Light-PoEDDP: A Fast and Lightweight RSA-Based Proof of Possession of Outsourced Data & Correct Computation

Abdel Ali Harchaoui, Ali Younes, Abdelaaziz El Hibaoui
ECT Laboratory, Faculty of Science
Abdelmalek Essaadi University
Tetuan, Morocco
{a.harchaoui*, ayounes, aelhibaoui}@uae.ac.ma

Ahmed Bendahmane[1] , Abdellatif Machti [2]
LaSAD Laboratory, Ecole Normale Supérieure
Abdelmalek Essaadi University
Martil, Morocco
1. abendahman@uae.ac.ma
2. amachti@etu.uae.ac.ma

***Abstract*—Smart cities rely on seamless integration and communication of diverse sectors through high-end connectivity. Each urban sector is connected to a mesh of domain-specific devices that react to sector-centric data through different events such as capturing, transmitting, storing, training models, and performing inferences. Consequently, the data lifecycle and these events impact urban management and the lives of the citizens positively when sound events trigger correct actions at the right time using available resources. However, relying on these resource-restricted devices to handle critical computations (e.g., cryptographic operations) raises concerns regarding the integrity of the data that directly affects the soundness of these events and the correctness of the taken decisions. To this end, we introduce Light-PoEDDP, a lightweight integrity verification protocol. It leverages the cloud service provider to operate as a compute service of the outsourced computation and as a data store service. This protocol is an enhanced version of our previous protocol, Proof of Exponentiation of Dynamic Data Possession PoEDDP, which is based on RSA-Accumulators.**

***Index Terms*—RSA-Accumulator, Cloud Storage Security, Data Integrity Verification Protocol, Outsourced Computation, Proof of Exponentiation, Proof of Data Possession.**

I. Introduction

Smart cities leverage interconnected Internet of Things (IoT) devices to enhance urban efficiency and citizen well-being [1]–[4]. These devices, including sensors, actuators, smart meters, and gateways, form a network that collects and exchanges data to facilitate informed decision-making across various urban sectors. However, data exchange among these devices, city infrastructure, and cloud service providers introduces security concerns and potential risks. These risks encompass various aspects, including IoT identity verification and authorization, data integrity, confidentiality, and the correctness of outsourced computation [5]–[7].

To address these challenges, researchers have developed numerous schemes focusing on data encryption, aggregation, and outsourcing computation using different technologies such as blockchain [8] and protocols [9]. While some schemes prioritize ensuring the integrity and confidentiality of remotely stored data, others concentrate on outsourcing computation and verifying its integrity. Despite these efforts, and to the best of our knowledge and available resources, none of the existing schemes comprehensively address both data integrity and computation verifiability to construct a unified integrity scheme tailored for resource-restricted devices in the era of smart cities. This gap underscores the need for a novel approach that integrates both data integrity and correctness of outsourced computation to ensure the integrity of exchange data within smart cities.

In this paper, we present a fast and efficient data integrity verification protocol that incorporates outsourced computation, aiming to reduce the computational burden on resource-constrained IoT devices while maintaining the integrity and confidentiality of data exchanged in smart city environments. By addressing both data integrity and computation verifiability, our proposed protocol provides a comprehensive solution to the security challenges associated with data exchange in smart cities, considering the resource limitations of connected devices.

A. Contributions

This paper introduces $\mathsf{Light-PoEDDP}$, a fast and lightweight integrity verification scheme that ensures the correctness of both data and computation. This scheme, an imporoved version of our PoEDDP [10], enables low-power devices to delegate critical computations while securely storing data on the Cloud Service Provider (CSP), focusing on large integer multiplication and modular exponentiation. $\mathsf{Light-PoEDDP}$ offloads computation tasks to the CSP to mitigate the computation and storage challenges facing connected low-power devices, allowing them to focus on domain-specific missions. The scheme addresses the storage challenge by storing only a small fingerprint of a dynamic mapping used during the auditing phase. Our scheme ensures data integrity even if the CSP caches values to accelerate proof generation without holding the full version of the data. Additionally, we introduce **Trap Queries**, run at specific intervals, to verify that the CSP still possesses the data. This check adds an extra layer

of certainty to ensure the ongoing integrity and the availability of the outsourced data. Moreover, our scheme supports various characteristics of data integrity verification, including blockless verification, unrestricted challenge frequency, dynamic data handling, public auditability, and fairness. Furthermore, it ensures security against multiple data attacks, such as tag forgery attacks, data leakage attacks, replace attacks, replay attacks, and pollution attacks.

B. Related Works

In the literature of data auditing [9], several schemes have been proposed to address different aspects of outsourcing data to the CSP. These schemes aim to ensure the integrity and confidentiality of data stored and processed remotely. Conversely, some schemes focused on outsourcing computation and verifying its integrity [7]. This section provides an overview of the various approaches to data auditing, categorizing them into Static Auditing, Dynamic Auditing, Blockchain-based Auditing, and outsourced computation auditing.

1) **Static data Auditing** focus on verifying the integrity of data stored in the CSP at specific points in time without considering its dynamic nature. [11] introduces a signature-based architecture for cloud data integrity verification. However, it relies on random masking techniques and does not provide support for auditing in multi-replica CSP environments. [12] Presents a provable data integrity scheme that limits incremental fingerprinting and ensures data file integrity post-fingerprinting. [13] Scalable Provable Data Possession technique supports specific operations but faces constraints with updates and challenges, limiting its feasibility for large files.
2) **Dynamic data Auditing** schemes extend the concept of static auditing by enabling continuous monitoring and verification of data integrity over time. These schemes allow for the detection of unauthorized modifications or deletions of data in real-time, providing enhanced security and accountability. [14] Presents an innovative Merkle Hash Tree (MHT)-based public auditing scheme termed RITS-MHT, leveraging Third-Party Auditors (TPA). However, the reliance on TPAs introduces security concerns regarding their trustworthiness. Writers of [15] [16] examine dynamic data modification through a traditional Merkle Hash Tree construction, highlighting the potential risk of data corruption by the auditor. [17] leverages an algebraic structure to construct a variant of a one-way RSA-Accumulator introducing a non-coprime representation of data blocks. This technique aims to circumvent primality tests and collisions, and to prevent the CSP from forging proofs without possessing the data. However, the scheme encounters significant computational overhead during both the block generation and proof generation phases, primarily due to the extensive length of the generated blocks involved in large integer multiplication and modular exponentiation processes. In comparison, our scheme, presented in [10], addresses the limitations of [17] by improving both the block generation and proof generation times by a factor of 20.
3) **Blockchain-based data Auditing** leverages the decentralized and immutable nature of blockchain technology to ensure the integrity and transparency of data transactions. By recording data audit trails on a distributed ledger, these schemes offer tamper-proof audit logs and enable trustless verification of data integrity [18] [19] [20] [21] [22] [23]. [24] investigate range of approaches, encompassing encrypted storage, off-chain data storage coupled with on-chain hashes, and role-based access control. [25] proposes a zero-knowledge proofs technique to enhance auditability. [26] introduces a smart-contract-based auditing scheme that eliminates the need for a TPA but incurs storage overheads. [21] proposes the off-chain storage-based auditing approach to addresses rational behavior assumptions but necessitates additional security measures for data transfer and encounters challenges related to network latency. [22] solution achieves both data migration and integrity checking simultaneously without relying on a TPA. This approach mitigates service interruptions and potential privacy breaches resulting from the single-point-of-failure associated with TPAs.
4) **Outsourced Computation Auditing** schemes focus on verifying the correctness and integrity of computations outsourced to CSP. These schemes allow data owners to delegate computationally intensive tasks while ensuring the accuracy and security of the results. [27] proposes the an efficient outsource-secure algorithm for simultaneous modular exponentiations for (variable-exponent, variable-base) modulo a prime in the two untrusted program model. [28] presents a novel outsourcing algorithm for modular exponentiation within the framework of two non-colluding CSPs. This algorithm ensures the privacy of both the base and the power of the outsourced data. [29] proposes a secure two outsourcing schemes for modular exponentiation using a single server called MExp and M2Exp. The former, ensures privacy during outsourcing through a logical division method and mitigates collusion attacks typical in schemes utilizing two untrusted servers for single-server-based schemes. The latter, facilitates multiple modular exponentiation. [23] achieves the same outsourcing of modular exponentiation using smart contracts. [30] presents two outsourcing algorithms for modular exponentiation. The first algorithm utilizes two untrusted servers, allowing the outsourcer to detect errors with a probability of 1 using Euler's theorem if either server misbehaves. The second algorithm, designed for a single server, maintains a checkability probability of 1 for error detection. [31] presents two privacy-preserving outsourcing algorithms specifically tailored for smart meter devices engaging in multi-dimensional data aggregation.

C. PAPER ORGANIZATION

Section II provides the reader with the necessary background, definitions, and notations used throughout the paper.

Sections III and IV delve into the building blocks of our proposed scheme. Section V focuses on the key characteristics of our scheme. Section VI presents a performance analysis, including a discussion of the scheme's costs. Finally, Section VII concludes the paper.

II. Preliminaries

A. Notations

This section highlight the adopted notations, useful definitions, and important concepts.

- $\mathsf{negl}[\lambda]$ is a negligible function of the security parameter λ.
- $Primes(\lambda)$ is the set of odd primes less than 2^λ
- $\chi \xleftarrow{\$} \mathbb{G}$ denotes a uniform sampling of the element χ from the set $\mathbb{G}$.
- $\chi \xleftarrow{\$} \mathcal{A}(.)$ a randomized algorithm $\mathcal{A}$ which produces a random variable χ.
- $GGen(\lambda)$ is a randomized algorithm that generates either a group of known order or a group of unknown order.
- $KeyGen(\lambda)$ a key generation algorithm which produces the encryption key sk and public key pk.

B. Definitions

- RSA **Group**: We define the set of all invertible elements relatively prime to N as an RSA group, denoted by $\mathbb{Z}_N^*$. Here, N is the product of two large primes p and q. We refer to g as a generator, such that there exists an element g in $\mathbb{Z}_N^*$ where $\mathbb{Z}_N^* = \{g_{i=0}^{n-2} | g^{n-1} = (1 \mod N)\}$. Additionally, we define the RSA quotient group $\mathbb{Z}_N^*/\{\pm 1\}$, in which every element x has its corresponding element $N - x$. In this quotient group, no element x has a known order, except for the identity element.
- **Collision-resistant hash function** : Let $H : \{0,1\}^* \to \{0,1\}^\lambda$ be a hashing function, we say that H is collision-resistant if the advantage of a PPT adversary $\mathcal{A}$ to find $x_1 \neq x_2 \wedge H(x_1) = H(x_2)$ is negligible.
- **Division-intractable hashing function**: Let $H : \{0,1\}^\lambda \to \{0,1\}^\lambda$ be a collision-resistant hash function, H is division-intractable if the advantage of PPT adversary $\mathcal{A}$ to find and element x_j and a set of $\{x_i\} \in \{0,1\}^\lambda$ and $H(x_j)$ divides $\prod_{i=1}^n H(x_i)$ is negligible.
- RSA-**Accumulators**:
  - Let $\mathcal{T}$ be a set of n-unique odd prime τ_i of l-bits $\tau_i \in \{0,1\}^l$, and $H : \{0,1\}^* \to \{0,1\}^\lambda$ be a division-intractable hashing function, such that $\tau_i = H(x_i)$.
  - $\mathbb{G}$ an RSA quotient group with a random element $g \xleftarrow{\$} \mathbb{G}$.

  An RSA-Accumulator is a commitment scheme $(Commit, Open, Verify)$ used by a sender denoted by V to bind the elements τ's of $\mathcal{T}$ to a digest $h = g^{\prod_{i=1}^n \tau_i} \in \mathbb{G}$ by running the algorithm $h \leftarrow commit(\mathcal{T})$. Then, it asks a prover denoted by P to issue a proof π that an element τ_s belongs to the set, using the algorithm $\pi \leftarrow open(\tau_s)$. This allows V to verify the claim that $\tau_s \in \mathcal{T}$ holds using the verify algorithm $\{0,1\} \leftarrow verify(\pi)$. This positive claim, indicating that an element $\tau_s \in \mathcal{T}$ is referred to as a *proof of membership* or *proof of inclusion*.Additionally, RSA-Accumulators support negative claims, indicating that an element $\tau_s \notin \mathcal{T}$, known as a *proof of non-membership* or *proof of exclusion*.
- **Proof of Exponentiation** The Proof of Exponentiation (PoE) protocol enables a prover to demonstrate to a verifier that they possess the correct value α such that $\alpha = g^\tau$, where both g and τ are known to the verifier. This protocol provides efficient verification for computationally intensive tasks, such as iterated exponentiation, and allows for the outsourcing of these tasks when the verifier has limited resources.
  In most RSA-Accumulators, iterated exponentiation is the standard approach for computing the commitment A_X and the proof π_k for a given element x_k. However, this method is both time and resource-intensive. The PoE protocol addresses this issue by utilizing a shortcut that enables resource-constrained verifiers to efficiently confirm the correct computation of the exponentiation without having to perform the entire computation themselves.

III. CONSTRUCTIONS

Our proposed scheme is built upon a generalization [32] of the Proof of Exponentiation PoE protocol introduced by Wesolowski [33], which we denote as Light Proof of Exponentiation of Dynamic Data Possession ($\mathsf{Light-PoEDDP}$). This scheme leverages an RSA-Accumulator and employs a Hashing-to-prime technique to support dynamic data and enable fast proof generation and verification while reducing the size of the blocks. The $\mathsf{Light-PoEDDP}$ scheme imposes minimal computation and storage overhead on connected devices by outsourcing critical tasks, as shown in Table II. Key features of the $\mathsf{Light-PoEDDP}$ scheme include public auditability, blockless verification, unbonded queries, and robustness using **Trap Queries**. It fulfills the soundness property and ensures fairness against dishonest verifiers. In the following section, we outline the building blocks of the $\mathsf{Light-PoEDDP}$ scheme and provide a detailed explanation of the procedures used.

A. Phases

1) Setup phase: Given a security parameter λ, this phase creates the necessary environment for the protocol. It executes $\mathsf{GGen}(\lambda)$ algorithm to produce the public parameters. The first parameter is a group $\mathbb{G}$ that could be of known order $N = p \cdot q$ such that p and q are two relatively big primes (i.e., $\mathbb{Z}_{\mathbb{N}}^*$). Once $\mathbb{G}$ is determined, we randomly sample the second parameter g from $\mathbb{G}$ used as a group generator and to instantiate our accumulator. To (de)encrypt data with a secure Enc cipher, this phase generates the pair of public and verification keys $(\mathsf{pk}, \mathsf{vk})$ using the keyGen algorithm. Finally, we instantiate a mapping of the hashes-to-prime called HPM, which consists of a dynamic indexed mapping used to track some elements

involved to generate the accumulated value α_t in a given time t.

2) Preprocessing phase: During this phase, the data owner V preprocesses the collected data to be outsourced to CSP denoted as P for prover. He splits this data $\mathcal{D}$ into n-blocks f_i of size l_1 using split procedures. If the final block f_n is not a multiple of l_1, V pads this block accordingly. We denote by $\mathcal{F} = \{f_i\}_{i=1}^{n}$. Subsequently, he encrypts each block f_i, using a secure block cipher Enc (i.e., AES) with the vk key to produce the encrypted blocks c_i of size l_2. We denote $\mathcal{C} = \{c_i\}_{i=1}^{n}$ the set of the encrypted blocks. Then, he selects a number of elements $x \geq 2$ of ciphertexts c_i at random and applies the $\mathsf{H}_{\mathsf{prime}}$ algorithm to produce the hashes-to-prime τ_i of size $l3$. These values are saved into the mapping HPM to be used during the auditing phase. Finally, V outsources the encrypted data $\mathcal{C}$ to P concatenated with the public parameters $\mathbb{G}$ and g as $(\mathcal{C} \mid\mid \mathbb{G} \mid\mid g)$, using a secure channel (i.e., TLS 1.3). Then, he asks the CSP to perform the heavy computation part that consists of hashing to prime the set of ciphertexts $\mathcal{C}$ to create the set of the hashes-to-prime $\mathcal{T}'$. The elements of this set are used to perform modular multiplications and exponentiation to compute the accumulated value α_t which equals to $g^{\prod_{i=1}^{n} \tau_i'}$ such that $\tau_i' \in \mathcal{T}'$. Once the computation done, the CSP sends the value of α_t to data owner.

3) Auditing phase: During this phase, V initiates the protocol to verify that P correctly computes the public accumulated value $\alpha_t = g^{\prod_{i=1}^{n} \tau_i'}$, holds the full version of the outsourced data and maintains its integrity. He will use his secret knowledge of the random x-elements previously chosen to check the correct computation of α_t that he owned.

Firstly, he selects a prime from $\left[0, 2^{\lambda}\right]$ at random; $l \xleftarrow{\$} \mathsf{Primes}(\lambda)$ and two random indices (j, k) from HPM and sends the concatenation $(l \mid\mid j \mid\mid k)$ as a challenge to the CSP.

Upon the CSP received $(l \mid\mid j \mid\mid k)$, he searches for the two ciphertexts of the indices (j, k) (i.e., c_j' and c_k'), using the LookupBlock() procedure. Then, he tries to find a pair $(q, r) \in \mathbb{Z}_{\mathbb{N}} \times [l]$ such that $\prod_{i=1}^{n} \tau_i' \times \tau_j'^{-1} = q \cdot l + r$.

Once finished, he sends the concatenation $(\pi \mid\mid c_j' \mid\mid c_k' \mid\mid r)$ to the data owner. This concatenation consists of the proof of data possession $\pi = g^q$, the current witness c_j', the next witness c_k', and the remainder r.

Upon V received $(\pi \mid\mid c_j' \mid\mid c_k' \mid\mid r)$, he hashes to prime c_j' to obtain τ_j'. The value τ_j' is then checked against his local version τ_j. This step ensures with a high probability that the CSP stores the data. Then, he checks if $\pi^l g^r$ is equals to $\alpha_t^{\tau_j'^{-1}}$. If this condition holds, the data owner confirms the correct computation of the accumulated value α_t. Finally, he will store the hash-to-prime τ_k' of the next witness c_k' on the HPM mapping and delete his local version of the data.

IV. Data Dynamics Support

Smart cities are built upon the propagation and transformative events affecting the data across the various integrated IoTs across cities' sectors to trigger proactive actions to enhance urban management. Thus, this dynamic environment requires a scheme that handles the integrity of the data at rest and in transit. And has to ensure the correct outsourced computing from resource-restricted devices. Consequently, our scheme leverages the necessary mechanism to ensure the security of the dynamic nature of the data without sacrificing efficiency. Our approach to handle dynamic data is straightforward. We require the data owner to send a set of m-blocks to the remote CSP Fig. 1.

Let $\mathcal{D}_m$ be the set of the new unique m-blocks, that the data owner wanted to outsource, $\mathcal{F}_m$ be the set of the m-padded blocks of size l_1 of $\mathcal{D}_m$, $\mathcal{C}_m$ be the set of the m-ciphertexts of elements of $\mathcal{F}_m$, and $\mathcal{T}_m$ be the set of the m-hashes-to-prime created from each element of $\mathcal{C}_m$.

A. Robust auditing and Trap Queries

Given the various potential points of failure, such as unintentional issues like hardware and software bugs and intentional threats like viewing the CSP as malicious, relying solely on the trustworthiness of the CSP to assert the correctness of outsourced computing (i.e., accumulated value α_t) and data integrity poses a significant security risk to our protocol. Since our scheme is transparent to all parties including the CSP, who controls all the parameters involved in computing the accumulated value, he may act maliciously and mount various vectors to undermine the soundness and robustness of our protocol. Therefore, we consider the CSP as an active adversary with unlimited resources, particularly in the context of pre-quantum computing. With malicious behavior, the CSP may invest computational power upfront to learn the set x of witnesses held by the data owner, potentially deleting data and forging arbitrary values without being detected.

To enhance the robustness of our scheme, we propose simple mechanisms to strengthen its guarantees. First, the data owner is asked to retain his version of the data until he verifies the correct computation of α_t during the first audit. Second, we introduce the **Trap Queries** mechanism to ensure that the CSP holds and does not tamper with the data. This mechanism generates a single trap block of data, c_trap, with a hash-to-prime value τ_trap, which is sent to the CSP to be added using the add block procedure. The data owner then challenges the CSP with a regular new prime and two indices (u, z) as $(l \mid\mid u \mid\mid z)$. The CSP must respond with the correct proof $(\pi \mid\mid c\_u' \mid\mid c\_z' \mid\mid r)$, which is used to validate that he genuinely holds the data. Later, this trap block is deleted using the delete procedure with similar challenge. This mechanism is triggered once the data owner exceeds the number of stored elements x.

V. Characteristics of Light − PoEDDP scheme

Ensuring the integrity of both data and computations is crucial for maintaining the reliability and security of cloud-based services. This section outlines the key characteristics of our integrity verification scheme, prioritizing data integrity with a specific focus on computation correctness, especially in cryptographic computations such as modular multiplication

Preprocessing:
The data owner V prepares a subset of data $\mathcal{D}_m$ to add, and the computation to outsource:

1) he splits $\mathcal{D}_m$ into blocks f_j of size l_1; $\mathcal{F}_m \leftarrow \mathcal{D}_m.split(l_1)$ and pads the final block if necessary,
2) encrypts each block f_j into c_j; $c_j \leftarrow \mathsf{Enc}(\mathsf{vk}, f_j)$ such that $f_j \in \mathcal{F}_m$ and $c_i \in \mathcal{C}$ where $j = \{1, \ldots, m\}$,
3) hashes to prime each block c_j into τ_j; $\tau_j \leftarrow \mathsf{H}_{\mathsf{prime}}(c_j)$ such that $\tau_j \in \mathcal{T}_m$,
4) selects new random x-elements c_i from $\mathcal{C}_m$ such that $x \geq 2$,
5) hashes to prime these x-elements c_i into τ_i; $\tau_i \leftarrow \mathsf{H}_{\mathsf{prime}}(c_i)$,
6) replaces some old elements from HPM with new x-elements,
7) outsources the encrypted data ($\mathcal{C}_m$) to P,
8) asks P to compute $\alpha_{t+1} \leftarrow \alpha_t^{\prod_{j=1}^{m} \tau'_j}$ such that $\tau'_i \in \mathcal{T}'_m$ over the new set $\mathcal{C} \cup \mathcal{C}_m$
9) request the value α_{t+1} and stores it to perfom later audits.

Challenge:

1) sends the concatenation ($l \mid\mid k \mid\mid s$) of random $l \xleftarrow{\$} \mathsf{Primes}(\lambda)$ and two random indices (k, s) from HPM.

ProofGen:
Once P receives ($l \mid\mid k \mid\mid s$):

1) he finds $c'_k \leftarrow \mathsf{LookupBlock}(\mathcal{C}, k)$ of index k,
2) he finds $c'_s \leftarrow \mathsf{LookupBlock}(\mathcal{C}, s)$ of index s,
3) inserts $\mathcal{C}_m$ into the correct m positions in $\mathcal{C}$,
4) tries to find $(q, r) \in \mathbb{Z}_{\mathbb{N}} \times [l]$ such that $\left(\prod_{i=1}^{n} \tau'_i\right) \cdot \left(\prod_{j=1}^{m} \tau'_j\right) \cdot \tau_k'^{-1} = q \cdot l + r$,
5) produces the proof $\pi \leftarrow g^q$, and sends the concatenation ($\pi \mid\mid c'_k \mid\mid c'_s \mid\mid r$) to V.

ProofVerify:
Once V receives ($\pi \mid\mid c'_k \mid\mid c'_s \mid\mid r$),

1) he hashes to prime c'_k; $\tau'_k \leftarrow \mathsf{H}_{\mathsf{prime}}(c'_k)$,
2) verifies and accepts if: $\begin{cases} \tau_k = \tau'_k \\ \pi^l g^r = \alpha_{t+1}^{\tau_k'^{-1}} \end{cases}$
3) hashes to prime c'_s; $\tau'_s \leftarrow \mathsf{H}_{\mathsf{prime}}(c'_s)$ and stores τ'_s in HPM,
4) deletes $\mathcal{D}_m$.

Fig. 1. m-blocks addition

and exponentiation. The following subsections delve into the specific data integrity characteristics [34] and introduce characteristics related to computation correctness.

A. Light − PoEDDP *Data Integrity characteristics*

1) **Blockless Verification**: Our protocol eliminates the necessity for the data owner to download the entire dataset from the remote storage for integrity verification. Instead, the data owner issues a challenge to the CSP using randomly selected prime numbers and random indices of the hashes-to-prime values stored in HPM. Subsequently, the CSP generates proofs and provides the required witnesses to pass the audit.
2) **Unrestricted Challenge Frequency**: Our challenging mechanism relies on the number x elements stored in HPM and on **Trap Queries**. To make sure that the CSP holds the data. It uses also, the set of odd primes less than 2^λ such that λ is the chosen security parameter (i.e. 128, 192, or 256).
3) **Batch Auditing**: Our scheme support batch verification checks of multiple queries at once. These queries could be bound to multiple connected devices (clients) against one data store.
4) **Soundness**: Our scheme allows the CSP to cache any amount of information that could help him accelerate the computing (i.e., $\prod_{i=1}^{n} \tau'_i$ and storing a version of HPM). This fact couldn't help him pass the audits without holding all the data, including the correct witnesses, and computing correct proof for each challenge query.
5) **Stateless Verification**: Our scheme relies on fresh challenges for each query including the **Trap Queries**. Each query consists of unique odd prime l from $[1, 2^\lambda]$, and two indices from HPM when the number of queries is less than x and data owner won't to lunch the **Trap Queries** mechanism. The CSP willing store challenges' states won't help him pass the audit, but will only cost him storage.
6) **Robustness**: Minor tampering or data deletion will be immediately detected by the V, who received the proof π concatenated with current block $c_j{}'$ used to check whether $\tau'_j \leftarrow \mathsf{H}_{\mathsf{prime}}(c'_j)$ equals to the stored value τ_j on HPM. Upon exceeding the number of stored elements x, the data owner initiates the **Trap Query** mechanism to enhance the robustness of the data integrity verification process.
7) **Data Recovery**: Our scheme is a PDP scheme but can easily support data recovery, using any error-correcting codes during the preprocessing phase to embed the necessary metadata.
8) **Dynamic Data Handling**: Our scheme supports dynamic data operations to respond to the environment in which it operates. Data owners can perform any data operation on the cloud storage without downloading the data locally or altering the protocol. They can also check the integrity of the operations and spot faulty ones.
9) **Public Auditability**: Due to the nature of our resources-restricted protocol that relies on the CSP to perform

most of the computations, It's easy to involve any TPA to perform the blockless verification on behalf of the resource-restricted device, using any process of public auditing integration [35] [36].

10) **Confidentiality**: In our protocol, the data owner encrypts the data using a secret key vk during the preprocessing phase. It relies on a secure cipher that provides Indistinguishability under Chosen Ciphertext Attack ($\mathsf{IND-CCA}$) [37] to safeguard data confidentiality even in the event of a data breach. However, a drawback of our scheme is that the client must protect the secret key vk.
11) **Privacy-Preserving**: Our scheme does not leak information during the data integrity checks to the CSP or the TPA.
12) **Fairness**: To protect the reputation of reliable and honest CSPs against dishonest data owners, our scheme supports fairness characteristics by implementing any standard fairness mechanism as in [38].
13) **Light Weight**: Our scheme minimizes computation and storage overhead on the client side by outsourcing most of the computation to the CSP and storing only minimal metadata for data verification.

B. Light − PoEDDP *Computation Correctness Characteristics*

A scheme that delegates computationally intensive tasks to a CSP or to a third-party compute service must implement a verifiable computation mechanism to guarantee that the offloaded computation was done correctly. Light − PoEDDP takes care of the computation cost during the construction of the accumulated value, which involves modular multiplication and exponentiation of large integers and outsourced it to the CSP. It provides the following characteristics:

1) **Correctness**: It guarantees that the offloaded task is executed correctly.
2) **Verifiability**: Our scheme allows the verification of computation results by a trusted, untrusted or semi-trusted third party and by the data owner.
3) **Efficiency**: Our scheme is efficient in terms of computational overhead and communication costs for resources-constrained devices.
4) **Transparency**: Our scheme is transparent in terms of setup and algorithms. All parties know how and what they execute.
5) **Resilience**: Our scheme is resilient against failures, errors, or malicious activities due to the implemented integrity mechanism.
6) **Flexibility**: Our scheme lacks flexibility for different types of computations. This version tackles the offloading of modular multiplication and exponentiation only.

VI. Performance Analysis & Discussion

In this section, we assess the computation, communication, and storage expenses accrued by the involved parties using the same notation as in [17]. We select Khadr's scheme as a candidate due to its similarity to our proposed scheme, particularly in both being RSA-based Accumulator schemes. Table I outlines the notation used to quantify these metrics such as • ($s1 = |\tau| = |l| = |r| = 128$-bits) represents the size of tags, the prime challenge, and the remainder of the division modulo l, respectively. • ($s2 = |N| = |q| = 3072$-bits) represents the size of the RSA-moduli, and the quotient of the division modulo l,respectively. • ($s3 = |\phi(N)| = 3071$-bits) represents the of the Euler's phi function, • ($s4 = |b_i^{''}| = 3327$-bits) represents the size of the generated block in Khadr's scheme..

TABLE I
Notations of operation costs.

Symbol	Operation cost
T_{muls1}	modular integer multiplication with s1 bits operands
T_{muls4}	modular integer multiplication with s4 bits operands
T_{exps2}	modular exponentiation s2 bit exponent
T_{exps4}	modular exponentiation s4 bit exponent
T_{div}	modular division
T_{inv}	modular inverse
T_{rand}	random number sampling
T_{Rprime}	random prime sampling
T_{hash}	hashing
$T_{\mathsf{H_{prime}}}$	Hash-to-Prime function
T_{Enc}	encryption
T_{lookup}	lookup operation
T_{setup}	setup cost
T_{pre}	Preprocessing cost
T_{aud}	audit cost
T_{chal}	challenge cost
T_{pGen}	proof generation
T_{pVer}	proof verification
T_{comp}	comparison operation

A. Computation Cost

Our protocol consists of three phases: setup, preprocessing, and auditing, with the setup phase being a one-time operation such that

1) The setup phase takes $T_{setup} = 2T_{Rprime} + T_{muls2} + T_{rand}$.
2) The preprocessing phase takes $T_{pre} = nT_{\mathsf{Enc}} + xT_{\mathsf{H_{prime}}}$ such that n is the number of data blocks and x is the selected number of hashes-to-prime to store in HPM mapping.
3) The auditing phase takes $T_{aud} = T_{chal} + T_{pGen} + T_{pVer}$ such that $T_{chal} = T_{Rprime}$, $T_{pGen} = T_{lookup} + (n-1)T_{muls1} + T_{div} + T_{exps2}$ and $T_{pVer} = T_{\mathsf{H_{prime}}} + (n-1)T_{muls1} + T_{div} + T_{inv} + 3T_{exps1}$.

For a concise performance comparison detailed in Table II with the candidate scheme, we retained tasks that require substantial computational resources while excluding others mentioned earlier. Additionally, we merged the setup and preprocessing phases, referring to them collectively as the setup phase.

B. Communication Cost

During the outsourcing phase, the data owner sends the public parameters ($\mathbb{G}$,g) and the encrypted data $\mathcal{C}$ and receives the computation of α_t. Then, he sends the challenge represented as

the following concatenation ($l \mid\mid j \mid\mid k$) and receives the proof ($\pi \mid\mid c'_j \mid\mid c'_k \mid\mid r$). Thus, the overall communication during the auditing phase is $(l_h + 2\left(|j| + |N| + |c'_j|\right) + |\pi| + |r|)$.

C. Storage Cost

Our proposed scheme significantly reduces the storage overhead for the data owner. Instead of storing n hashes-to-prime, it only needs to store a small subset of x hashes-to-prime, where this subset is much smaller than the total number of data blocks n ($x \lll n$). Additionally, the scheme requires storing the public parameters ($\mathbb{G}$,g), the (de)encryption keys (vk,pk), and the accumulated value α_t. Table II shows a performance comparison between our Light − PoEDDP scheme and khadr's scheme [17].

D. Discussion

In Khadr et al.'s scheme, each block captured by the resource-restricted device undergoes encryption, hashing, and concatenation with the corresponding tag. Subsequently, a non-coprime representation is constructed by left-shifting this concatenation by one position, resulting in a block size of 3327 bits. This block is then used in modular large integer multiplications, where the modulus $\phi(N)$ is 3071 bits in size. Performing n modular large integer multiplications, of 3327 bits each, represents an intensive computational task on the resource-restricted devices, particularly when n is moderate to large. The modular large integer multiplications is required to compute the accumulated value used during the auditing phase. In contrast, our proposed scheme encrypts each block captured by the resource-restricted device and then outsources the final set of n encrypted blocks to the CSP. Leveraging the computational resources of the CSP to compute the public digest α_t, our scheme effectively alleviates the computational burden on the data owner, leading to improved efficiency. Additionally, the hash-to-prime representation of size 128 bits imposes minimal computational load on the CSP during the modular large integer multiplications. On the other hand, our scheme stores x hashes-to-prime of size 128 bits, where $x \lll n$, compared to Khadr's scheme, which stores n tags of size 64 bits, resulting in higher storage overhead.

VII. Conclusion

In conclusion, this paper presented a fast lightweight integrity verification protocol for smart city environments, addressing the resource constraints of IoT devices. Our proposed protocol, Light − PoEDDP, integrates data integrity checks and verifiable outsourced computation. It supports dynamic data while providing fast proof generation and verification. By leveraging a variant of an RSA-Accumulator built upon the Proof of Exponentiation protocol and the Hash-to-prime and **Trap Queries** technique, our scheme introduces minimal computation and storage overhead on connected devices. Furthermore, Light − PoEDDP ensures public auditability, blockless verification, unbonded queries, and robustness, fulfilling soundness and fairness properties. The performance analysis demonstrated the efficiency and practicality of our scheme in enhancing data integrity and computation verifiability in smart city applications. Future work may explore the integration of post-quantum ciphers, to further strengthen the security of smart city environments.

References

[1] M. Mazhar Rathore, Awais Ahmad, Anand Paul, and Seungmin Rho. Urban planning and building smart cities based on the Internet of Things using Big Data analytics. *Computer Networks*, 101:63–80, June 2016.

[2] Simon Elias Bibri, John Krogstie, Amin Kaboli, and Alexandre Alahi. Smarter eco-cities and their leading-edge artificial intelligence of things solutions for environmental sustainability: A comprehensive systematic review. *Environmental Science and Ecotechnology*, 19:100330, May 2024.

[3] Pitchai Pandiyan, Subramanian Saravanan, Kothandaraman Usha, Raju Kannadasan, Mohammed H. Alsharif, and Mun-Kyeom Kim. Technological advancements toward smart energy management in smart cities. *Energy Reports*, 10:648–677, November 2023.

[4] Amin Ullah, Syed Myhammad Anwar, Jianqiang Li, Lubna Nadeem, Tariq Mahmood, Amjad Rehman, and Tanzila Saba. Smart cities: the role of Internet of Things and machine learning in realizing a data-centric smart environment. *Complex & Intelligent Systems*, 10(1):1607–1637, February 2024. Company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 1 Publisher: Springer International Publishing.

[5] Thiago Poleto, Thyago Celso Cavalcante Nepomuceno, Victor Diogho Heuer de Carvalho, Ligiane Cristina Braga de Oliveira Friaes, Rodrigo Cleiton Paiva de Oliveira, and Ciro José Jardim Figueiredo. Information Security Applications in Smart Cities: A Bibliometric Analysis of Emerging Research. *Future Internet*, 15(12):393, December 2023. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[6] Kyriaki Tsantikidou and Nicolas Sklavos. Threats, Attacks, and Cryptography Frameworks of Cybersecurity in Critical Infrastructures. *Cryptography*, 8(1):7, March 2024. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[7] Xixun Yu, Zheng Yan, and Athanasios V. Vasilakos. A Survey of Verifiable Computation. *Mobile Networks and Applications*, 22(3):438–453, June 2017.

[8] Jong Hyuk Park, Sushil Kumar Singh, Abir EL Azzaoui, Choo Kim-Kwang Raymond, and Laurence Tianruo Yang. A Comprehensive Survey on Blockchain for Secure IoT-enabled Smart City beyond 5G: Approaches, Processes, Challenges, and Opportunities. *Human-centric Computing and Information Sciences*, 13(0):15–55, November 2023.

[9] Mai Rady, Tamer Abdelkader, and Rasha Ismail. Integrity and Confidentiality in Cloud Outsourced Data. *Ain Shams Engineering Journal*, 10(2):275–285, June 2019.

[10] Abdel Ali Harchaoui, Ali Younes, Abdelaaziz El Hibaoui, and Ahmed Bendahmane. Poeddp-a fast rsa-based proof of possession accumulator of dynamic data on the cloud. *IEEE Access*, 12:52878–52901, 2024.

[11] Hongliang Zhu, Ying Yuan, Yuling Chen, Yaxing Zha, Wanying Xi, Bin Jia, and Yang Xin. A Secure and Efficient Data Integrity Verification Scheme for Cloud-IoT Based on Short Signature. *IEEE Access*, 7:90036–90044, 2019. Conference Name: IEEE Access.

[12] Ke Zeng. Publicly Verifiable Remote Data Integrity. In *Information and Communications Security: 10th International Conference, ICICS 2008 Birmingham, UK, October 20 - 22, 2008 Proceedings*, pages 419–434, Berlin, Heidelberg, October 2008. Springer-Verlag.

[13] Giuseppe Ateniese, Roberto Di Pietro, Luigi V. Mancini, and Gene Tsudik. Scalable and Efficient Provable Data Possession, 2008. Publication info: Published elsewhere. Unknown where it was published.

[14] Neenu Garg and Seema Bawa. RITS-MHT: Relative indexed and time stamped Merkle hash tree based data auditing protocol for cloud computing. *Journal of Network and Computer Applications*, 84:1–13, April 2017.

[15] Qian Wang, Cong Wang, Jin Li, Kui Ren, and Wenjing Lou. Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing. In Michael Backes and Peng Ning, editors, *Computer Security – ESORICS 2009*, volume 5789, pages 355–370. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. Series Title: Lecture Notes in Computer Science.

TABLE II
PERFORMANCE COMPARISON BETWEEN OUR PoEDDP SCHEME AND KHADR ET AL.'S SCHEME

Costs		Our Light − PoEDDP Scheme	Khadr et al.'s scheme [17][a]
Computation	Setup	$nT_{\mathsf{Enc}} + xT_{\mathsf{H_{prime}}}$ (s.t $x \lll n$)	$nT_{\mathsf{Enc}} + nT_{hash} + nT_{muls4} + T_{exps2}$
	Challenge Gen	T_{rand}	T_{rand}
	Proof Gen	$(n-1)T_{muls1} + T_{exps2}$	$(n-1)T_{muls4} + T_{exps4}$
	Proof Ver	$3T_{exps1}$	T_{exps4}
Communication		2 ($\|N\| + \|j\| + \|c_j\|$) + $\|\alpha_t\|$	$\|N\| + \|j\| + \|\phi(N)\| + \|w_j\| + \|b_j^{''}\|$
Storage	Data owner	$x\|l_h\| + \|N\| + \|g\| + \|\alpha_t\| + \|\mathsf{pk}\| + \|\mathsf{vk}\|$ (s.t. $x \lll n$)	$n\|l_h\| + \|N\| + \|g\| + \|\phi(N)\| + \|acc_t\| + \|K_E\| + \|K_H\|$
	CSP	$\|N\| + \|g\| + \|\mathcal{C}\|$	$\|N\| + \|g\| + \|\phi(N)\| + \|\{b_i^{''}\}_{i=1}^{n}\|$

[a]: The notation used in [17] are; K_E encryption key, K_H hashing key, $\phi(N)$ Euler's phi function, acc_t accumulated value (digest), $b_i^{''}$ generated block, l_h length of the hashed block, w_j j-th witness

[16] Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, and Jin Li. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems*, 22(5):847–859, May 2011. Conference Name: IEEE Transactions on Parallel and Distributed Systems.

[17] W. I. Khedr, H. M. Khater, and E. R. Mohamed. Cryptographic Accumulator-Based Scheme for Critical Data Integrity Verification in Cloud Storage. *IEEE Access*, 7:65635–65651, 2019. Conference Name: IEEE Access.

[18] Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi, and Simon Duquennoy. Towards Blockchain-based Auditable Storage and Sharing of IoT Data. In *Proceedings of the 2017 on Cloud Computing Security Workshop*, CCSW '17, pages 45–50, New York, NY, USA, November 2017. Association for Computing Machinery.

[19] Jiaxing Li, Jigang Wu, Guiyuan Jiang, and Thambipillai Srikanthan. Blockchain-based public auditing for big data in cloud storage. *Information Processing & Management*, 57(6):102382, November 2020.

[20] Pedro W. Abreu, Manuela Aparicio, and Carlos J. Costa. Blockchain technology in the auditing environment. In *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6, June 2018.

[21] Danilo Francati, Giuseppe Ateniese, Abdoulaye Faye, Andrea Maria Milazzo, Angelo Massimo Perillo, Luca Schiatti, and Giuseppe Giordano. Audita: A Blockchain-based Auditing Framework for Off-chain Storage. In *Proceedings of the Ninth International Workshop on Security in Blockchain and Cloud Computing*, SBC '21, pages 5–10, New York, NY, USA, May 2021. Association for Computing Machinery.

[22] Changsong Yang, Feng Zhao, Xiaoling Tao, and Yong Wang. Publicly verifiable outsourced data migration scheme supporting efficient integrity checking. *Journal of Network and Computer Applications*, 192:103184, October 2021.

[23] Danting Xu, Yanli Ren, Xiangyu Li, and Guorui Feng. Efficient and Secure Outsourcing of Modular Exponentiation Based on Smart Contract. *International Journal of Network Security*, 22(6):934–944, November 2020.

[24] Fran Casino, Eugenia Politou, Efthimios Alepis, and Constantinos Patsakis. Immutability and Decentralized Storage: An Analysis of Emerging Threats. *IEEE Access*, 8:4737–4744, 2020. Conference Name: IEEE Access.

[25] Shiwei Xu, Xiaowen Cai, Yizhi Zhao, Zhengwei Ren, Le Du, Qin Wang, and Jianying Zhou. zkrpChain: Towards multi-party privacy-preserving data auditing for consortium blockchains based on zero-knowledge range proofs. *Future Generation Computer Systems*, 128:490–504, March 2022.

[26] Cheng Zhang, Yang Xu, Yupeng Hu, Jiajing Wu, Ju Ren, and Yaoxue Zhang. A Blockchain-Based Multi-Cloud Storage Data Auditing Scheme to Locate Faults. *IEEE Transactions on Cloud Computing*, 10(4):2252–2263, October 2022. Conference Name: IEEE Transactions on Cloud Computing.

[27] Xiaofeng Chen, Jin Li, Jianfeng Ma, Qiang Tang, and Wenjing Lou. New Algorithms for Secure Outsourcing of Modular Exponentiations. *IEEE Transactions on Parallel and Distributed Systems*, 25(9):2386–2396, September 2014.

[28] Jun Ye, Zheng Xu, and Yong Ding. Secure outsourcing of modular exponentiations in cloud and cluster computing. *Cluster Computing*, 19(2):811–820, June 2016.

[29] Anmin Fu, Yiming Zhu, Guomin Yang, Shui Yu, and Yan Yu. Secure outsourcing algorithms of modular exponentiations with optimal checkability based on a single untrusted cloud server. *Cluster Computing*, 21(4):1933–1947, December 2018.

[30] Yanli Ren, Min Dong, Zhenxing Qian, Xinpeng Zhang, and Guorui Feng. Efficient Algorithm for Secure Outsourcing of Modular Exponentiation with Single Server. *IEEE Transactions on Cloud Computing*, 9(1):145–154, January 2021. Conference Name: IEEE Transactions on Cloud Computing.

[31] Feng Zhai, Ting Yang, Bing Zhao, and Hao Chen. Privacy-Preserving Outsourcing Algorithms for Multidimensional Data Encryption in Smart Grids. *Sensors*, 22(12):4365, January 2022. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[32] Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to IOPs and stateless blockchains. *Advances in cryptology – CRYPTO 2019*, 11692:561–586, 2019. Publisher: Springer.

[33] Benjamin Wesolowski. Efficient verifiable delay functions. *Advances in cryptology – EUROCRYPT 2019*, 11478:379–407, 2019. Publisher: Springer.

[34] Faheem Zafar, Abid Khan, Saif Ur Rehman Malik, Mansoor Ahmed, Adeel Anjum, Majid Iqbal Khan, Nadeem Javed, Masoom Alam, and Fuzel Jamil. A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends. *Computers & Security*, 65:29–49, 2017.

[35] Giuseppe Ateniese, Seny Kamara, and Jonathan Katz. Proofs of storage from homomorphic identification protocols. In *Advances in cryptology - ASIACRYPT 2009, 15th international conference on the theory and application of cryptology and information security, tokyo, japan, december 6-10, 2009. Proceedings*, volume 5912 of *Lecture notes in computer science*, pages 319–333. Springer, 2009.

[36] Wenjun Luo and Guojing Bai. Ensuring the data integrity in cloud data storage. In *2011 IEEE International Conference on Cloud Computing and Intelligence Systems*, pages 240–243, September 2011. ISSN: 2376-595X.

[37] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[38] Qingji Zheng and Shouhuai Xu. Fair and dynamic proofs of retrievability. In *Proceedings of the first ACM conference on Data and application security and privacy*, pages 237–248, 2011.