The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume L-4/W3-2026
10th International Conference on Smart Data and Smart Cities (SDSC), 29 September–1 October 2026, Sofia, Bulgaria

Privacy-Preserving Aggregation of Smart City Sensor Data using Paillier Encryption on Embedded Devices

Jan Seedorf, Bianca Maier, Nele Waldbaur, Tim Schmid, Brusk Mencuetek, Timo Sigle, Manuel Sigle, Annegret Weng

HFT Stuttgart, Schellingstr. 24, 70174 Stuttgart, Germany
jan.seedorf@hft-stuttgart.de, bianca.maier@cancom.de, 31wane1bif@hft-stuttgart.de, 31scti1bif@hft-stuttgart.de
31mebr1bif@hft-stuttgart.de, 31siti1bif@hft-stuttgart.de, 31sima1bif@hft-stuttgart.de, annegret.weng@hft-stuttgart.de

Keywords: Smart City, IoT, Homomorphic Encryption, Paillier, Raspberry Pi, Privacy-Preserving Data Processing

Abstract

Smart-city applications increasingly rely on distributed sensor infrastructures that continuously collect and process potentially sensitive data. Homomorphic encryption provides a promising approach for protecting such data by enabling computations directly on encrypted values, but its computational overhead remains a concern for resource-constrained edge devices. This paper investigates the practical feasibility of additive homomorphic encryption for smart-city sensor data processing using a baseline implementation of the Paillier cryptosystem without hardware-specific optimizations. A distributed prototype was implemented on Raspberry Pi devices, in which sensor readings are encrypted locally, combined in the encrypted domain, and decrypted only at the sink node. The prototype was first evaluated using a complete end-to-end aggregation workflow on Raspberry Pi 3 and Raspberry Pi Zero 2 W devices. Subsequently, the cryptographic operations were evaluated in detail for key sizes of 1024, 2048, and 4096 bits on Raspberry Pi 4 devices using real-world smart-city sensor datasets. The results demonstrate a strong dependency of execution time on the selected key size, while differences between the evaluated sensor datasets have only a minor influence on cryptographic performance. A comparison with RSA quantifies the additional computational cost introduced by Paillier encryption and its additive homomorphic functionality. Finally, practical sensor processing rates are derived from the measured execution times to relate the cryptographic overhead to realistic sensing frequencies. The results show that, despite its considerable computational cost compared with conventional public-key encryption, Paillier-based aggregation can provide a practical privacy-preserving processing approach for smart-city sensing scenarios with moderate sampling frequencies.

1. Introduction

The increasing deployment of Internet of Things (IoT) technologies in smart city environments has led to a continuous growth in the collection and processing of sensor data. Applications such as environmental monitoring, energy management, and infrastructure optimization rely on large numbers of distributed sensors that continuously generate measurements. While these systems enable more efficient urban services, they also raise significant concerns regarding data privacy, particularly when data must be processed across multiple devices and administrative domains (Sookhak et al., 2019).

Traditional cryptographic approaches protect data during transmission and storage but require decryption before computations can be performed. As a result, intermediate processing nodes must gain access to plaintext information, creating potential privacy risks. Homomorphic encryption addresses this limitation by allowing computations to be performed directly on encrypted data. In particular, the Paillier cryptosystem (Paillier, 1999) provides an additive homomorphic property that enables the aggregation of encrypted values without revealing individual measurements. Among partially homomorphic schemes, the Paillier cryptosystem remains one of the most widely adopted approaches for privacy-preserving aggregation due to its additive homomorphic property and comparatively moderate computational complexity.

The use of homomorphic encryption for privacy-preserving aggregation has been studied extensively in domains such as smart grids, IoT systems, and distributed sensing infrastructures (Li et al., 2010, Erkin et al., 2013, Loukil et al., 2021). Existing work has primarily focused on improving the computational efficiency of homomorphic encryption schemes through algorithmic optimizations, specialized cryptographic libraries, or dedicated hardware support. While these approaches demonstrate the potential of homomorphic encryption, they often provide limited insight into the practical feasibility of deploying such techniques in realistic edge-computing environments without extensive optimization efforts.

In this paper, we investigate a complementary question: whether a straightforward implementation of the Paillier cryptosystem can already provide sufficient performance for practical smart city sensing applications on commodity edge hardware. Rather than proposing a new cryptographic scheme or an optimized variant of Paillier, we focus on the end-to-end deployment of homomorphic aggregation within a realistic smart city data processing pipeline.

To this end, we implemented the Paillier cryptosystem in C and extended it to support the processing of individual sensor values, homomorphic aggregation, and the handling of floating-point measurements through integer scaling. Building on this implementation, we developed a distributed prototype system comprising Raspberry Pi devices connected through peer-to-peer communication links. The prototype supports the complete processing chain from sensor data encryption and transmission to homomorphic aggregation and final decryption.

The system was evaluated using real-world smart city sensor data, including temperature measurements, energy consumption values, cooling power percentages, and flow-rate measurements. In addition to validating the correctness of homo-

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-L-4-W3-2026-159-2026

morphic aggregation, we performed a comprehensive performance evaluation across multiple key sizes and compared the results against RSA as a performance reference, representing a widely used public-key cryptosystem. The comparison with RSA is intended to quantify the computational overhead introduced by additive homomorphic encryption rather than to compare equivalent functionality.

The contributions of this paper are as follows:

- We present and evaluate a practical deployable baseline implementation of the Paillier cryptosystem for privacy-preserving sensor data aggregation without algorithmic or hardware-specific optimizations.
- We integrate the implementation into a complete end-to-end smart city data processing pipeline including encryption, transmission, homomorphic aggregation, and decryption.
- We validate the proposed system using heterogeneous real-world smart-city sensor data, including temperature measurements, energy consumption values, cooling power percentages, and flow rates. In contrast to evaluations based solely on synthetic benchmark values, this enables a realistic assessment of homomorphic encryption for smart city deployments under practical operating conditions.
- We perform a comprehensive performance evaluation across multiple key sizes and compare the results against RSA as a performance reference. Furthermore, we derive practical sensor processing rates, providing quantitative guidance on the applicability of additive homomorphic encryption in resource-constrained smart-city environments.

Our results show that, although homomorphic encryption incurs a substantially higher computational overhead than RSA, practical deployment on resource-constrained devices is feasible for a wide range of smart city applications whose sensor sampling frequencies remain within the processing capabilities of the platform.

2. Background: Homomorphic Encryption and the Paillier Cryptosystem

2.1 Homomorphic Encryption

Homomorphic encryption enables computations to be performed directly on encrypted data without requiring access to the underlying plaintext. Homomorphic encryption schemes can be broadly classified as partially or fully homomorphic. Fully homomorphic encryption (FHE) supports arbitrary computations on encrypted data but typically incurs substantial computational overhead. For resource-constrained IoT and edge-computing environments, partially homomorphic schemes are often preferred because they provide specific computation capabilities at significantly lower computational cost.

Homomorphic encryption is particularly attractive for distributed sensing and smart city applications, where data may be processed across multiple nodes that should not have access to individual measurements. By allowing computations on ciphertexts, sensitive information remains protected throughout the processing pipeline, reducing the need to expose plaintext data at intermediate stages.

2.2 The Paillier Cryptosystem

A prominent example of a partially homomorphic encryption scheme is the Paillier cryptosystem (Paillier, 1999). We briefly recall its construction. Our description differs slightly from the general construction given in (Paillier, 1999), as we adopt a commonly used simplification that facilitates implementation.

For key generation, two distinct large primes p and q are chosen such that

$$\gcd\big(pq, (p-1)(q-1)\big) = 1. \tag{1}$$

One then sets $n = pq$ and $g = n + 1$. The public key is (n, g), or simply n, since g is uniquely determined by n. The private key may be represented either by the pair (p, q) or, equivalently, by Euler's totient $\varphi(n) = (p-1)(q-1)$.

To encrypt a message m with $0 \leq m < n$, the sender chooses a uniformly random value r satisfying $0 < r < n$ and $\gcd(r, n) = 1$. The ciphertext is then computed as

$$E(m, r) = c = g^m \cdot r^n \bmod n^2. \tag{2}$$

A receiver who knows the private key can recover the plaintext by computing

$$D(c) = L\left(c^{\varphi(n)} \bmod n^2\right) \cdot \varphi(n)^{-1} \bmod n, \tag{3}$$

where

$$L(x) = \frac{x-1}{n} \tag{4}$$

for $x \equiv 1 \pmod n$, and $\varphi(n)^{-1}$ denotes the multiplicative inverse of $\varphi(n)$ modulo n, that is,

$$\varphi(n)\varphi(n)^{-1} \equiv 1 \pmod n. \tag{5}$$

Using the binomial theorem, it is easy to verify that

$$D(E(m, r)) = m. \tag{6}$$

The semantic security of the cryptosystem relies on the Decisional Composite Residuosity Assumption (DCRA) (Paillier, 1999). Factoring n would break the system, since knowledge of the factorization $n = pq$ immediately reveals $\varphi(n)$.
Paillier's cryptosystem has an additive homomorphic property:

$$\begin{aligned} &E(m_1, r_1)E(m_2, r_2) \\ &\equiv E\big((m_1 + m_2) \bmod n, (r_1 r_2) \bmod n\big) \bmod n^2 \end{aligned} \tag{7}$$

Consequently,

$$D(E(m_1, r_1)E(m_2, r_2) \bmod n^2) = (m_1 + m_2) \bmod n. \tag{8}$$

This property enables encrypted values to be aggregated without revealing the underlying plaintexts. It makes the Paillier cryptosystem attractive for a wide range of applications, including electronic voting, smart-grid/smart-meter data aggregation, and electronic auctions. Only the final recipient needs access to the private decryption key, whereas the required additive operations can be performed directly on the encrypted data.

The Paillier cryptosystem is generally slower than RSA because its arithmetic operations are performed modulo n^2 rather than

modulo n. In contrast to Paillier, however, textbook RSA does not provide additive homomorphism. The additional computational cost of Paillier can therefore be regarded as the price paid for its useful additive homomorphic property.

In this work, the additive homomorphic property is exploited to enable privacy-preserving aggregation of smart city sensor data. Since the Paillier cryptosystem operates on integer values, floating-point sensor measurements are transformed using a scaling approach prior to encryption and converted back after decryption.

3. System Architecture and Implementation

The objective of this work was to investigate the practical feasibility of privacy-preserving sensor data aggregation in a smart city environment using additive homomorphic encryption. To this end, we implemented a complete processing pipeline that supports the encryption, transmission, aggregation, and decryption of sensor measurements.

Figure 1 illustrates the workflow of the proposed privacy-preserving sensor data aggregation scheme. Each sensor node encrypts its locally acquired sensor reading before transmission. Sensor node A forwards the resulting ciphertext to sensor node B, where it is homomorphically combined with the ciphertext of the local sensor reading using the additive homomorphic property of the Paillier cryptosystem. The resulting aggregated ciphertext is subsequently transmitted to the sink node, where only the final aggregated sensor value is decrypted. Consequently, no intermediate node has access to plaintext measurements generated by other sensor nodes.

3.1 Paillier Implementation

As a foundation for the system, we first implemented the Paillier cryptosystem in C and adapted it for the processing of individual sensor measurements. While the mathematical principles of the cryptosystem remained unchanged, the software architecture was then significantly restructured to support automated benchmarking and distributed operation.

The implementation was organized into separate modules for key generation, encryption, decryption, and homomorphic aggregation. In addition to the original functionality, we introduced dedicated components for processing individual sensor values and for performing homomorphic aggregation of arbitrary ciphertexts. This modular design enabled the independent evaluation of cryptographic operations and simplified their integration into the distributed testbed.

To support different security levels, the implementation allows configurable key lengths of 1024, 2048, and 4096 bits. Furthermore, the software exposes command-line interfaces that facilitate automated execution and benchmarking.

3.2 Support for Real-World Sensor Data

A practical challenge when applying the Paillier cryptosystem to sensor measurements is that the scheme operates on integer values, whereas many real-world sensors generate floating-point data. Examples include temperature measurements, energy consumption values, and flow rates.

To address this issue, sensor values were converted into integers through a configurable scaling mechanism prior to encryption. Depending on the required precision, values were multiplied by a fixed scaling factor and rounded to the nearest integer. After decryption, the inverse transformation was applied to reconstruct the original measurement values. This approach enabled the processing of heterogeneous sensor data without modifications to the underlying cryptographic operations.

3.3 Distributed Testbed

Building on the proposed aggregation workflow shown in Figure 1, a distributed prototype system was implemented using Raspberry Pi devices to realize the generic sensor and sink nodes. The testbed represents a resource-constrained edge-computing environment typical of many IoT and smart city deployments.

The nodes were interconnected through peer-to-peer communication links, enabling encrypted sensor data to be exchanged and aggregated across multiple devices. The resulting setup provides a realistic representation of a distributed sensing infrastructure in which intermediate nodes are able to process encrypted information without gaining access to sensitive measurements.

To support experimental evaluation, the testbed was integrated with an automated measurement framework that records execution times for key generation, encryption, decryption, and homomorphic aggregation. This framework enabled reproducible benchmarking across different hardware platforms, key sizes, and sensor datasets.

4. Experimental Setup

The objective of the experimental evaluation was to assess the practical feasibility of a baseline implementation of the Paillier cryptosystem for smart-city sensor data processing on resource-constrained edge devices. In contrast to previous work focusing on implementation-specific or hardware-specific optimizations, the proposed prototype was implemented without such optimizations in order to quantify the performance that can already be achieved in a realistic end-to-end deployment. The evaluation focused on three aspects: (i) the computational performance of the Paillier cryptosystem, (ii) the correctness and efficiency of homomorphic aggregation, and (iii) the comparison with a conventional public-key cryptosystem.

4.1 Hardware Platform

The experiments were conducted on Raspberry Pi devices representing a typical edge-computing environment. Initial implementation and validation were performed on Raspberry Pi Zero 2 W and Raspberry Pi 3 systems, while the main benchmark campaign was executed on Raspberry Pi 4 devices. The Raspberry Pi platform was selected due to its widespread use in IoT and edge-computing applications and its limited computational resources compared to desktop-class hardware.

4.2 Benchmark Methodology

To investigate the impact of security parameters on performance, experiments were conducted using key lengths of 1024, 2048, and 4096 bits. For each configuration, the execution times of key generation, encryption, and decryption were measured separately. An automated benchmarking framework was developed to ensure reproducible measurements. Execution

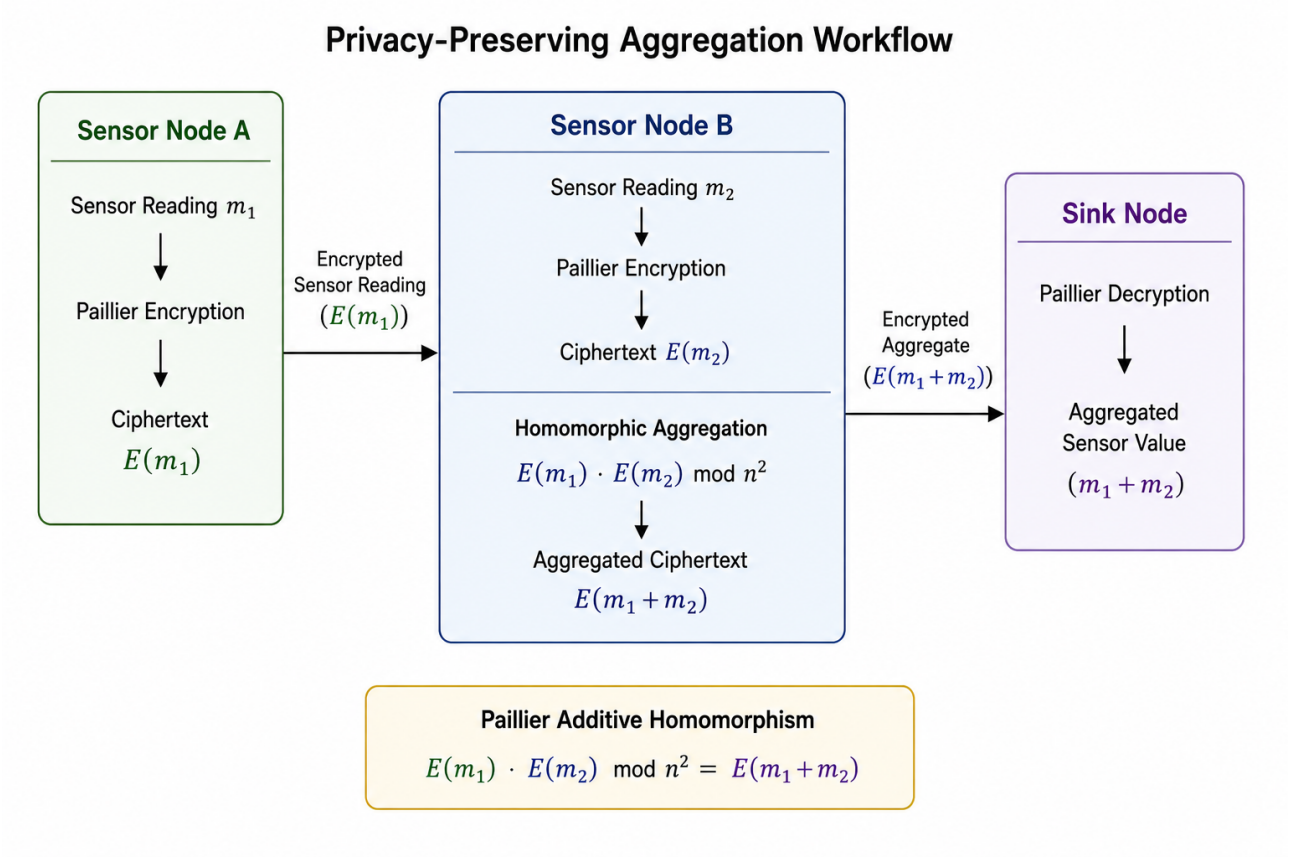

Figure 1. Workflow of the proposed privacy-preserving sensor data aggregation scheme. Sensor node A encrypts its local sensor reading and forwards the resulting ciphertext to sensor node B. Sensor node B encrypts its own sensor reading and homomorphically combines both ciphertexts using the additive homomorphic property of the Paillier cryptosystem. The aggregated ciphertext is subsequently transmitted to the sink node, where only the aggregated sensor value is decrypted.

times were recorded individually and stored for subsequent statistical analysis.

In addition to the Paillier cryptosystem, RSA (Rivest et al., 1978) was evaluated as a performance reference. The comparison is not intended to assess equivalent functionality, as RSA does not support additive homomorphic operations. While textbook RSA possesses a multiplicative homomorphic property, this property is not suitable for the additive aggregation considered in this work. Consequently, RSA serves as a widely deployed public-key cryptosystem that provides a useful performance reference for quantifying the computational overhead introduced by additive homomorphic encryption.

4.3 Sensor Dataset

In addition to synthetic benchmark values, the evaluation was performed using real-world smart-city sensor data, collected within the iCity project, a collaborative research initiative on intelligent and sustainable urban infrastructures (HFT Stuttgart, 2025).

Four representative sensor types were selected from a larger smart city dataset:

- Cooling water supply temperature (°C)
- Total cold-water energy consumption (MW h)
- Minimum cooling capacity (%)
- Flow rate downstream of solar collectors ($\mathrm{m^3\,h^{-1}}$)

For each sensor type, 300 consecutive measurements sampled at one-hour intervals were processed. The selected sensors represent different physical quantities, value ranges, and numerical characteristics, providing a realistic workload for evaluating the behavior of the cryptographic operations.

Since the Paillier cryptosystem operates on integers, floating-point measurements were transformed using the scaling approach described in Section 3.2. This allowed realistic sensor values to be processed without modifying the underlying cryptographic scheme.

4.4 Homomorphic Aggregation Validation

To verify the correctness of the implementation, homomorphic aggregation experiments were conducted using randomly generated input values. The tests repeatedly combined encrypted measurements and compared the decrypted results with the corresponding plaintext sums.

The validation experiments confirmed the correctness of the additive homomorphic property and demonstrated stable execution times across repeated aggregation operations. No deviations between expected and computed results were observed during the experiments.

Platform	Processing Step	Mean (s)	Std. Dev. (s)
Pi Zero 2 W	Encryption	0.449	0.007
Pi Zero 2 W	Aggregation Step	0.615	0.010
Pi Zero 2 W	Decryption	0.443	0.011
Pi Zero 2 W	End-to-End Workflow	1.516	0.019
Pi 3	Encryption	0.442	0.009
Pi 3	Aggregation Step	0.583	0.010
Pi 3	Decryption	0.438	0.008
Pi 3	End-to-End Workflow	1.472	0.016

Table 1. Execution times (in seconds) of the end-to-end Paillier prototype workflow on Raspberry Pi Zero 2 W and Raspberry Pi 3 devices with a key size of 2048 bits.

5. Results and Discussion

5.1 Initial Validation of the Paillier Prototype

Before conducting detailed performance measurements, the complete prototype was evaluated to verify the correctness of the implemented end-to-end processing pipeline. The evaluation considered the complete processing chain from sensor data encryption over homomorphic aggregation to final decryption (as shown in Figure 1).

For each experiment, an initial fixed integer value was encrypted on the Raspberry Pi. Subsequently, a fixed integer value was encrypted in each iteration and homomorphically combined with the encrypted aggregate obtained from the previous iteration. Finally, the aggregated ciphertext was decrypted to recover the resulting plaintext. The measured execution times therefore correspond to the individual processing stages of the prototype workflow. In particular, the reported aggregation time represents the complete aggregation step, including the encryption of the second value and its homomorphic combination with the previously encrypted value (i.e., its execution time exceeds that of the isolated ciphertext multiplication and reflects the complete processing overhead occurring in the practical smart-city sensing pipeline, compare with Figure 1). All measurements were performed consecutively on an otherwise idle system to minimize the influence of background processes.

The initial end-to-end evaluation was performed on both a Raspberry Pi Zero 2 W and a Raspberry Pi 3. As shown in Table 1, both platforms exhibited similar execution times across all workflow stages (the measurements were obtained from 1000 aggregation iterations using integer input values): Using a 2048-bit Paillier key, the complete workflow required 1.472 s on average on the Raspberry Pi 3, compared to 1.516 s on the Raspberry Pi Zero 2 W.

The observed performance difference is relatively small, which is consistent with the similar hardware characteristics of both platforms, as they are based on ARM Cortex-A53 processor cores with comparable clock frequencies. These results indicate that lower-cost edge hardware can be sufficient for smart-city sensing scenarios with moderate sampling frequencies, depending on the performance requirements of the application.

5.2 Evaluation using Real-World Smart-City Sensor Data

In addition to the synthetic benchmark measurements, the proposed prototype was evaluated using four heterogeneous real-world smart-city datasets (see Section 4.3). The datasets comprise temperature measurements, energy consumption values, cooling power percentages, and flow rates, representing typical sensing applications encountered in smart-city environments. This evaluation was performed on Raspberry Pi 4 devices.

Dataset	1024 bit	2048 bit	4096 bit
Temp.	0.057 ± 0.003	0.346 ± 0.116	1.875 ± 0.509
Energy C.	0.062 ± 0.019	0.341 ± 0.105	1.883 ± 0.471
Cooling C.	0.071 ± 0.030	0.340 ± 0.107	1.860 ± 0.437
Flow R.	0.062 ± 0.019	0.332 ± 0.092	1.869 ± 0.470

Table 2. Encryption times for four heterogeneous real-world smart-city sensor datasets on Raspberry Pi 4 devices. Values represent the mean execution time $\pm$ standard deviation over 300 measurements for each key size in seconds.

Dataset	1024 bit	2048 bit	4096 bit
Temp.	0.055 ± 0.003	0.339 ± 0.113	1.816 ± 0.364
Energy C.	0.059 ± 0.018	0.340 ± 0.111	1.864 ± 0.485
Cooling C.	0.067 ± 0.029	0.335 ± 0.101	1.868 ± 0.493
Flow R.	0.059 ± 0.018	0.331 ± 0.095	1.871 ± 0.499

Table 3. Decryption times for four heterogeneous real-world smart-city sensor datasets. Values represent the mean execution time $\pm$ standard deviation over 300 measurements for each key size in seconds.

For each dataset, the complete encryption benchmark was repeated using key sizes of 1024, 2048, and 4096 bits. The reported results are based on 300 sensor readings for each sensor dataset. Tables 2 and 3 summarize the measured encryption and decryption times.

For each evaluated key size, only minor differences were observed between the four sensor datasets. Instead, the execution time is primarily determined by the selected cryptographic key size. This behaviour is expected, since the computational complexity of the Paillier cryptosystem depends on the underlying modular arithmetic rather than on the numerical values of the processed sensor data.

The results therefore demonstrate that our implementation provides consistent computational performance across heterogeneous real-world smart-city sensing applications while preserving the confidentiality of the processed measurements.

5.3 Performance Comparison with RSA

To quantify the computational overhead introduced by additive homomorphic encryption, the performance of the proposed Paillier implementation was compared with RSA, representing a widely deployed public-key cryptosystem. Although RSA exhibits a multiplicative homomorphic property, it does not support additive homomorphic aggregation and therefore cannot provide the functionality required for privacy-preserving sensor data aggregation. Consequently, the comparison is intended solely as a computational performance reference rather than an evaluation of equivalent functionality.

Since the previous section demonstrated that the computational performance is largely independent of the processed sensor data, the benchmark results presented in this section correspond to the average execution times obtained over the four evaluated real-world smart-city datasets.

Key generation, encryption, and decryption were evaluated for key sizes of 1024, 2048, and 4096 bits. Encryption and decryption were measured over 300 iterations (per sensor), whereas

Key Size	RSA (ms)	Paillier (ms)
1024 bit	159 ± 78	276 ± 131
2048 bit	431 ± 245	2431 ± 1207
4096 bit	5625 ± 4334	23215 ± 14356

Table 4. Comparison of RSA and Paillier key-generation times averaged over the four evaluated real-world smart-city sensor datasets. Values represent the mean execution time $\pm$ standard deviation on Raspberry Pi 4 devices.

Key Size	RSA (ms)	Paillier (ms)
1024 bit	0.16 ± 0.08	63 ± 21
2048 bit	0.24 ± 0.07	340 ± 105
4096 bit	0.87 ± 0.41	1872 ± 472

Table 5. Comparison of RSA and Paillier encryption times averaged over the four evaluated real-world smart-city sensor datasets. Values represent the mean execution time $\pm$ standard deviation on Raspberry Pi 4 devices.

key generation was measured over 10 iterations because it is typically performed only during system initialization or occasional key renewal. Tables 4, 5, and 6 summarize the measured execution times on a Raspberry Pi 4 device, while Figure 2 provides a visual comparison of the corresponding benchmark results.

As expected, RSA consistently achieves substantially lower execution times than the Paillier cryptosystem for all evaluated operations and key sizes. The largest performance differences are observed for encryption and decryption, where the additional modular arithmetic required by additive homomorphic encryption results in significantly higher computational costs. Although key generation also exhibits a considerably higher execution time for the Paillier cryptosystem, this operation is typically performed only during system initialization or occasional key renewal and therefore has only a limited impact on the overall runtime of practical sensing applications.

The substantially higher execution times measured for Paillier are consistent with its more expensive modular arithmetic. In particular, Paillier encryption requires modular exponentiation in the ciphertext space modulo n^2, whereas RSA encryption typically uses a small public exponent and benefits from highly optimized implementations. The comparison therefore quantifies the cost of the additional additive homomorphic functionality rather than comparing functionally equivalent cryptosystems.

In general, these observed differences reflect the additional computational effort associated with additive homomorphic encryption, which enables privacy-preserving aggregation of encrypted sensor data. The practical implications of these execution times are analyzed in the following section by deriving achievable sensor processing rates.

5.4 Practical Sensor Processing Rates

While the execution times presented in the previous sections quantify the computational performance of the proposed prototype, they do not directly indicate whether the system is suitable for practical sensing applications. To facilitate this assessment, the measured encryption times were translated into approximate sensor processing rates. Since every sensor reading must be encrypted before transmission, the encryption throughput determines the maximum sustainable sampling frequency of a single sensing node.

Key Size	RSA (ms)	Paillier (ms)
1024 bit	1.3 ± 0.6	60 ± 20
2048 bit	6.1 ± 0.3	337 ± 105
4096 bit	42.7 ± 7.5	1855 ± 464

Table 6. Comparison of RSA and Paillier decryption times averaged over the four evaluated real-world smart-city sensor datasets. Values represent the mean execution time $\pm$ standard deviation on Raspberry Pi 4 devices.

Key Size (bits)	Mean (s)	Std. Dev. (s)	R_{avg} (Hz)	R_{cons} (Hz)	$T_{\min}$
1024	0.0629	0.0208	15.89	11.94	63 ms
2048	0.3399	0.1051	2.94	2.25	340 ms
4096	1.8718	0.4719	0.53	0.43	1.87 s

Table 7. Estimated sensor processing rates derived from the measured Paillier encryption times.

The average sensor processing rate was calculated as

$$R_{\text{avg}} = \frac{1}{\mu}, \tag{9}$$

where μ denotes the measured mean encryption time.

To additionally account for the observed execution-time variability, a conservative processing rate was estimated as

$$R_{\text{cons}} = \frac{1}{\mu + \sigma}, \tag{10}$$

where σ denotes the measured standard deviation. By incorporating one standard deviation into the execution time, this estimate provides a lower-bound approximation of the sustainable sampling rate under the observed timing variations.

The minimum achievable sampling interval is given by

$$T_{\min} = \mu, \tag{11}$$

where μ denotes the mean encryption time. Since each sensor reading must be encrypted before it can be transmitted and subsequently aggregated, the encryption time determines the shortest interval at which consecutive sensor readings can be processed by a single sensor node.

Table 7 summarizes the resulting processing rates. For the 1024-bit configuration, the proposed implementation supports approximately 17 encrypted sensor readings per second on average, while the conservative estimate still exceeds 15 readings per second. For 2048-bit keys, approximately three sensor readings per second can be processed, whereas the 4096-bit configuration still supports approximately one reading every two seconds.

Typical smart-city monitoring applications, including environmental sensing, district cooling supervision, energy management, and infrastructure monitoring, commonly operate with sampling intervals ranging from several seconds to several minutes. Consequently, even the evaluated 4096-bit configuration seems suitable for many practical smart-city sensing applications despite its higher computational overhead.

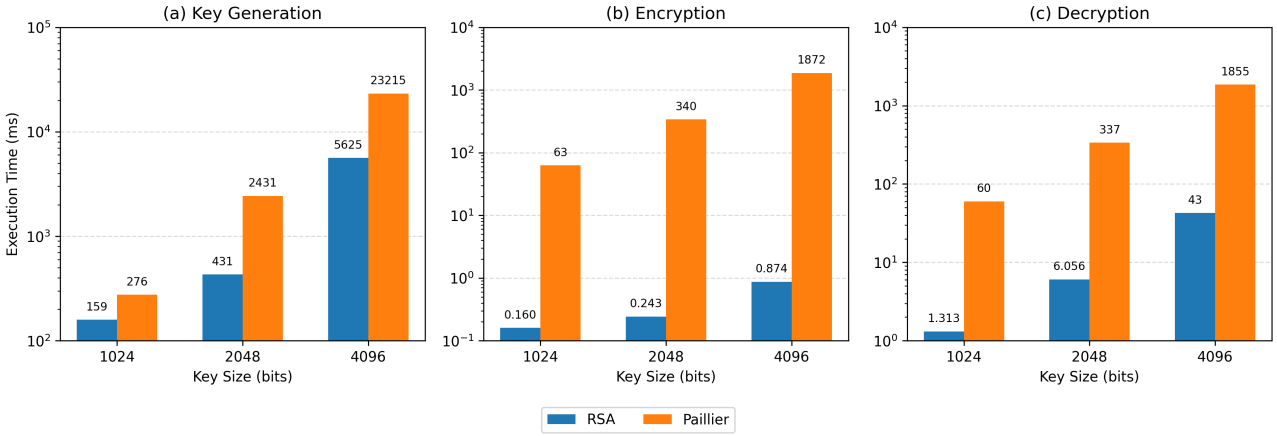

Figure 2. Performance comparison between RSA and the Paillier cryptosystem on a Raspberry Pi 4 for (a) key generation, (b) encryption, and (c) decryption. Execution times are shown on a logarithmic scale.

5.5 Discussion

The presented results demonstrate that additive homomorphic encryption can be integrated into realistic smart-city sensing pipelines without requiring highly optimized implementations or specialized hardware. While previous work has primarily focused on optimizing cryptographic implementations, this study shows that even a straightforward implementation already provides sufficient performance for many practical sensing applications.

Furthermore, the evaluation using heterogeneous real-world smart-city sensor data demonstrates that the proposed approach performs reliably under realistic operating conditions rather than only under synthetic benchmark scenarios. Together with the derived sensor processing rates, these findings provide quantitative evidence that privacy-preserving aggregation based on the Paillier cryptosystem is practically deployable on commodity edge hardware for a wide range of smart-city monitoring applications.

5.6 Limitations

The presented evaluation focuses on a baseline implementation of the Paillier cryptosystem and does not incorporate algorithmic optimizations, specialized cryptographic libraries, or hardware accelerators. Consequently, the reported execution times should be interpreted as a conservative estimate of achievable performance.

Future work could investigate optimized implementations and compare their performance against the baseline established in this study. Such an analysis would help quantify the benefits of optimization techniques and further clarify the trade-offs between implementation complexity and runtime performance.

6. Related Work

Privacy-preserving data aggregation has been studied extensively in smart-grid, wireless sensor network, and Internet of Things (IoT) environments. Many approaches employ the additive homomorphic property of the Paillier cryptosystem (Paillier, 1999) to aggregate consumption measurements, environmental data, or monitoring information while preventing the disclosure of individual readings. Typical application domains include smart metering, energy management, and distributed sensing infrastructures, where aggregated values are often sufficient for decision-making (Li et al., 2010, Erkin et al., 2013, Loukil et al., 2021).

Beyond application-specific aggregation schemes, several studies have investigated the deployment of homomorphic encryption on embedded and edge-computing platforms. Existing work has demonstrated the feasibility of Paillier-based processing on portable and resource-constrained devices and has explored implementation-specific optimizations to improve performance on such platforms (Karageorgopoulou et al., 2023). These efforts primarily focus on adapting and optimizing homomorphic encryption for constrained hardware environments.

In parallel, research has explored more powerful homomorphic encryption schemes. Fully homomorphic encryption (FHE), introduced by Gentry (Gentry, 2009), enables arbitrary computations on encrypted data and has stimulated extensive research in privacy-preserving computing. However, the computational requirements of FHE remain substantially higher than those of partially homomorphic schemes, despite significant advances in optimization and acceleration techniques (Gong et al., 2024). As a result, FHE is still challenging to deploy in many IoT and edge-computing scenarios.

While previous work has primarily focused on optimizing homomorphic encryption implementations for constrained hardware platforms, our goal is to assess the performance that can already be achieved without extensive optimization efforts in a realistic end-to-end smart-city deployment.

We investigate the practical feasibility of a baseline implementation integrated into a complete smart-city sensing pipeline. Our evaluation considers the entire workflow comprising sensor data processing, encrypted transmission, homomorphic aggregation, and final decryption. In contrast to evaluations based solely on synthetic benchmark values, the proposed system is validated using real-world smart-city sensor data, including temperature measurements, energy consumption values, cooling power percentages, and flow rates, processed on a distributed Raspberry-Pi-based testbed representing a realistic resource-constrained edge environment.

This perspective complements optimization-focused studies in two ways. First, it quantifies the performance that can already be achieved without extensive implementation-specific optimizations. Second, it demonstrates the practical applicability of homomorphic aggregation for processing heterogeneous real-world smart-city sensor data. The resulting measurements provide insights into the practical deployability of homomorphic aggregation and the sensor processing rates that can be supported on commodity edge hardware.

7. Conclusion

This paper presents a practical evaluation of homomorphic encryption for privacy-preserving smart city data aggregation. We implemented the Paillier cryptosystem in C and integrated it into a distributed sensor-data processing pipeline supporting encryption, transmission, homomorphic aggregation, and decryption. The resulting system was deployed on a Raspberry-Pi-based edge-computing testbed and evaluated using real-world smart city sensor data.

The experimental results confirm the correctness of homomorphic aggregation and demonstrate that privacy-preserving data processing can be realized on resource-constrained hardware. Although the computational overhead is significantly higher than that of RSA, the measured execution times indicate that practical deployment is feasible for many smart city applications whose sensor sampling rates remain within the processing capabilities of the platform. For the evaluated configurations, processing rates ranged from more than 15 sensor readings per second for 1024-bit keys to approximately 0.5 readings per second for 4096-bit keys.

Unlike many existing studies that focus on optimized implementations or specialized hardware support, this work investigated the performance achievable with a baseline implementation integrated into a complete smart-city sensing workflow. The results therefore provide a realistic assessment of the computational costs associated with deploying homomorphic aggregation in practical edge-computing environments.

Future work will investigate optimized implementations, alternative homomorphic encryption schemes, and larger-scale deployments involving additional sensor nodes and aggregation layers. Furthermore, a direct comparison with existing Paillier libraries and privacy-preserving aggregation frameworks would provide further insight into the trade-offs between implementation complexity, performance, and deployability.

Declaration on the Use of Generative AI

ChatGPT (OpenAI) was used to assist with drafting, language editing, and preparing schematic illustrations and data visualizations for this manuscript. All technical content, implementation, experimental design, results, figures, and conclusions were developed, reviewed, and verified by the authors, who take full responsibility for the final manuscript.

References

Erkin, Z., Troncoso-Pastoriza, J. R., Lagendijk, R. L., Pérez-González, F., 2013. Privacy-Preserving Data Aggregation in Smart Metering Systems: An Overview. *IEEE Signal Processing Magazine*, 30(2), 75–86.

Gentry, C., 2009. Fully homomorphic encryption using ideal lattices. *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, Association for Computing Machinery, Bethesda, MD, USA, 169–178.

Gong, Y., Chang, X., Misic, J. V., Misic, V. B., Wang, J., Zhu, H., 2024. Practical Solutions in Fully Homomorphic Encryption: A Survey Analyzing Existing Acceleration Methods. *Cybersecurity*, 7(5).

HFT Stuttgart, 2025. iCity – Intelligent and Sustainable Urban Development. `https://www.hft-stuttgart.com/research/projects/current/icity-1-impulse-project` (Research project, accessed July 2026).

Karageorgopoulou, A., Tsoukas, V., Spathoulas, G., Kakarountas, A., Koziri, M., 2023. Porting the paillier algorithm for homomorphic encryption on portable devices. *2023 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 1–5.

Li, F., Luo, B., Liu, P., 2010. Secure information aggregation for smart grids using homomorphic encryption. *Proceedings of the First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Gaithersburg, MD, USA, 327–332.

Loukil, F., Ghedira-Guegan, C., Boukadi, K., Benharkat, A.-N., 2021. Privacy-Preserving IoT Data Aggregation Based on Blockchain and Homomorphic Encryption. *Sensors*, 21(7), 2452.

Paillier, P., 1999. Public-key cryptosystems based on composite degree residuosity classes. *Advances in Cryptology – EUROCRYPT '99*, Lecture Notes in Computer Science, 1592, Springer, 223–238.

Rivest, R. L., Shamir, A., Adleman, L. M., 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2), 120–126.

Sookhak, M., Tang, H., He, Y., Yu, F. R., 2019. Security and Privacy of Smart Cities: A Survey, Research Issues and Challenges. *IEEE Communications Surveys & Tutorials*, 21(2), 1718–1743.