Size</td>
</tr>
<tr>
<td>PBTree</td>
</tr>
<tr>
<td>CAT</td>
</tr>
<tr>
<td>FHMT</td>
</tr>
<tr>
<td>PCAT</td>
</tr>
</tbody>
</table>

The time cost comparison of the range query of the PBTree, PCAT, and FHMT under different size of data is presented in Fig. 5. The following are the results of querying 32 identical data points; the number of data and query range is the same. Since the PCAT requires performing only a simple comparison operation and the union of prefix sequences is precomputed at the time of data appending, it has the best query time. When the PBTree performs the range query, it needs to generate the union of prefix families for each node on the path. Therefore, as the number of data grows, the query time of the PBTree decreases significantly. At the same time, the PCAT is more efficient than the FHMT in terms of the range query.

The comparison of the PCAT and FHMT in terms of range verification under different size of data is presented in Fig. 6. The range and position of the verification are the same. It can be seen that the PCAT is slightly better than the FHMT. The core concept behind the PCAT and FHMT is the same, that is, recomputing the root node, and comparing it with the original root node. The cost of the FHMT is mainly located in the homomorphic encryption operation, while the cost of the PCAT is mainly in the chameleon hash function operation.

We then increased the number of data and conducted a more detailed experiment to test the performance of the PCAT under these circumstances. The experiment selected \( n = 1024, 2048, 4096, 8192 \) data points to test the PCAT. The results are as follows:

The average time for appending each data point under different number of data is shown in Fig. 7. It can be seen that after increasing the number of data, the PCAT is still able to maintain efficient data appending.

The query time of the PCAT under different number of data and query ranges is presented in Fig. 8. It can be seen that for the query, an increase in the number of data has little effect on the query time. Moreover, since the efficiency of the verification depends on the number of layers in the PCAT, an increase in the data has almost no impact on the verification time. Therefore, we next tested the verification time of PCAT under different query ranges. The number of data in this test is \( n = 8192 \), and the result is shown in Fig. 9. It can be seen from Fig. 8 and Fig. 9 that the query and verification times increase almost linearly with the query range and are always very efficient.

In summary, the PCAT achieves high performance on the data appending, range querying, and range verification tasks. In particular, the PCAT performs better on data appending, range querying, and range verification than the FHMT, which also supports range querying and verification. As the number of data grows significantly. Meanwhile, PCAT achieves a slightly lower appending time than that of the PBTree and CAT. This outcome occurs because the PCAT computes the union of the prefix sequence of the node on the path when appending data, while the nodes in the PBTree store the set of prefix families. Since the CAT does not involve range querying and verification, there is no need to compute the prefix sequence when appending data. Therefore, the appending time of the PCAT is lower than that of the CAT and PBTree, as is expected. The appending time of the PCAT exhibits an obvious advantage over the FHMT, which is mainly because the fully homomorphic encryption algorithm used by the FHMT is less efficient at present.

The time cost comparison of the range query of the PBTree, PCAT, and FHMT under different size of data is presented in Fig. 5. The following are the results of querying 32 identical data points; the number of data and query range is the same. Since the PCAT requires performing only a simple comparison operation and the union of prefix sequences is precomputed at the time of data appending, it has the best query time. When the PBTree performs the range query, it needs to generate the union of prefix families for each node on the path. Therefore, as the number of data grows, the query time of the PBTree decreases significantly. At the same time, the PCAT is more efficient than the FHMT in terms of the range query.

The comparison of the PCAT and FHMT in terms of range verification under different size of data is presented in Fig. 6. The range and position of the verification are the same. It can be seen that the PCAT is slightly better than the FHMT. The core concept behind the PCAT and FHMT is the same, that is, recomputing the root node, and comparing it with the original root node. The cost of the FHMT is mainly located in the homomorphic encryption operation, while the cost of the PCAT is mainly in the chameleon hash function operation.

We then increased the number of data and conducted a more detailed experiment to test the performance of the PCAT under these circumstances. The experiment selected \( n = 1024, 2048, 4096, 8192 \) data points to test the PCAT. The results are as follows:

The average time for appending each data point under different number of data is shown in Fig. 7. It can be seen that after increasing the number of data, the PCAT is still able to maintain efficient data appending.

The query time of the PCAT under different number of data and query ranges is presented in Fig. 8. It can be seen that for the query, an increase in the number of data has little effect on the query time. Moreover, since the efficiency of the verification depends on the number of layers in the PCAT, an increase in the data has almost no impact on the verification time. Therefore, we next tested the verification time of PCAT under different query ranges. The number of data in this test is \( n = 8192 \), and the result is shown in Fig. 9. It can be seen from Fig. 8 and Fig. 9 that the query and verification times increase almost linearly with the query range and are always very efficient.

In summary, the PCAT achieves high performance on the data appending, range querying, and range verification tasks. In particular, the PCAT performs better on data appending, range querying, and range verification than the FHMT, which also supports range querying and verification. As the number of data grows.
The PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate that the PCAT can be seen to have high efficiency and practicability.

VI. CONCLUSIONS

Data streaming has been widely used in the industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT.

REFERENCES


[7] S. Luo, Y. Wen, W. Xu, and D. Puthal, “A cloud-based cyber-physical system for industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT.

VI. CONCLUSIONS

Data streaming has been widely used in the industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT.

REFERENCES


[7] S. Luo, Y. Wen, W. Xu, and D. Puthal, "A cloud-based cyber-physical system for industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT.

VI. CONCLUSIONS

Data streaming has been widely used in the industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT.

REFERENCES


[7] S. Luo, Y. Wen, W. Xu, and D. Puthal, “A cloud-based cyber-physical system for industrial control and automation systems. Because most IoT devices are resource-constrained, they have to outsource their data to the cloud; as a result, integrity verification has become increasingly important. However, most existing integrity verification schemes are inefficient, do not support range querying and cannot be implemented in practice. Therefore, we combine the PBTree and CAT to propose a PCAT, which is capable of overcoming the shortcomings that affect most VDS schemes. This paper proposes the PCAT algorithm, including initialization, data appending, query and verification. The security of the PCAT is analyzed, and the results demonstrate that our scheme is sufficiently secure. Finally, we analyze the PCAT and compare it with other VDS schemes. The results illustrate the efficiency and practicability of the PCAT."
Jian Xu received the B.S., M.S. and Ph.D. degrees in computer application technology from the College of Information Science and Engineering, Northeastern University, Shenyang, China, in 2001, 2007, and 2013, respectively. He is currently an Associate Professor/ Ph.D. Supervisor of Software College, Northeastern University (NEU), China. He is also the Dean of information security, NEU. His research interests include network and information security, cloud security, AI security, etc.

Qingyu Meng received a B.E. degree in software engineering from the Software College in Northeastern University, Shenyang, China in 2018. He is currently studying for an M.E. degree at the Software College, Northeastern University, Shenyang, China. His research interests include cryptography and cloud security.

Jun Wu received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011. He was a Post-Doctoral Researcher with National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with Waseda University, Japan, from 2011 to 2013. He is currently a professor of School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), China. He is also the vice director of National Engineering Laboratory for Information Content Analysis Technology, SJTU. His research interests include the advanced computing, communications and security techniques of software-defined networks (SDN), information-centric networks (ICN), Internet of Things (IoT), 5G/6G, molecular communication, etc.. He has been the Track Chair of VTC 2019, VTC 2020. He has been a Guest Editor of IEEE Transactions on Industrial Informatics, IEEE Sensors Journal, Sensors. He is an Associate Editor of the IEEE Access, IEEE Networking Letters.

Dr. James Xi Zheng, PhD in Software Engineering from UT Austin USA; Chief Solution Architect for Menulog Australia, now director of Intelligent systems research center (itsrc.org), deputy director of software engineering, global engagement, and assistant professor in Software Engineering at Macquarie University. Specialised in Service Computing, IoT Security and Reliability Analysis. Published more than 80 high quality publications in top journals and conferences. Awarded multiple best papers in leading conferences and a few highly competitive funding including ARC LP and Data61 CRP as the Lead Investigator.

Dr Xuyun Zhang is currently working as a senior lecturer in Department of Computing at Macquarie University in Australia. He worked as a lecturer in University of Auckland during 2016~2019, and a postdoc researcher in NICTA during 2014~2016. He got his Phd degree in Computer Science and Technology from University of Technology Sydney (UTS), Australia, and his Master’s and Bachelor’s degrees in the same major from Nanjing University, China. He is an early/mid-career researcher with an international track record of research in areas including scalable and secure machine learning, big data privacy and cyber security, big data mining and analytics, cloud/edge/service computing and IoT, etc. He has so far published over 100 refereed academic papers on many high-quality and influential conferences and journals in the abovementioned areas, such as IEEE TPDS, TKDE, TC, TSE, TII, JSAC and ICDE.

Qingyu Meng received a B.E. degree in software engineering from the Software College in Northeastern University, Shenyang, China in 2018. He is currently studying for an M.E. degree at the Software College, Northeastern University, Shenyang, China. His research interests include cryptography and cloud security.

Jun Wu received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011. He was a Post-Doctoral Researcher with National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with Waseda University, Japan, from 2011 to 2013. He is currently a professor of School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), China. He is also the vice director of National Engineering Laboratory for Information Content Analysis Technology, SJTU. His research interests include the advanced computing, communications and security techniques of software-defined networks (SDN), information-centric networks (ICN), Internet of Things (IoT), 5G/6G, molecular communication, etc.. He has been the Track Chair of VTC 2019, VTC 2020. He has been a Guest Editor of IEEE Transactions on Industrial Informatics, IEEE Sensors Journal, Sensors. He is an Associate Editor of the IEEE Access, IEEE Networking Letters.

Dr. James Xi Zheng, PhD in Software Engineering from UT Austin USA; Chief Solution Architect for Menulog Australia, now director of Intelligent systems research center (itsrc.org), deputy director of software engineering, global engagement, and assistant professor in Software Engineering at Macquarie University. Specialised in Service Computing, IoT Security and Reliability Analysis. Published more than 80 high quality publications in top journals and conferences. Awarded multiple best papers in leading conferences and a few highly competitive funding including ARC LP and Data61 CRP as the Lead Investigator.

Dr Xuyun Zhang is currently working as a senior lecturer in Department of Computing at Macquarie University in Australia. He worked as a lecturer in University of Auckland during 2016~2019, and a postdoc researcher in NICTA during 2014~2016. He got his Phd degree in Computer Science and Technology from University of Technology Sydney (UTS), Australia, and his Master’s and Bachelor’s degrees in the same major from Nanjing University, China. He is an early/mid-career researcher with an international track record of research in areas including scalable and secure machine learning, big data privacy and cyber security, big data mining and analytics, cloud/edge/service computing and IoT, etc. He has so far published over 100 refereed academic papers on many high-quality and influential conferences and journals in the abovementioned areas, such as IEEE TPDS, TKDE, TC, TSE, TII, JSAC and ICDE.

Qingyu Meng received a B.E. degree in software engineering from the Software College in Northeastern University, Shenyang, China in 2018. He is currently studying for an M.E. degree at the Software College, Northeastern University, Shenyang, China. His research interests include cryptography and cloud security.

Jun Wu received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011. He was a Post-Doctoral Researcher with National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with Waseda University, Japan, from 2011 to 2013. He is currently a professor of School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), China. He is also the vice director of National Engineering Laboratory for Information Content Analysis Technology, SJTU. His research interests include the advanced computing, communications and security techniques of software-defined networks (SDN), information-centric networks (ICN), Internet of Things (IoT), 5G/6G, molecular communication, etc.. He has been the Track Chair of VTC 2019, VTC 2020. He has been a Guest Editor of IEEE Transactions on Industrial Informatics, IEEE Sensors Journal, Sensors. He is an Associate Editor of the IEEE Access, IEEE Networking Letters.

Dr. James Xi Zheng, PhD in Software Engineering from UT Austin USA; Chief Solution Architect for Menulog Australia, now director of Intelligent systems research center (itsrc.org), deputy director of software engineering, global engagement, and assistant professor in Software Engineering at Macquarie University. Specialised in Service Computing, IoT Security and Reliability Analysis. Published more than 80 high quality publications in top journals and conferences. Awarded multiple best papers in leading conferences and a few highly competitive funding including ARC LP and Data61 CRP as the Lead Investigator.

Dr Xuyun Zhang is currently working as a senior lecturer in Department of Computing at Macquarie University in Australia. He worked as a lecturer in University of Auckland during 2016~2019, and a postdoc researcher in NICTA during 2014~2016. He got his Phd degree in Computer Science and Technology from University of Technology Sydney (UTS), Australia, and his Master’s and Bachelor’s degrees in the same major from Nanjing University, China. He is an early/mid-career researcher with an international track record of research in areas including scalable and secure machine learning, big data privacy and cyber security, big data mining and analytics, cloud/edge/service computing and IoT, etc. He has so far published over 100 refereed academic papers on many high-quality and influential conferences and journals in the abovementioned areas, such as IEEE TPDS, TKDE, TC, TSE, TII, JSAC and ICDE.

Qingyu Meng received a B.E. degree in software engineering from the Software College in Northeastern University, Shenyang, China in 2018. He is currently studying for an M.E. degree at the Software College, Northeastern University, Shenyang, China. His research interests include cryptography and cloud security.

Jun Wu received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011. He was a Post-Doctoral Researcher with National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with Waseda University, Japan, from 2011 to 2013. He is currently a professor of School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), China. He is also the vice director of National Engineering Laboratory for Information Content Analysis Technology, SJTU. His research interests include the advanced computing, communications and security techniques of software-defined networks (SDN), information-centric networks (ICN), Internet of Things (IoT), 5G/6G, molecular communication, etc.. He has been the Track Chair of VTC 2019, VTC 2020. He has been a Guest Editor of IEEE Transactions on Industrial Informatics, IEEE Sensors Journal, Sensors. He is an Associate Editor of the IEEE Access, IEEE Networking Letters.

Dr. James Xi Zheng, PhD in Software Engineering from UT Austin USA; Chief Solution Architect for Menulog Australia, now director of Intelligent systems research center (itsrc.org), deputy director of software engineering, global engagement, and assistant professor in Software Engineering at Macquarie University. Specialised in Service Computing, IoT Security and Reliability Analysis. Published more than 80 high quality publications in top journals and conferences. Awarded multiple best papers in leading conferences and a few highly competitive funding including ARC LP and Data61 CRP as the Lead Investigator.

Dr Xuyun Zhang is currently working as a senior lecturer in Department of Computing at Macquarie University in Australia. He worked as a lecturer in University of Auckland during 2016~2019, and a postdoc researcher in NICTA during 2014~2016. He got his Phd degree in Computer Science and Technology from University of Technology Sydney (UTS), Australia, and his Master’s and Bachelor’s degrees in the same major from Nanjing University, China. He is an early/mid-career researcher with an international track record of research in areas including scalable and secure machine learning, big data privacy and cyber security, big data mining and analytics, cloud/edge/service computing and IoT, etc. He has so far published over 100 refereed academic papers on many high-quality and influential conferences and journals in the abovementioned areas, such as IEEE TPDS, TKDE, TC, TSE, TII, JSAC and ICDE.