electronics

Article

PDPHE: Personal Data Protection for Trans-Border Transmission Based on Homomorphic Encryption

Yan Liu [1], Changshui Yang [1], Qiang Liu [1], Mudi Xu [1], Chi Zhang [1], Lihong Cheng [2,*] and Wenyong Wang [1,*]

[1] Trusted Trans-Border Dataspace Group, Department of Computer Science and Engineering, Macau University of Science and Technology, Avenida Wai Long, Taipa, Macau 999078, China; 2109853jin30001@student.must.edu.mo (Y.L.); 2109853pin30002@student.must.edu.mo (C.Y.); 2109853uin30002@student.must.edu.mo (Q.L.); 2109853gin30002@student.must.edu.mo (M.X.); 2109853pin30001@student.must.edu.mo (C.Z.)

[2] Department of Foreign Language, Dalian Jiaotong University, 794 Huanghe Road, Shahekou District, Dalian 116028, China

* Correspondence: nancycheng001@163.com (L.C.); wywang@must.edu.mo (W.W.); Tel.: +86-1594-093-3916 (L.C.); +86-1390-801-4292 (W.W.)

Abstract: In the digital age, data transmission has become a key component of globalization and international cooperation. However, it faces several challenges in protecting the privacy and security of data, such as the risk of information disclosure on third-party platforms. Moreover, there are few solutions for personal data protection in cross-border transmission scenarios due to the difficulty of handling sensitive information between different countries and regions. In this paper, we propose an approach, personal data protection based on homomorphic encryption (PDPHE), to creatively apply the privacy computing technology homomorphic encryption (HE) to cross-border personal data protection. Specifically, PDPHE reconstructs the classical full homomorphic encryption (FHE) algorithm, DGHV, by adding support for multi-bit encryption and security level classification to ensure consistency with current data protection regulations. Then, PDPHE applies the reconstructed algorithm to the novel cross-border data protection scenario. To evaluate PDPHE in actual cross-border data transfer scenarios, we construct a prototype model based on PDPHE and manually construct a data corpus called PDPBench. Our evaluation results on PDPBench demonstrate that PDPHE cannot only effectively solve privacy protection issues in cross-border data transmission but also promote international data exchange and cooperation, bringing significant improvements for personal data protection during cross-border data sharing.

Keywords: cross-border; homomorphic encryption; privacy protection; data transmission; blockchain

check for updates

Citation: Liu, Y.; Yang, C.; Liu, Q.; Xu, M.; Zhang, C.; Cheng, L.; Wang, W. PDPHE: Personal Data Protection for Trans-Border Transmission Based on Homomorphic Encryption. *Electronics* **2024**, *13*, 1959. https://doi.org/10.3390/electronics13101959

Academic Editors: Simeone Marino and Ping-Feng Pai

Received: 8 April 2024
Revised: 23 April 2024
Accepted: 8 May 2024
Published: 16 May 2024

1. Introduction

In the context of economic integration, cross-border data flow has become a key driver of international business, technological innovation, and global cooperation. With the development of digitization, data have emerged as new strategic resources in the economy, inevitably impacting the global economy. However, along with these opportunities come significant challenges in data privacy and security, especially in the transnational economic environment, where privacy and security protection during data transmission become more complex and sensitive.

As the volume of personal data increases and data types become more diverse, safeguarding personal privacy while promoting data flow has become a global issue. The European Union's General Data Protection Regulation (GDPR) [1] is a response to this issue, imposing strict requirements on privacy data and emphasizing the protection of personal data and the compliance of cross-border data transmission. The implementation of GDPR has not only changed the data governance landscape in Europe but also had significant implications for businesses handling the data of EU citizens globally, becoming a consideration for international data exchange and cooperation.

With the continuous improvement of regulations on personal information protection such as GDPR, entities responsible for data governance face several challenges. Firstly, according to these regulations, data subjects must take measures to ensure the auditability of personal data, including patient information entries. For example, GDPR introduces the right to consent, requiring specific data processors or controllers to obtain consent before processing data. Secondly, in the fields of health and finance, the innovation and use of big data technology lead to an inevitable increase in the volume of data collected by institutions. This not only increases the demand for data but also requires data sharing while protecting the privacy of data owners, especially among different institutions, posing new challenges to data security, organization, and technology. Thirdly, under the current global legal framework, determining data ownership depends on the environment of data sharing, adding complexity to data governance.

Furthermore, utilizing data for research purposes is a double-edged sword. Many data owners support using their data for research or improving service quality. However, they are concerned about the potential infringement of their information privacy or improper use or processing of data.

With technological advancements, emerging privacy computing technologies such as homomorphic encryption (HE) [2] and secure multi-party computation (MPC) [3] offer new possibilities for addressing privacy and security issues in cross-border data transmission [4–6]. These technologies enable data processing and analysis without exposing the original data content, thereby ensuring data privacy while extracting data value. This is not only a technological innovation but also an important supplement to existing data governance models. However, despite the immense potential of these technologies, they still face many challenges in practical applications. For example, how to effectively utilize these technologies under different national and regional legal frameworks, how to address the cost and efficiency of technical implementation, and how to ensure the universality and scalability of technical solutions are urgent issues that need to be addressed. HE is a form of encryption that allows computations to be performed on ciphertexts, generating an encrypted result that, when decrypted, matches the result of operations performed on the plaintext.

HE is fundamental to PDPHE as it allows the secure processing of personal data across different jurisdictions without exposing the actual data, maintaining privacy and security. Cross-border data protection challenges. International regulations variability: Different countries have varying regulations regarding data privacy (e.g., GDPR in Europe, CCPA in California), which complicates the management and transfer of personal data across borders. By using HE, PDPHE aims to mitigate the complexities associated with these regulations, ensuring that data can be processed in compliance with local laws without having to be decrypted or exposed. Traditional FHE schemes are often not practical for large-scale or real-time applications due to their high computational demands. This paper modifies the FHE algorithm (specifically the DGHV scheme) to support multi-bit encryption and security level classification, aiming to optimize performance while adhering to regulatory standards.

1.1. Hypotheses

Feasibility of enhanced FHE for practical use. Hypothesis: By reconstructing the classical FHE approach to include multi-bit encryption capabilities and classification of security levels, the PDPHE can be made computationally feasible for real-world applications, particularly in cross-border scenarios where data security and compliance are critical. Effectiveness in protecting data privacy. Hypothesis: The application of the modified FHE algorithm within PDPHE will effectively protect personal data during cross-border transmission against unauthorized access, thus maintaining confidentiality and integrity of the data. Impact on international data exchange. Hypothesis: PDPHE will facilitate safer and more efficient international data exchanges by providing a secure method to process and analyze encrypted data across borders, thus promoting global cooperation and data-driven innovation without compromising data privacy. Adaptability to regulatory changes. Hypothesis: The flexibility of the PDPHE framework in accommodating different encryption levels based on the data protection laws of the involved countries will make it adaptable to ongoing changes and updates in international data protection regulations.

1.2. Contribution

Therefore, the purpose of this paper is to explore in depth how privacy computing technologies can be used within cross-border data sharing to solve privacy and security issues in data transmission. We will analyze existing legal frameworks in detail, explore the principles of privacy computing technologies and their application in cross-border data transmission, and evaluate the practical effects of these technologies through case studies. Through this research, this paper aims to provide new perspectives and solutions for privacy protection and data security in cross-border data transmission. The contributions of this paper are as follows:

- We propose a novel approach, PDPHE (Personal Data Protection based on Homomorphic Encryption), to creatively apply the privacy computing technology Homomorphic Encryption (HE) to cross-border personal data protection.
- We reconstruct the classical full homomorphic encryption (FHE) algorithm DGHV by adding support to multi-bit encryption and security level classification, to ensure consistency with current data protection regulations (e.g., European Union's General Data Protection Regulation, GDPR).
- We construct a prototype model based on PDPHE, and manually construct a data corpus PDPBench, consisting of three categories, 10 specific types, and 440 cases of cross-border personal data information. Our evaluation results on PDPBench demonstrate that PDPHE cannot only effectively solve the privacy protection issues in cross-border data transmission but also promote international data exchange and cooperation, and bring great improvements for personal data protection during cross-border data sharing.

The article is structured as follows: We provide some related work about cross-border and privacy computing in Section 2, and the challenges of cross-border data sharing are presented in Section 3. Then propose a new model PDPHE based on HE in Section 4. A description of our experiment appears in Section 5. Finally, we conclude in Section 6 with a summary of existing work and a plan for future work.

2. Motivation

We ensure the security and privacy of data as it is shared globally, amidst the explosive growth of data volumes and the increasing importance of global data governance. This task is exceptionally complex due to factors such as privacy protection, national security, and trade secrets. In this context, ensuring that data are secure and private while still fully leveraging its value becomes a global challenge.

The emerging global trend suggests that data should not leave its local environment, and while users can utilize the data, they should not have direct access to it. This should become the fundamental application model for the value sharing of cross-border data. However, existing security solutions are not yet capable of supporting this goal.

As early as 2019, MIT's report for the U.S. Department of Health proposed a healthcare big data security application scheme based on blockchain and homomorphic encryption. This approach, which has been recognized by U.S. officials [7], suggests a "usable but invisible" model for protecting healthcare big data and outlines a preliminary technical route.

The technology route of making data usable but invisible primarily revolves around homomorphic encryption. This type of encryption allows for computations to be performed on encrypted data, with the results, once decrypted, remaining accurate. This effectively mitigates the variability and uncertainty of upper-level laws and means that data processors can analyze and process data without actually "seeing" the data itself, thereby protecting its privacy. This technology is particularly crucial for cross-border data sharing because it ensures that even if data are sent to other countries or regions for processing, personal and sensitive information can remain protected.

Therefore, this article applies homomorphic encryption technology to the scenario of cross-border data sharing to address the issue of privacy protection for cross-border data.

3. Background and Relate Work

3.1. Data Protection Regulations

Research in the field of cross-border data transmission encompasses multiple aspects, ranging from legal and regulatory differences to technological challenges, cultural disparities, and geopolitical considerations. These studies are interrelated and collectively contribute to a comprehensive understanding of the complexity of this field. The starting point of research is often the differences in legal and regulatory frameworks, as deeply explored by Kuner [8,9], especially regarding the conflicts in legal systems when dealing with cross-border data flows. This research emphasizes the importance of understanding the legal frameworks governing global data flows. However, differences in legal frameworks are just one part of the many challenges, leading to the crucial issue of data security risks. Greenleaf (2016) [10] shifted the focus to data security, providing an overview of global data privacy laws, particularly in the context of security challenges during cross-border data transfers [11]. This study, starting from a legal perspective, explores the necessity of data protection but offers less discussion on technological implementation, naturally leading to issues of data sovereignty and jurisdiction. Millard [12] filled this gap by focusing on the legal frameworks of cloud computing, especially regarding data sovereignty and jurisdiction [13,14]. This study not only complements the aforementioned legal and security perspectives but also introduces discussions on technological standards and compatibility. On the technological front, Weber (2015) [15] focused on issues of technological standards and interoperability protocols in the Internet of Things. This research highlights the technological challenges in cross-border data transmission, particularly the importance of achieving compatibility between different technological systems and standards. Subsequently, Cate and Mayer-Schönberger (2016) [16] expanded the discussion to the impact of cultural and linguistic differences on data processing. This study, from a sociocultural perspective, explores the challenges in data interpretation and usage, supplementing the technological and legal viewpoints [11]. Finally, Cattaruzza [17] extended the perspective to the geopolitics of data, exploring how international relations and economic policies influence data flows. This research provides a macroscopic view of the challenges in cross-border data transmission, emphasizing the significance of political and economic factors in global data flows [18,19].

While the above research addresses cross-border data-sharing issues through a broad systemic framework, when focusing on specific technologies, there are various forms of solutions available, among which multi-party computation (MPC) [20] has become a research hotspot. MPC is a technology that allows multiple participants to jointly complete specific computational tasks while maintaining the privacy of their respective input data. From the current development of technology, MPC has become an important research direction in the fields of cryptography and data security [21]. It mainly includes three technologies: homomorphic encryption, zero-knowledge-proof, and secret sharing [22].

3.2. Homomorphic Encryption

Homomorphic encryption technology is a method of encryption that allows computations to be directly performed on encrypted data, yielding an encrypted result [23]. This result, when decrypted, is the same as if the same operations had been performed on the original data. This technology has long been a research hotspot in the fields of cryptography and data security [24,25].

The concept of HE was first introduced by Rivest, Adleman, and Dertouzos in 1978. In their paper, they posed the question: Is it possible to design an encryption system in which specific arithmetic operations can be carried out on ciphertexts, and the results remain in an encrypted form [26]? This question marked the beginning of research into HE, but due to technological limitations at the time, this concept was not immediately realized. It was not until 1996 that Craig Gentry, a researcher at IBM, proposed the first partial HE scheme in his doctoral thesis. This scheme supported an unlimited number of additional operations on encrypted data but was limited to that operation alone [2]. Although this was a significant advancement, its limitations meant that it did not fully realize the vision of Rivest and others.

In 2009, Craig Gentry, in his further research at Stanford University, constructed the first fully homomorphic encryption (FHE) scheme [27]. This scheme supported not only addition but also multiplication operations, making it possible to perform arbitrary computations on encrypted data. Gentry's scheme used "ideal lattices" from lattice cryptography and introduced a "bootstrapping" technique to maintain the security of the ciphertext. This groundbreaking work laid the foundation for the development of HE technology.

Despite the revolutionary theoretical significance of Gentry's FHE scheme, its efficiency and computational cost were major obstacles in practical applications. To address this issue, researchers began exploring more efficient algorithms. For example, an IBM research team proposed an improved algorithm based on Gentry's scheme, significantly enhancing computational efficiency [28]. Additionally, HE technology began to be applied in explorations in fields such as cloud computing, data security, and medical privacy protection. For instance, Microsoft Research developed a library called "SEAL", supporting machine learning and statistical analysis on encrypted data [29].

HE is categorized into three main types: Partial homomorphic encryption (PHE), somewhat homomorphic encryption (SHE), and fully homomorphic encryption (FHE) [30].

PHE is the earliest form of HE, supporting one type of arithmetic operation on encrypted data, either addition or multiplication. Notable examples include the RSA encryption algorithm and the ElGamal encryption algorithm. RSA encryption, proposed by Rivest, Shamir, and Adleman in 1978, is a multiplication HE algorithm [31]. ElGamal encryption, introduced by Taher Elgamal in 1985, is an addition HE algorithm based on the discrete logarithm problem [32]. Although these early HE schemes were theoretically significant, their limited functionality restricted their application scope.

SHE represents a significant advancement in HE technology, supporting a limited number of addition and multiplication operations on encrypted data. This category of encryption technology has an advantage over PHE in handling more complex data operations. In 2009, Craig Gentry proposed somewhat of a HE scheme based on ideal lattices [2]. This scheme can be seen as an important step toward FHE.

FHE is the ultimate goal of HE technology, supporting an unlimited number of addition and multiplication operations on encrypted data. The proposal of FHE marked a significant breakthrough in HE technology. In 2009, Gentry presented the first FHE scheme in his doctoral thesis [27]. This scheme used lattice cryptography and introduced the "bootstrapping" technique, and while it had limitations in efficiency, it theoretically proved the possibility of FHE.

Although FHE has been theoretically proven feasible, it faces significant challenges in efficiency and computational cost in practical applications. In recent years, researchers have been striving to improve the efficiency of FHE, for example, through algorithm optimization and hardware acceleration [33].

In terms of algorithms, HE technology, as a method that allows computations on encrypted data, has made significant progress over the past decade. Lattice-based cryptography is an important approach to implementing HE. The security of lattice cryptography is based on mathematical lattice problems, which are considered strong candidates against quantum computer attacks. Regev introduced the learning with errors (LWE) problem in 2005, which has become the core of many HE schemes [34]. Over time, the Bootstrapping technique has emerged as a key technology in HE, allowing encryption schemes to maintain data security after multiple operations, a crucial step in achieving FHE, first introduced by Gentry in his 2009 doctoral thesis [2]. Although HE technology is theoretically revolutionary, its practical application is limited by efficiency and computational costs. In recent years, researchers have been working to improve the efficiency of HE algorithms. Researchers have optimized algorithms to enhance the efficiency of HE. For example, Chen et al. proposed an optimized HE scheme in 2018, which significantly improved computational efficiency while maintaining security [29]. Besides software-level optimization, hardware acceleration is also an important direction for enhancing the efficiency of HE. For example, using specially designed hardware to accelerate HE computations [35]. In 2023, Zainab H. Mahmood and others proposed a new symmetric fully homomorphic encryption scheme that can perform operations on integers without the need for binary conversion. It encrypts messages using prime secret keys. Compared to the DGHV system, this system has made significant improvements. The scheme notably reduces the ciphertext size and execution time compared to the DGHV system. It is approximately 23.8 times faster than DGHV, and generates a single ciphertext for the entire message, unlike traditional schemes which create a ciphertext for each bit of the message [36]. As HE technology develops, its standardization efforts are also gradually advancing. Standardization is crucial for the widespread application and acceptance of HE technology, with influential organizations like HomomorphicEncryption, a group dedicated to the standardization of HE, aiming to promote the development and application of the technology [37,38].

3.3. Zero-Knowledge Proof

ZKP is a cryptographic method that allows one to prove the truth of a statement without revealing any useful information. Since its introduction in the early 1980s, ZKP has become an important research direction in the fields of cryptography and information security.

In 1985, Goldwasser, Micali, and Rackoff introduced the concept of ZKP. Their work provided a rigorous definition of ZKP and demonstrated how to construct such proof systems theoretically [39]. In the 1990s, ZKP began to be applied in various cryptographic protocols, such as identity verification and secure computation [40]. With the improvement in computational capabilities and algorithm optimization, ZKP started to be used in a broader range of fields in the 2000s. For instance, electronic voting systems utilized ZKP technology to ensure the anonymity and untraceability of votes while verifying their validity [41]. In the field of cryptocurrencies, ZKP has been used to create more privacy-preserving transaction systems. For example, Zcash is a cryptocurrency that uses ZKP to protect transaction privacy. Over time, the drawbacks of ZKP became apparent; early systems were often inefficient and not suited for large-scale or real-time applications. The growing demand for applications led to challenges in designing scalable zero-knowledge-proof systems and efficient, secure protocols. It was not until the 2010s that Ben-Sasson, E., et al. designed a lightweight model called zk-SNARKs, a small, non-interactive ZKP technology widely applied in blockchain and cryptocurrency [42]. They later improved upon zk-SNARKs, naming the advancement zk-STARKs, which offered stronger security and better scalability [43]. By 2020, significant progress in the practicality of ZKP was made by Chiesa, A., et al [6]. In recent years, the development of artificial intelligence has further advanced ZKP. In interdisciplinary fields, Weng, J., et al. utilized ZKP in their research on machine learning and data analysis [44].

3.4. Secret Sharing

Secret sharing is a method of distributing secret information among multiple participants, ensuring that the secret can only be recovered when a sufficient number of participants collaborate.

In 1979, Adi Shamir and George Blakley independently but almost simultaneously proposed the concept of secret sharing. Shamir's scheme (Shamir's secret sharing) is based on polynomial interpolation, while Blakley's scheme relies on geometric properties [45,46]. By the 1980s, researchers began exploring different types of secret-sharing schemes, including threshold-based schemes and access structure-based schemes. Threshold-based schemes allow secrets to be shared among a certain number of participants, with the secret only recoverable when enough participants cooperate. Access structure-based schemes are more flexible, allowing for the definition of more complex combinations of participants to recover the secret [47]. In the 1990s, the application of secret sharing expanded, particularly in MPC and distributed systems [48]. With the rise of cloud computing and big data in the 2000s, secret sharing's application in data storage and management gained attention, being used to protect sensitive data stored in the cloud, ensuring data privacy and security. However, large-scale applications brought challenges, such as how to flexibly adjust participants and access rights. To address this issue, in recent years, Boneh, D., and others proposed schemes combining HE with secret sharing, achieving more secure and efficient data processing [49]. Subsequently, Yang combined secret sharing with blockchain, enhancing the privacy and security of transactions [50].

4. Materials and Methods

Overview

In response to the challenges faced by society today, as discussed in the previous chapter, we propose a privacy data protection model based on HE technology, as illustrated in Figure 1. This model includes the following participants:

Data owner: Authorizes the use of their private data.

Provider: Holds the private data of the data owner, including personal information such as patient details.

User: A participant who utilizes the private data of the data owner.

Blockchain-based evidence system (BES): Used for storing the hash values of encrypted data, ensuring the integrity and correctness of the data.

Trusted computing platform (TCP): The most crucial aspect of this platform is its trustworthiness, where encrypted data are processed.

Now, consider a data owner who has lived in Country A for a while and then moves to Country B. They wish to transfer their previous personal information to Country B. This is to ensure that Country B can quickly understand their background, ensuring that they can enjoy the full citizen rights offered by Country B, while also guaranteeing that their privacy is not compromised.

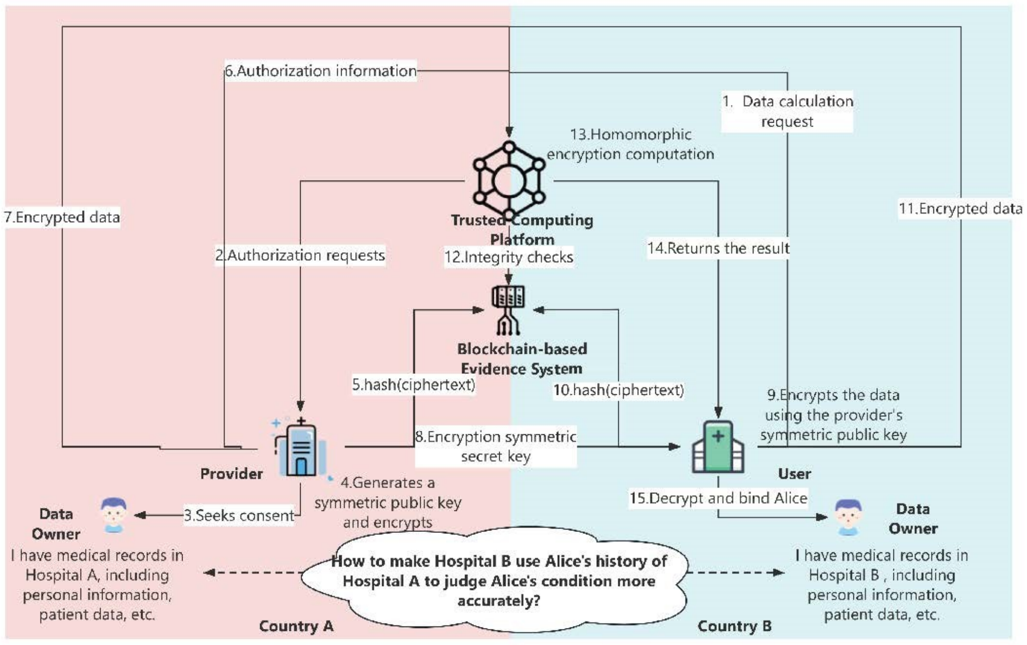
6.Authorization information
1. Data calculation request
13.Homomorphic encryption computation
Trusted Computing Platform
7.Encrypted data
11.Encrypted data
2.Authorization requests
12.Integrity checks
14.Returns the result
Blockchain-based Evidence System
5.hash(ciphertext)
10.hash(ciphertext)
9.Encrypts the data using the provider's symmetric public key
8.Encryption symmetric secret key
Provider
User
4.Generates a symmetric public key and encrypts
15.Decrypt and bind Alice
Data Owner
3.Seeks consent
Data Owner
I have medical records in Hospital A, including personal information, patient data, etc.
How to make Hospital B use Alice's history of Hospital A to judge Alice's condition more accurately?
I have medical records in Hospital B , including personal information, patient data, etc.
Country A
Country B

Figure 1. Proposal model.

Firstly, the data provider sends a data computation request to the TCP. This request includes the computation function $f(m_1, \dots m_i)$, the data provider's identifier $ID_{provider}$, and the public and private keys (PK_{user} and SK_{user}) generated by the data user using the RSA asymmetric encryption algorithm. Upon receiving the request, the TCP initiates an authorization request to the data provider based on their identifier. After verifying the correctness of the authorization request, the provider seeks data authorization from the Data Owner. Subsequently, the provider generates a public key $PK_{provider}$ for HE and encrypts the data. The encrypted data are hashed to generate a hash value, denoted as $hash(ciphertext)$, and this value is stored in the BES for future verification. The data provider sends the encrypted data $\text{Encrypt}\left(PK_{provider}, m_1\right)$ to the TCP and also sends the data provider's public key encrypted with the data user's public key$\text{Encrypt}\left(PK_{provider}, m_1\right)$ to the data user. Upon receiving the encrypted key $\text{Encrypt}\left(PK_{user}, PK_{provider}\right)$, the data user decrypts it using their private key SK_{user} to obtain the data provider's public key $PK_{provider}$. The data user then encrypts data $\text{Encrypt}\left(PK_{provider}, m_2\right)$ with $PK_{provider}$ and sends it to the TCP. The TCP, after receiving the ciphertext, sends an integrity verification request to the blockchain evidence platform to ensure the integrity of the data. It then performs homomorphic encrypted computation. After the computation is completed, the result is sent to the data user, who decrypts it to obtain the actual data computation result and associates it with the data owner, as shown in Algorithm 1.

Algorithm 1 PDPHE model algorithm

```
1:  Initialization: $f(m_1, \dots m_i)$, $PK_{user}$, $SK_{user}$
2:  User $\left\{\begin{array}{c} f(m_1, \dots m) \\ ID_{\text{provider}} \\ PK_{\text{user}} \end{array}\right. \rightarrow TCP$
3:  $TCP\left\{\begin{array}{c} \text{authorization} \\ PK_{\text{user}} \end{array}\right. \rightarrow Provider$
4:  $Provider_{authorization} \rightarrow Owner$
5:  $PK_{provider} = KeyGen(\lambda)$
6:  $C_1 = Encrypt(PK_{provider}, m_1)$
7:  $hash(ciphertext) \rightarrow BES$
8:  $\left(\begin{array}{c} \text{Result}_{\text{auth}} \\ \text{Encrypt}\left(\text{PK}_{\text{provider}}, m_1\right) \end{array}\right) \rightarrow TCP$
9:  $Encrypt(PK_{user}, PK_{provider}) \rightarrow User$
10: $Decrypt(SK_{user}, PK_{provider})$
11: $C_2 = Encrypt(PK_{provider}, m_2)$
12: **if** $BES.hash(C_1) = TCP.hash(C_1)$ && $BES.hash(C_2) = TCP.hash(C_2)$ **then**
13:     $C = f(C_1, C_2)$
14: **else**
15:     $Notify(provider, user)$
16: **end if**
17: $TCP.C \rightarrow User$
18: $User.bind(Owner)$
```

In this paper, the HE algorithm is a key factor affecting the efficiency of data sharing. Considering the computational overhead, we adopt the FHE algorithm, DGHV [51].

In designing the PDPHE fully homomorphic encryption scheme, the decision to base improvements on the DGHV scheme rather than other schemes was guided by the following considerations:

Simplicity and accessibility: The DGHV scheme's relatively simple structure and conceptual clarity provide an easy-to-modify platform. This simplicity serves as a solid starting point for gradual improvements and allows for the transparent integration of new ideas.

Space for improvement: Although other schemes have been extensively refined, DGHV presents untapped opportunities for optimization. Once the efficiency issues within its structural framework are resolved, there can be significant performance gains.

Advantages in cross-border scenarios: PDPHE addresses the unique challenges of cross-border data transmission. The simplicity and clarity of DGHV better suit the characteristics of small volumes of private data typically involved in cross-border transfers, complementing each other effectively.

The HE technology can be represented by the following formula:

$$f(m_1, \ldots m_i) = D(f(E(m_1), E(m_2), E(m_3), \ldots E(m_i))) \tag{1}$$

$E(m)$ and $D(f(E(m_1), E(m_2), E(m_3), \ldots E(m_i)))$ represent the encryption and decryption functions, respectively. An encryption algorithm that supports effective addition and multiplication operations is referred to as supporting fully HE. DGHV is a FHE method that is simpler and easier to understand compared to traditional algorithms. Although its public key size is relatively large, it can be reduced by sacrificing the noise resistance of the original algorithm. The key generation method is defined as follows:

$$\text{KeyGen}(\lambda) \tag{2}$$

Based on the security parameter λ, a large odd number P key is generated, with η being the number of bits in the generated key P.

$Encrypt(PK, m)$ is the encryption process, where pk is the public key, and m is the plaintext. According to the paper [52], the following is stipulated:

$$m \in \{0, 1\} \tag{3}$$

This means that m can represent a bit. r and q are both positive random numbers, with lengths of ρ bits and γ bits, respectively. Therefore, the entire process can be represented as follows:

$$\text{Encrypt}(pk, m) : c = m + 2r + pq \tag{4}$$

$Decrypt(sk, c)$ represents the decryption process, where sk is the private key, and c is the ciphertext. The decryption process can be represented as follows:

$$\text{Decrypt}(sk, c) : m = (c \bmod p) \bmod 2 \tag{5}$$

Since this paper uses symmetric keys, the public key and private key are the same at this time. The homomorphic property of this method is proven as follows, assuming the key is p, and the 1-bit plaintexts are $m1$ and $m2$, then we have the following:

$$\text{Encrypt}(sk, m_1) = c_1 = m_1 + 2r_1 + pq_1 \tag{6}$$

$$\text{Encrypt}(sk, m_2) = c_2 = m_2 + 2r_2 + pq_2 \tag{7}$$

Direct computation of ciphertexts yields the following:

$$c_1 + c_2 = (m_1 + m_2) + 2(r_1 + r_2) + p(q_1 + q_2) \tag{8}$$

$$\begin{aligned} c_1c_2 &= (m_1 + 2r_1 + pq_1)(m_2 + 2r_2 + pq_2) \\ &= m_1m_2 + 2(m_1r_2 + m_2r_1 + 2r_1r_2) + p(m_1q_2 + m_2q_1 + 2r_1q_2 + 2r_2q_1 + pq_1q_2) \end{aligned} \tag{9}$$

Under the condition that $2(m_1r_2 + m_2r_1 + 2r_1r_2) < p$ or $r < \frac{\sqrt{p}}{2}$, we have the following:

$$\text{Decrypt }(sk, \text{c1} + \text{c2}) = ((c_1 + c_2) \bmod p) \bmod 2 = (m_1 + m_2) \bmod 2 \tag{10}$$

$$\text{Decrypt}\ (sk, \text{c1c2}) = ((c_1c_2) \bmod p) \bmod 2 = m_1m_2 \tag{11}$$

The above encryption and decryption processes rely on the generation of random numbers. Traditional random number generation processes are easily attacked. Considering both security and efficiency, this paper chooses a method combining a linear congruential generator with the keccak256 function. This method is suitable for generating integer random numbers, requires little computational power, and ensures unpredictability as long as appropriate parameters and random number seeds are selected. The generation formula is as follows:

$$\begin{cases} X_{n+1} = (aX_n + c) \bmod m, n \geq 0 \\ R_{n+1} = \text{keccak256}(X_{n+1}) \bmod k \end{cases} \tag{12}$$

Here, X_0 is the random number seed.

In reference [51], since the plaintext is $\{0,1\}$, which aligns with the false and true values in Boolean operations, this paper utilizes Boolean HE for computation and stipulates the following:

$$\text{G(m)} = \begin{cases} \text{false, if } m = 0 \\ \text{true, if } m = 1 \end{cases}$$
$$G^{-1}(F) = \begin{cases} 0, \text{if } F = \text{false} \\ 1, \text{if } F = \text{true} \end{cases} \tag{13}$$

The above formula facilitates the conversion from a 1-bit certificate to Boolean values. Common Boolean operations include *AND*, *NAND*, and *OR*, with other operations derivable through combinations of these three. DGHV supports homomorphic addition and multiplication. This paper presents a comparison of homomorphic addition and *XOR* operations, as shown in Tables 1 and 2.

Table 1. Comparison between the homomorphic addition and *XOR* operations.

$m_1(G(m_1))$	$m_2(G(m_2))$	$m_1 + m_2$	$F_1 \oplus F_2$
1 (true)	0 (false)	1	true
1 (true)	1 (true)	0	false
0 (false)	0 (false)	0	false
0 (false)	1 (true)	1	true

Table 2. Comparison of Homomorphic Multiplication and *AND* Operations.

$m_1(G(m_1))$	$m_2(G(m_2))$	$m_1 \times m_2$	$F_1 \wedge F_2$
1 (true)	0 (false)	0	false
1 (true)	1 (true)	1	true
0 (false)	0 (false)	0	false
0 (false)	1 (true)	0	false

From the table, it can be concluded that the HE computation for a single bit can achieve a homomorphic logical *XOR* computation. Homomorphic multiplication can implement homomorphic logical *AND* operations. Since OR operations have an identity property, homomorphic *OR* operations can also be realized.

$$F_1 \vee F_2 \equiv F_1 \oplus F_2 \oplus (F_1 \wedge F_2) \tag{14}$$

Assuming there are two Boolean values F_1 and F_2, whose corresponding integer plaintexts are $m1$ and $m2$, these are then encrypted to obtain the following ciphertexts:

$$\begin{aligned} \text{Encrypt}(pk, m_1) = c_1 = m_1 + 2r_1 + pq_1 \\ \text{Encrypt}(pk, m_2) = c_2 = m_2 + 2r_2 + pq_2 \end{aligned} \tag{15}$$

Calculate c based on the Formula (14), where c is the ciphertext of the corresponding integers:

$$\begin{aligned} c = F_1 \vee F_2 = c_1 + c_2 + c_1 c_2 \\ F_1 \vee F_2 = \mathrm{G}(Decrypt(sk, c_3)) \end{aligned} \tag{16}$$

The logical NOT operation is a unary operation. Firstly, it can be established that 1 itself can act as the ciphertext of 1, because

$$\mathrm{Decrypt}(sk, 1) = (1 \; mod \; p) \; mod \; 2 = \; 1, p \geq 2 \tag{17}$$

Since the value of p is always greater than 2, we can define a constant $const_1 = 1$ to represent the ciphertext of 1. Similarly, it can be deduced that the constant $const_0 = 0$ can represent the ciphertext of 0. Therefore, the logical NOT operation can be represented as follows:

$$\begin{aligned} m_1 &= G^{-1}(F_1) \\ \neg F_1 &= G(Decrypt(sk, c_1 + const_1)) \end{aligned} \tag{18}$$

After mapping OR and NOT logic to HE, other logical operations can be obtained through combinations of these three operations.

The above describes the HE process for a single bit. Following computer logic operations, all data are composed of a bitstream of $\{0, 1\}$. Therefore, we extend the single-bit approach to multiple bits to implement HE for all data. Compared to single-bit operations, long integers require consideration of carry-over in calculations.

This paper first utilizes Boolean operations to compare two positive integers. Table 3 shows the comparison results for a single bit.

Table 3. Single-bit integer comparison.

$m_1(G(m_1))$	$m_2(G(m_2))$	$m_1 > m_2$	$m_1 < m_2$	$m_1 = m_2$
0 (false)	0 (false)	false	false	true
0 (false)	1 (true)	false	true	false
1 (true)	0 (false)	true	false	false
1 (true)	1 (true)	false	false	true

This can be summarized as follows:

$$\begin{aligned} m_1 > m_2 \equiv G(m_1) \wedge (G(m_1) \oplus G(m_2)) \\ m_1 < m_2 \equiv G(m_2) \wedge (G(m_1) \oplus G(m_2)) \\ m_1 = m_2 \equiv \neg(G(m_1) \oplus G(m_2)) \end{aligned} \tag{19}$$

Assuming the key is pk, the plaintexts are m_1 and m_2, and the ciphertexts are c_1 and c_2, then the homomorphic computation method is as follows:

$$\begin{aligned} m_1 > m_2 \equiv G(\mathrm{Decrypt}(sk, c1(c1 + c2))) \\ m_1 < m_2 \equiv G(\mathrm{Decrypt}(sk, c2(c1 + c2))) \\ m_1 = m_2 \equiv G(\mathrm{Decrypt}(sk, c1 + c2 + 1)) \end{aligned} \tag{20}$$

Extending to multi-bit computation, assume there are two positive integers x and y. After converting to binary, x and y can be represented as follows:

$$\begin{aligned} x : x_n x_{n-1} \ldots x_1 \\ y : y_n y_{n-1} \ldots y_1 \end{aligned} \tag{21}$$

where $x_i, y_i \in 0, 1$. When $x > y$, the comparison process between x and y can be simplified as follows: if the highest bit $x_n > y_n$, then it is directly determined that $x > y$; otherwise, if $x_{(n-1)} \ldots x_1 > y_{(n-1)} \ldots y_1$, then it is determined that $x > y$. This process is repeated until the lowest bit.

Throughout this process, we define a variable $z_{(i+1)}$:

$$z_{i+1} = \begin{cases} 1, & x_i x_{i-1} \dots x_1 > y_i y_{i-1} \dots y_1 \\ 0, & x_i x_{i-1} \dots x_1 \leq y_i y_{i-1} \dots y_1 \end{cases} \tag{22}$$

This can be summarized as $z_{(i+1)} = 1$, meaning that $(x_i > y_i)$ or $(x_i = y_i$ and $z_i = 1)$. This summary is equivalent to the Boolean operation diagram shown in Figure 2.

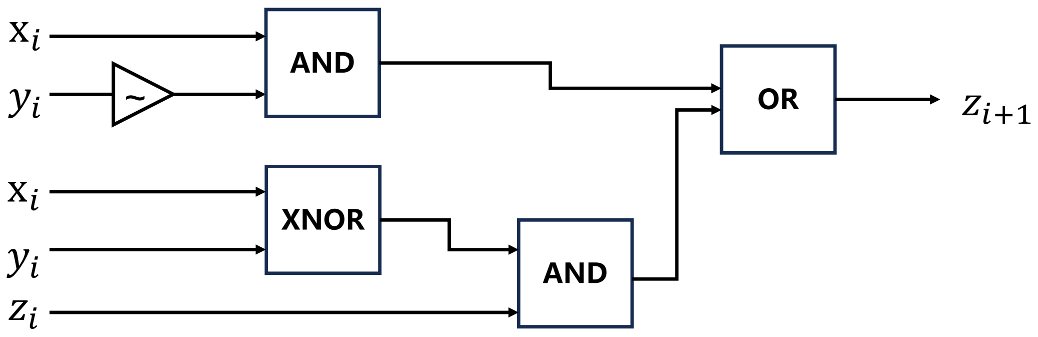

Figure 2. Boolean logic circuit.

Therefore, for positive integers x and y, by serially connecting the n of these Boolean operations as shown in Figure 3, a complete numerical comparison using HE can be formed, where $z_1 = 0$ and achieves multi-bit HE computation.

For the homomorphic computation of long integers, an adder is required. Considering the clear advantages of the multi-bit DGHV algorithm in computing positive integers, this paper only involves computations within the domain of positive integers. This computation is accomplished by simulating the principles of the binary complement. Table 4 is a binary addition table, where the result of adding a pair of binary numbers has two digits: one is the sum digit, and the other is the carry digit. Therefore, we can derive the logic table for the sum digit in Table 5, the logic table for the OR gate in Table 6, and the logic table for the $NAND$ gate in Table 7. Based on these, a full adder logic diagram can be constructed, as shown in Figure 4.

Table 4. Binary addition table.

+	0	1
0	0	1
1	1	10

Table 5. Logic table for the sum digit.

+	0	1
0	0	1
1	1	0

Table 6. Logic table for the OR gate.

OR	0	1
0	0	1
1	1	1

Table 7. Logic table for the $NAND$ gate.

NAND	0	1
0	1	1
1	1	0

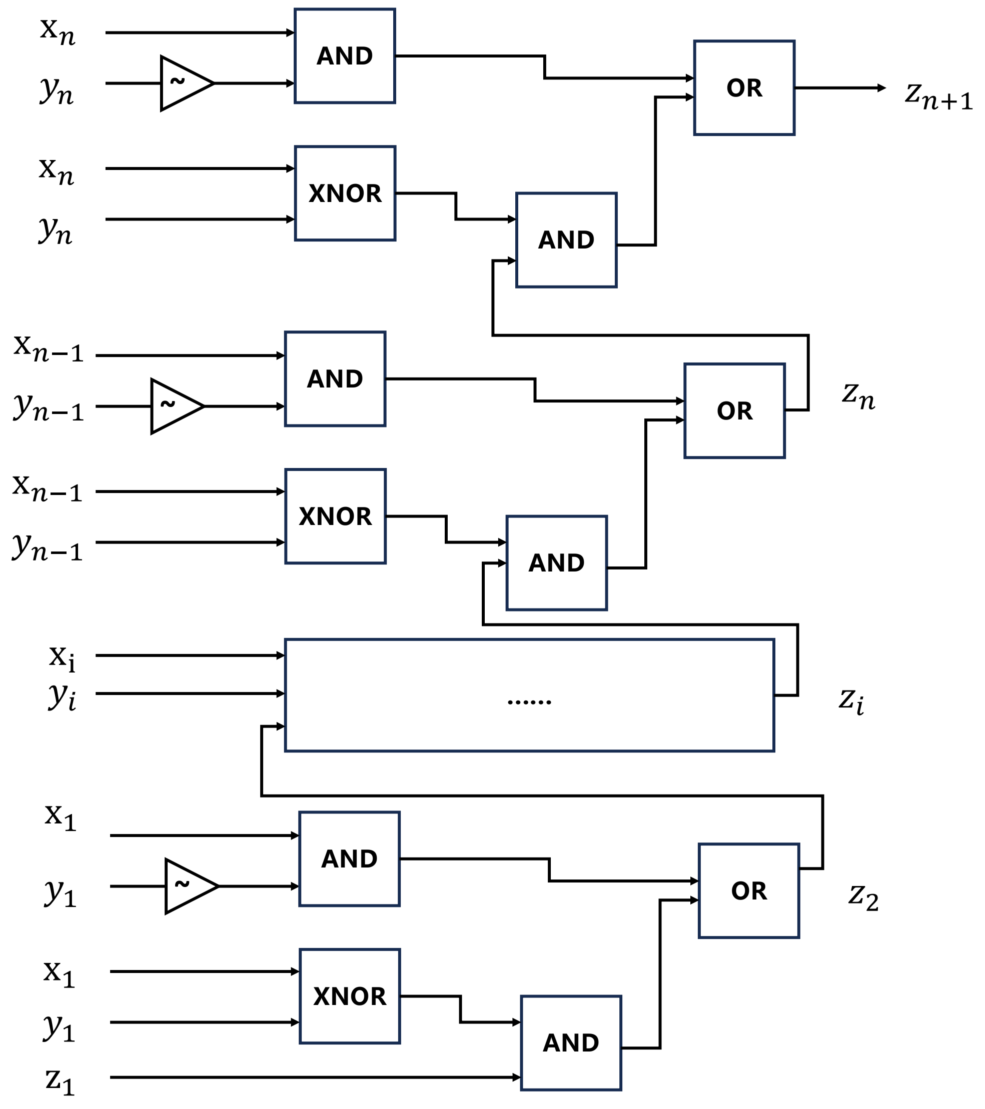

Figure 3. N-bit Boolean logic circuit.

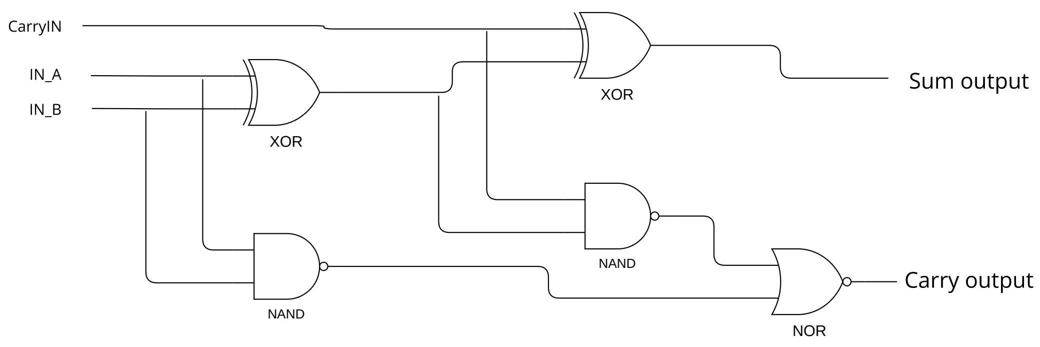

Figure 4. Full adder.

The full adder is composed of two half adders whose logic diagram is shown in Figure 4. Similarly, referring to a one-bit full adder, X and Y are the addends, E_{low} represents the carry-in from the lower bit, E_{high} represents the carry-out to the higher bit, and R is the sum.

$$\begin{aligned} R &= X \oplus Y \oplus E_{low} \\ E_{high} &= XE_{low} \vee YE_{low} \vee XY \end{aligned} \tag{23}$$

Constructing a homomorphic one-bit full adder:

$$\begin{aligned} R &= Decrypt(sk, c_1 + c_2 + E_{low}) \\ E_{high} &= Decrypt(sk, c_1E_{low} + c_2E_{low} + c_1c_2) \end{aligned} \tag{24}$$

By chaining one-bit full adders, a multi-bit homomorphic encrypted adder can be obtained. In the operation of E_{high}, two multiplications are required, and the error mainly accumulates during multiplication. Therefore, each time a full adder is chained, the error is magnified once. Thus, the main limitation of this method's error is the number of chained adders, i.e., the number of bits in X and Y.

The implementation of homomorphic multiplication relies on the principle of large-number multiplication. For example, considering two ten-digit binary numbers, X and Y, the following is assumed:

$$\begin{aligned} Y = 2^9y_9 + 2^8y_8 + 2^7y_7 + 2^6y_6 + 2^5y_5 + 2^4y_4 + 2^3y_3 + 2^2y_2 \\ +2^1y_1 + 2^0y_0 = \sum_{i0}^{9} 2^iy_i \end{aligned} \tag{25}$$

Therefore, we have the following:

$$X \times Y = X\sum_{i=0}^{9} 2^iy_i = \sum_{i=0}^{9} 2^iXy_i^4 \tag{26}$$

Thus, large-number multiplication can be performed by first multiplying the large number X with each digit y_i, and then shifting and adding them together. We know that in binary multiplication, we have the following:

$$(2^1 - 1) \times (2^1 - 1) < 2^1 \tag{27}$$

Therefore, the multiplication of two one-digit binary numbers does not produce a carry, which means the following:

$$\mathrm{n} \times \mathrm{m} = (\mathrm{n} \times \mathrm{m})mod2, \forall \mathrm{n}, \mathrm{m} \in \{0, 1\} \tag{28}$$

$$Xy_i = y_i\sum_{j=0}^{9} 2^jx_j = \sum_{j=0}^{9} 2^jx_jy_i = \sum_{j=0}^{9} 2^j\left(x_jy_i\right)mod2 \tag{29}$$

The problem of calculating Xy_i is simplified to multiplying single digits and then shifting. This solves the problem of homomorphic multiplication for large integers:

We compute all x_jy_i, where $i \in \{0, n_x - 1\}$, $j \in \{0, n_y - 1\}$, n_x and n_y are the number of bits in the large integers X and Y, respectively, and x_jy_i is the ciphertext of each bit.

We compute all Xy_j, where $j \in \{0, n_y - 1\}$, using the following formula:

$$\begin{aligned} Xy_j = \sum_{i=0}^{n_X - 1} 2^ix_iy_j = \sum_{i=0}^{n_X - 1} 2^ic_{i,j} \\ = \{c_{n_a-1,j}c_{n_a-2,j} \cdots c_{1,j}c_{0,j}\} \end{aligned} \tag{30}$$

$\{\cdots\}$ represents an n-bit integer. Then, we compute XY by appending j 0 at the end of Xy_j, equivalent to a left shift by j bits, and then perform the large integer addition to obtain $C = XY$, where C is the ciphertext.

Finally, we decrypt each bit of C, computing $m_k = Decrypt(sk, c_k)$, where $k \in \{0, n_x + n_b - 1\}$. The final result $\{m_k m_(k-1) \ldots m_1 m_0\}$ is the plaintext of the product.

This paper conducts experimental tests on the DGHV algorithm in the context of single-bit operations. From the described single-bit encryption process, it is understood that $keyGen(\lambda)$ generates a large odd number p as the secret key based on the security parameter λ, where η(bit) represents the bit length of the generated secret key p. Considering that the value of p needs to be large to ensure the stability of the system, we tested the impact of the value of η on stability.

5. Results

5.1. Research Question

Our experiments aim to answer the following research questions:

RQ1: How effective is PDPHE in protecting personal data during cross-border data transmission?

RQ2: What is the efficiency of PDPHE in the process of trans-border data transmission?

RQ3: What are the influencing factors of PDPHE in the process of cross-border data transmission?

5.2. Dataset

The dataset is categorized by industry into research data, medical data, and financial data. Within each industry, the data can be further detailed into the following categories: personal name, identity card number, phone number, deposit, weight, age, blood sugar, loan, etc. The specific details are shown in Table 8.

Table 8. Dataset.

Sector	Type	Format Requirements	Quantity/Items
Research Data	Personal Name	Measured in units	50
	Phone Number	Measured in units	50
	School	Measured in units	50
	Email	Measured in units	50
Medical Data	Weight	Measured in kg	50
	Age	Measured in years	50
	Blood Sugar	Measured in mg/dL	50
Financial Data	Loan	Measured in ten thousands	50
	Deposit	Measured in ten-thousands	50
	Salary	Measured in thousands	50
	Total		500

5.3. Implementation

We implemented the PDPHE model using Go1.14.3 on equipment from Google, located in Mountain View, United States with computational resources of 28 G CPU memory, and the system environment is Ubuntu 20.04 LTS. In validating the homomorphic encryption's capability to protect data privacy while enabling computation on encrypted data, this study employs a manually generated small-scale dataset comprising integers, floating-point numbers, or boolean values. Basic homomorphic encryption schemes are implemented using a homomorphic encryption library. Various operations including addition, subtraction, multiplication, and division are performed on different types of encrypted data. The primary focus of this paper is to measure the time overhead of encryption and decryption operations, as well as ensure the correctness of the computed results.

5.4. Result Analysis

5.4.1. PDPHE Effectiveness (Rq1)

Considering the lower computational performance of the machine and the large size of the dataset, this paper, in order to reduce the proportion of repetitive work and server pressure, only conducted experiments on single-bit HE. The single-bit encryption process $keyGen(\lambda)$ generates a large odd number p as the secret key based on the security parameter λ, where η is the bit length of the generated secret key p. Considering that the value of p needs to be large to ensure the stability of the system, we tested the impact of the value of η on the effectiveness of the model.

In the case of PDPHE, within our tested range $[2, 11]$, the results were 100% successful in completing the encryption and decryption process without errors. However, in the case of homomorphic multiplication, within our tested range, when $\eta \in [3, 8]$, the model is unstable. But when $\eta > 8$, the process of encryption and decryption stabilizes and can achieve 100% success in Table 9. This is due to the noise inherent in multiplicative homomorphism, which is within the normal and acceptable range. Considering key security, the value of η can be appropriately increased within the range allowed by the performance.

Table 9. Effective for homomorphic multiplication in PDPHE.

Type	**Accuracy**									
Method	η = **2**	η = **3**	η = **4**	η = **5**	η = **6**	η = **7**	η = **8**	η = **9**	η = **10**	η = **11**
Name	100%	55.08%	57.29%	55.26%	59.48%	62.86%	78.86%	100%	100%	100%
Phone	100%	59.77%	57.56%	56.36%	48.03%	62.70%	78.70%	100%	100%	100%
School	100%	58.96%	49.13%	53.89%	63.28%	62.53%	78.53%	100%	100%	100%
Email	100%	52.50%	50.75%	55.80%	54.22%	52.63%	68.63%	100%	100%	100%
Weight	100%	58.77%	58.37%	58.87%	62.82%	58.12%	74.12%	100%	100%	100%
Age	100%	47.79%	56.97%	54.66%	51.07%	59.08%	75.08%	100%	100%	100%
Blood Sugar	100%	56.16%	57.13%	58.86%	51.28%	46.72%	62.72%	100%	100%	100%
Loan	100%	52.33%	46.11%	54.49%	45.10%	45.02%	61.02%	100%	100%	100%
Deposit	100%	55.11%	54.96%	53.56%	58.15%	53.52%	69.52%	100%	100%	100%
Salary	100%	59.20%	53.17%	57.14%	58.47%	59.96%	75.96%	100%	100%	100%

5.4.2. PDPHE Efficiency (Rq2)

In the PDPHE model, symmetric key encryption is used to reduce the computation time of HE. Asymmetric key encryption is employed to encrypt the symmetric key, preventing the leakage of the symmetric key which could lead to the decryption of computation results. In this paper, the symmetric key algorithm is compared with traditional symmetric encryption algorithms such as DES and AES. Additionally, in the realm of asymmetric encryption algorithms, RSA and ECC are compared to select the most suitable key type for this model.

Currently, encryption algorithms can be divided into two types: asymmetric encryption and symmetric encryption. Both of these can serve as implementations for DGHV homomorphic encryption. However, due to the cross-border nature of this scenario and the requirement for high computational efficiency, the decision has been made to employ symmetric encryption, which is computationally efficient, for encrypting data. However, the drawback of symmetric encryption is that the encryption and decryption share the same key, making it susceptible to key leakage during transmission. To address this issue of key leakage, asymmetric encryption is used in this document to encrypt the symmetric key. The rationale behind this approach is that while using asymmetric encryption to encrypt data would require decryption every time, which could significantly impact transmission speed and computational efficiency due to the low decryption efficiency of asymmetric keys, encrypting the key with asymmetric encryption avoids the need for decryption every time, thus ensuring data security while improving computational efficiency.

To ensure the fairness of the tests, this paper uses a 100-character text as plaintext, with the value of η set to 9. We conducted 10 tests, timing from the generation of the key to its encryption, and the results are shown in Figure 5. From the figure, we can see that the overall duration of DES is significantly higher than that of AES and PDPHE, with the duration of PDPHE being slightly less than that of the AES algorithm. This is mainly because we chose a smaller η. If $\eta > 20$, then PDPHE would be significantly higher than AES, but with a noticeable improvement in security.

For the asymmetric encryption of the public key, we compared the advantages and disadvantages of ECC and RSA, as shown in Table 10, and conducted tests under the PDPHE model, the result is shown in Figure 6. Compared to RSA, ECC has absolute advantages in many aspects. In this model, its consumption time is also significantly less than that of RSA, so this paper selects the ECC algorithm as the asymmetric encryption algorithm for PDPHE.

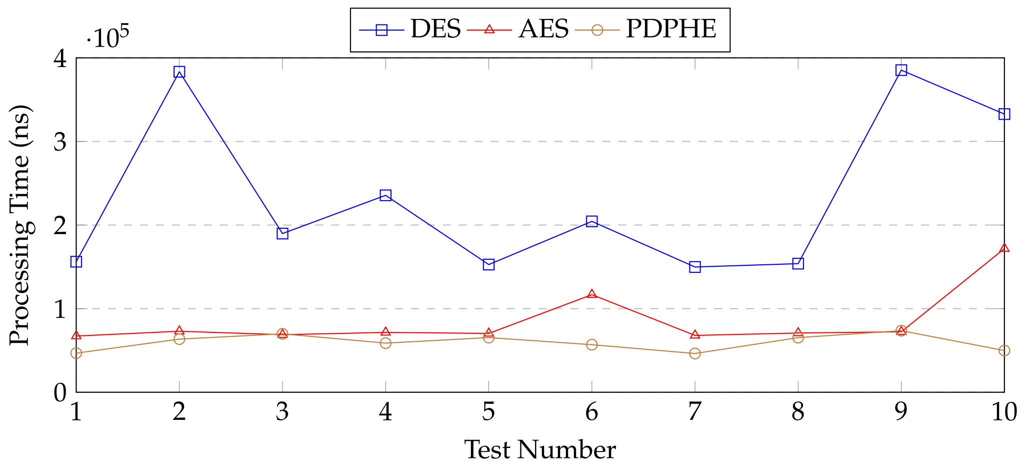

Figure 5. PDPHE, AES, DES algorithm comparison.

Table 10. Comparison between ECC and RSA Encryption.

Comparison Item	ECC Encryption	RSA Encryption
Key Length	256-bit	2048-bit
CPU Usage	Low	High
memory usage	Low	High
Bandwidth	Low	High
Implementation Complexity	Simple	Complex
Calculation Speed	Fast	Slow
Security Level	High	High

In this section, a comparison is made with other schemes in terms of security, execution time, and ciphertext size. Table 11 provides a comparison between this work and existing works, where reference [36] represents the currently more efficient encryption method. As can be seen from the table, the encryption efficiency of the method proposed in this work is not far from that method. When the text is small, the encryption time of this method is shorter, but as the text size increases, this method shows disadvantages compared to ref. [36], making this work more suitable for the encryption of small texts. Considering user privacy data, which is not large in volume, the PDPHE method is more practical in cross-border scenarios.

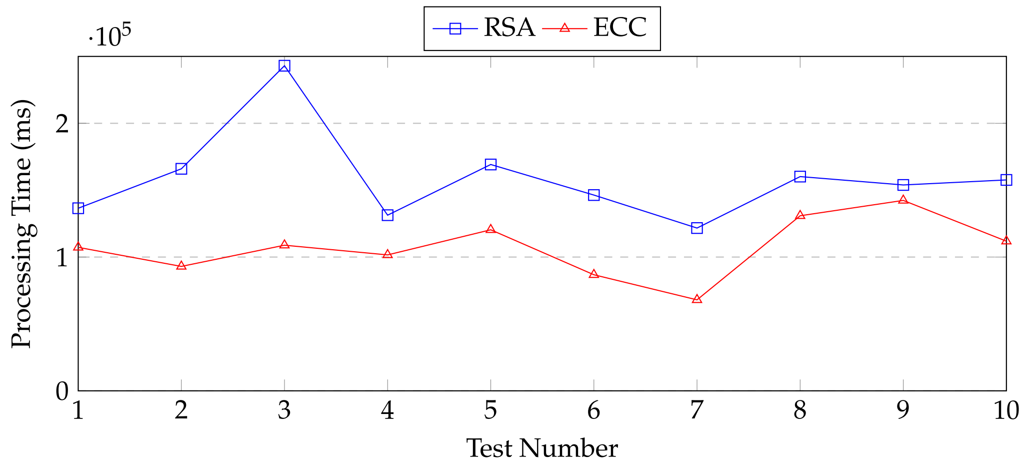

Figure 6. Performance comparison of RSA and ECC.

Table 11. Execution time for different plaintext lengths.

Plaintext Length	This Work	Ref. [36]	DGHV-FHE	SDC-FHE	SAM-FHE
12 bytes	4.32 ms	4.69 ms	1118 ms	1180 ms	1007 ms
1.4 Kbytes	23.47 ms	20.93 ms	124,182 ms	1,283,068 ms	20,818 ms
2.8 Kbytes	32.48 ms	27.003 ms	6,715,148 ms	4,425,899 ms	72,901 ms

In PDPHE, the robustness of security depends on choosing a large prime number as the secret key over any number by selecting the appropriate parameters for the scheme, and it has been proven to resist brute force attacks for at least 2λ times and other types of attacks. This can be explained by the cost of the prime number to the third party, where it is first important to verify that the number is prime, and then the prime number should be checked against the encryption equation, requiring multiple attempts to crack the code. Furthermore, the prime number offers a unique probability for decrypting the ciphertext.

5.4.3. Impact Factor for PDPHE (Rq3)

In the PDPHE model, the lengths of various integers are controlled by parameters γ, η, and ρ, which depend on the security parameter λ. When the security parameter is small, the encryption can be broken through extensive computation. However, if λ is large, solving an equation that requires exponential time in polynomial terms is impractical. As theoretically discussed in the previous sections of this paper, the model achieves the highest security rate and can realize 100% accuracy in encryption and decryption when $r \approx 2\sqrt{\eta}$ and $q \approx 2^{(\eta^3)}$. Therefore, we tested the impact of different r and q values on the PDPHE model.

Since r and q are positive random numbers, we generated their values through extensive experimentation. The security parameter λ is determined by η, with the value range of η being [0, 63). Meanwhile, when the range of η is within [9, 60], we define the security parameter levels of η to ensure the security level of encryption. These are categorized into minimal security parameter, small security parameter, medium security parameter, and large security parameter, as shown in Table 12.

Table 12. Security level.

Security Level	r	η	p	**Size of q/Byte**
Minimal	[9, 20)	[8, 20)	[256, 524,288)	[220, 2077)
Small	[20, 30)	[20, 39)	[524,288, 536,870,912)	[2077, 7388)
Medium	[30, 45)	[39, 97)	[536,870,912, 17,592,186,044,416)	[7388, 25,807)
Large	[45, 60]	[97, 212]	[17,592,186,044,416, 1,152,921,504,606,846,976)	[25,807, 65,443]

It can be seen from Figure 7 that when the value of r is within the range set by η, the processing results for all types of data are correct. When $\eta = 9$, the corresponding range of r is $[8, 20)$, and the accuracy rate of the processed data is 100%. However, when r is not within the range set by η, problems occur in data processing. Overall, the further r deviates from the set range, the lower the accuracy rate of the processing results. When r is taken as $[97, 212]$, the accuracy rate of data processing is already around 50%.

Considering the limitations of computational resources, this paper selects $\eta = 10$ to adjust the r and p parameters, in order to verify the optimal solution for r and p, as shown in Figure 7.

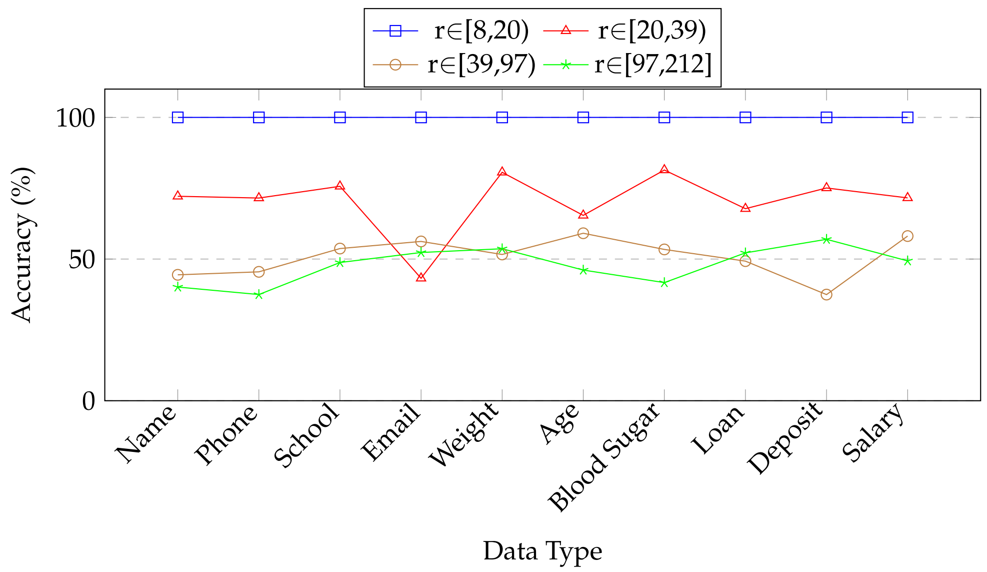

Figure 7. The effect of r on the accuracy for $\eta = 9$.

Due to the large values of q, this paper uses the size of the bytes occupied by q as the standard for testing, as shown in Figure 8. Similar to r, if the value of q exceeds the mapping range of q corresponding to η, the uncertainty in data processing will become apparent. As can be seen from Figure 8, when the value of q is changed, the accuracy rate of data processing fluctuates significantly. Since the value of q is large, its impact is also greater, and the randomness of the curve increases. When q exceeds the set range, it is impossible to distinguish the impact of the size of q on the results. After numerous experiments, it can be concluded that when $r \approx 2\sqrt{\eta}, q \approx 2^{\eta^3}$, the system exhibits better stability and the encryption effect is also optimal.

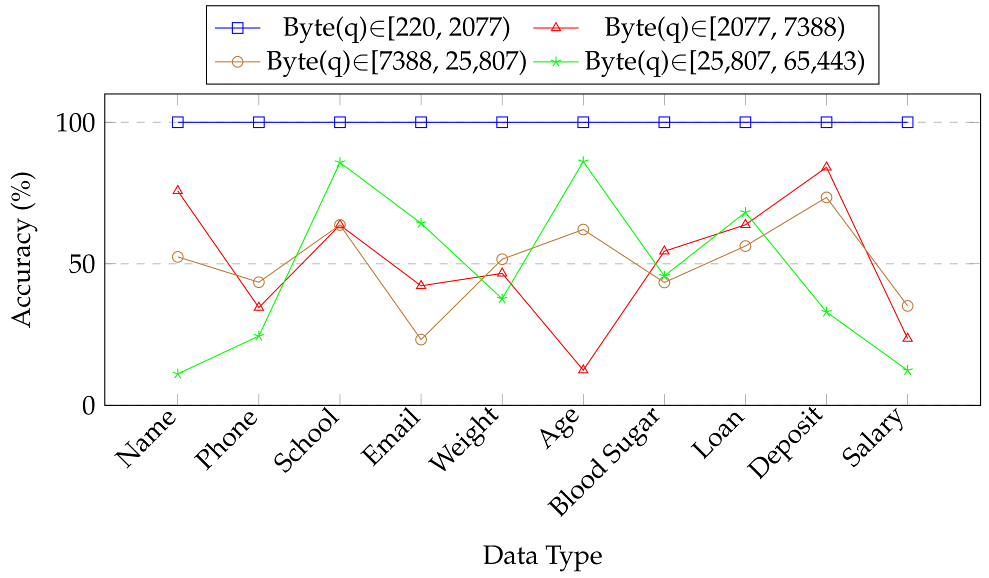

Figure 8. The effect of $Byte(q)$ on the accuracy for $\eta = 9$.

6. Conclusions

This paper presents the personal data protection homomorphic encryption (PDPHE) model, a novel approach designed to safeguard personal privacy in the context of cross-border data transmission.

The core innovation of the PDPHE model lies in its utilization of HE techniques, enabling computations on encrypted data without requiring access to the private key. This feature is particularly crucial in scenarios where sensitive data needs to be processed across different jurisdictions, ensuring that the confidentiality of personal information is maintained while still allowing for necessary data manipulations. A significant focus of this research is on optimizing the encryption and decryption processes to enhance system stability and performance. Through extensive experimentation, it was determined that the system stability and encryption effectiveness are optimal when the parameters r and q are set to $r \approx 2^{\sqrt{\eta}}, q \approx 2^{\eta^3}$, respectively. The PDPHE model stands out for its potential to revolutionize the way personal data are protected during cross-border transfers. Its implications extend beyond technical realms, offering a framework that could be adopted by policymakers and international data protection bodies to establish more secure and efficient data transfer protocols. However, our work still needs to progress further. Cross-border scenarios are complex and diverse, and many challenges remain:

- Most of the experiments in this paper were conducted in a single-bit environment. For multi-bit experiments, the requirements for bandwidth and server performance are quite stringent. How to enhance the security of HE while reducing computational overhead is a key focus for future research.
- Homomorphic encryption works well for small amounts of data, but in cross-border data transmission, the demand for large-scale data sharing is increasing. We need to use more universal methods to address this issue, among which secret sharing schemes are a direction of our research.
- The development of artificial intelligence increasingly demands distributed federated learning. However, some data in the federated learning process may be at risk of leakage. Ensuring data privacy during the federated learning process is also a problem we need to solve in the future.
- In the motivation section, we describe how privacy-preserving computation can mitigate differences in upper-level policies to ensure the normal flow of data. PDPHE can encrypt small amounts of data, such as personal privacy data, but its current encryption efficiency still cannot meet the demands of encrypting large batches of data. This is also a challenge in the practical application of the PDPHE algorithm.

Therefore, in the future, we will propose the concept of a trans-border trusted data space (TTDS) (the paper on this topic has been submitted and is still under review, so it is not cited). Within this data space, we will classify data, allowing most data, which do not require encryption, to flow freely. However, data defined as requiring encryption will be encrypted using the PDPHE method, thus enabling efficient circulation of all data.

Author Contributions: Conceptualization, Y.L. and W.W.; methodology, Y.L.; software , Y.L. and Q.L.; validation, Y.L., C.Y., C.Z. and M.X.; formal analysis, Y.L. and C.Y.; investigation, Q.L.; resources, Y.L. and C.Y.; data curation, Y.L.; writing—original draft preparation, Y.L.; writing—review and editing, W.W.; visualization, Y.L.; supervision, Y.L. and C.Y.; project administration, W.W. and L.C.; funding acquisition, C.Y. and Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Macao Science and Technology Development Fund Project under contract No. 0014/2023/EIB2.

Data Availability Statement: Due to the nature of this research, participants of this study did not agree for their data to be shared publicly, so supporting data are contained within the article.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Albrecht, J.P. How the GDPR Will Change the World. *Eur. Data Prot. Law Rev.* **2016**, *2*, 287–289. [CrossRef]
2. Gentry, C. A Fully Homomorphic Encryption Scheme. Ph.D. Thesis, Stanford University, Stanford, CA, USA, 2009.
3. Yao, A.C.C. How to generate and exchange secrets. In Proceedings of the 27th Annual Symposium on Foundations of Computer Science (SFCS 1986), IEEE, Toronto, ON, Canada, 27–29 October 1986; pp. 162–167.
4. Cheon, J.H.; Kim, A.; Kim, M.; Song, Y. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology—ASIACRYPT 2017, Proceedings of the 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 December 2017*; Springer International Publishing: Berlin/Heidelberg, Germany, 2017; pp. 409–437.
5. Mohassel, P.; Zhang, Y. Secureml: A system for scalable privacy-preserving machine learning. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 19–38.
6. Ben-Sasson, E.; Chiesa, A.; Genkin, D.; Tromer, E.; Virza, M. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *Advances in Cryptology—CRYPTO 2013, Proceedings of the 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2013*; Proceedings, Part II; Springer: Berlin/Heidelberg, Germany, 2013; pp. 90–108.
7. Ackerman, A.; Chang, A.; Diakun-Thibault, N.; Forni, L.; Landa, F.; Mayo, J.; van Riezen, R. Blockchain and Health It: Algorithms, Privacy and Data. Project PharmOrchard of MIT's Experimental Learning "MIT FinTech: Future Commerce". White Paper. August 2016. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3209023 (accessed on 12 January 2024).
8. Kuner, C. Data protection law and international jurisdiction on the Internet (part 1). *Int. J. Law Inf. Technol.* **2010**, *18*, 176–193. [CrossRef]
9. Kuner, C. Chapter 21: Data and extraterritoriality. In *Research Handbook on Extraterritoriality in International Law*; Edward Elgar Publishing: Cheltenham, UK, 2023. [CrossRef]
10. Greenleaf, G. Global Data Privacy Laws: 89 Countries, and Accelerating. Privacy Laws Business International Report 2012. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2000034 (accessed on 12 January 2024).
11. Chin, Y.-C.; Zhao, J. Governing Cross-Border Data Flows: International Trade Agreements and Their Limits. *Laws* **2022**, *11*, 63. [CrossRef]
12. Millard, C.J. (Ed.) *Cloud Computing Law*; Oxford University Press: Oxford, UK, 2013; Volume 2.
13. Obi, N.T., II; Zhao, B. The Balkanisation of the Internet: Data Nationalism in the European Union and Its Effects on the Development of Technology. 2022. Available online: https://arno.uvt.nl/show.cgi?fid=157767 (accessed on 12 January 2024).
14. Stolwijk, C.; Punter, M.; Timan, T.; Berkers, F.; Georgieva, I.; Gilsing, R.; Bastiaansen, H.; Hoekstra, M.; Yagafarova, A.; Mulder, W.; et al. Bridging the Dutch and European Digital Sovereignty Gap. 2022. Available online: https://publications.tno.nl/publication/34639349/urAkBu/TNO-2022-R10507.pdf (accessed on 12 January 2024).
15. Weber, R.H. Governance of the Internet of things—From infancy to first attempts of implementation? *Laws* **2016**, *5*, 28. [CrossRef]
16. Cate, F.H.; Mayer-Schönberger, V. Notice and consent in a world of Big Data. *Int. Data Priv. Law* **2013**, *3*, 67–73. [CrossRef]
17. Cattaruzza, A. The Geopolitics behind Data, the Data behind Geopolitics. Annales des Mines. 2020. Available online: https://www.academia.edu/44228266/The_geopolitics_behind_data_the_data_behind_geopolitics. (accessed on 12 January 2024).
18. Daniel, N.F. EU Data Governance: Preserving Global Privacy in the Age of Surveillance. Ph.D. Thesis, Johns Hopkins University, Baltimore, MD, USA, 2022.

19. Musoni, M.; Karkare, P.; Teevan, C.; Domingo, E. Global Approaches to Digital Sovereignty: Competing Definitions and Contrasting Policy. 2023. Available online: https://www.researchgate.net/profile/Poorva-Karkare/publication/371865306_Global_approaches_to_digital_sovereignty_Competing_definitions_and_contrasting_policy/links/6499ac47b9ed6874a5dcd499/Global-approaches-to-digital-sovereignty-Competing-definitions-and-contrasting-policy.pdf (accessed on 12 January 2024).
20. Luz, A.; Godwin Olaoye, O.J. Secure Multi-Party Computation (MPC): Privacy-Preserving Protocols Enabling Collaborative Computation without Revealing Individual Inputs, Ensuring AI Privacy. 2024. Available online: https://www.researchgate.net/profile/Joseph-Oluwaseyi-2/publication/379409115_Secure_Multi-Party_Computation_MPC_Privacy-preserving_protocols_enabling_collaborative_computation_without_revealing_individual_inputs_ensuring_AI_privacy/links/66076488390c214cfd24b406/Secure-Multi-Party-Computation-MPC-Privacy-preserving-protocols-enabling-collaborative-computation-without-revealing-individual-inputs-ensuring-AI-privacy.pdf (accessed on 12 January 2024).
21. Mouchet, C.V. Multiparty Homomorphic Encryption: From Theory to Practice (No. 8846). EPFL. 2023. Available online: https://infoscience.epfl.ch/record/305447 (accessed on 12 January 2024).
22. Saha, R.; Kumar, G.; Geetha, G.; Conti, M.; Buchanan, W.J. Application of Randomness for Security and Privacy in Multi-Party Computation. *IEEE Trans. Dependable Secur. Comput.* **2021**, *1*, 1–12. [CrossRef]
23. Rupa, C.; Greeshmanth Shah, M.A. Novel secure data protection scheme using Martino homomorphic encryption. *J. Cloud Comp.* **2023**, *12*, 47. [CrossRef]
24. Yang, W.; Wang, S.; Cui, H.; Tang, Z.; Li, Y. A Review of Homomorphic Encryption for Privacy-Preserving Biometrics. *Sensors* **2023**, *23*, 3566. [CrossRef]
25. Shah, P.; Prajapati, P. Provable data possession using additive homomorphic encryption. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 3448–3453. [CrossRef]
26. Rivest, R.L.; Adleman, L.; Dertouzos, M.L. On data banks and privacy homomorphisms. *Found. Secur. Comput.* **1978**, *4*, 169–180.
27. Gentry, C. Fully homomorphic encryption using ideal lattices. In Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, Bethesda, MD, USA, 31 May–2 June 2009; pp. 169–178.
28. Huang, H.L.; Zhao, Y.W.; Li, T.; Li, F.G.; Du, Y.T.; Fu, X.Q.; Zhang, S.; Wang, X.; Bao, W.S. Homomorphic encryption experiments on IBM's cloud quantum computing platform. *Front. Phys.* **2017**, *12*, 120305. [CrossRef]
29. Chen, H.; Laine, K.; Player, R. Simple encrypted arithmetic library-SEAL v2. 1. In *Financial Cryptography and Data Security, Proceedings of the FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, 7 April 2017*; Revised Selected Papers 21; Springer International Publishing: Cham, Swizerland, 2017; pp. 3–18.
30. Hamza, R.; Hassan, A.; Ali, A.; Bashir, M.B.; Alqhtani, S.M.; Tawfeeg, T.M.; Yousif, A. Towards secure big data analysis via fully homomorphic encryption algorithms. *Entropy* **2022**, *24*, 519. [CrossRef] [PubMed]
31. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
32. ElGamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
33. Halevi, S.; Shoup, V. Algorithms in helib. In *Advances in Cryptology–CRYPTO 2014, Proceedings of the 34th Annual Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2014*; Proceedings, Part I 34; Springer: Berlin/Heidelberg, Germany, 2014; pp. 554–571.
34. Regev, O. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM* **2009**, *56*, 34. [CrossRef]
35. Turan, F.; Roy, S.S.; Verbauwhede, I. HEAWS: An accelerator for homomorphic encryption on the Amazon AWS FPGA. *IEEE Trans. Comput.* **2020**, *69*, 1185–1196. [CrossRef]
36. Mahmood, Z.; Ibrahem, M. Fully Homomorphic Encryption Scheme Over Integers Based on DGHV Scheme. *Control. Syst. Optim. Lett.* **2023**, *1*, 169–173. [CrossRef]
37. Albrecht, M.; Chase, M.; Chen, H.; Ding, J.; Goldwasser, S.; Gorbunov, S.; Halevi, S.; Hoffstein, J.; Laine, K.; Lauter, K.; et al. Homomorphic encryption standard. In *Protecting Privacy through Homomorphic Encryption*; Springer: Cham, Swizerland, 2021; pp. 31–62.
38. Feng, T.; Yang, P.; Liu, C.; Fang, J.; Ma, R. Blockchain data privacy protection and sharing scheme based on zero-knowledge proof. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1040662. [CrossRef]
39. Goldwasser, S.; Micali, S.; Rackoff, C. The knowledge complexity of interactive proof-systems. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 203–225.
40. Fiege, U.; Fiat, A.; Shamir, A. Zero knowledge proofs of identity. In Proceedings of the STOC87: 19th Annual ACM Conference on Theory of Computing, New York, NY, USA, 25–27 May 1987; Association for Computing Machinery: New York, NY, USA, 1987; pp. 210–217.
41. Adida, B. Helios: Web-based Open-Audit Voting. In Proceedings of the USENIX Security Symposium, San Jose, CA, USA, 28 July–1 August 2008; Volume 17, pp. 335–348.
42. Sasson, E.B.; Chiesa, A.; Garman, C.; Green, M.; Miers, I.; Tromer, E.; Virza, M. Zerocash: Decentralized anonymous payments from bitcoin. In Proceedings of the 2014 IEEE Symposium on Security and Privacy, IEEE, Berkeley, CA, USA, 18–21 May 2014; pp. 459–474.

43. Ben-Sasson, E.; Bentov, I.; Horesh, Y.; Riabzev, M. Scalable, Transparent, and Post-Quantum Secure Computational Integrity. Cryptology ePrint Archive. 2018. Available online: https://eprint.iacr.org/2018/46 (accessed on 12 January 2024).
44. Weng, J.; Weng, J.; Zhang, J.; Li, M.; Zhang, Y.; Luo, W. Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 2438–2455. [CrossRef]
45. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
46. Blakley, G.R. Safeguarding cryptographic keys. In Proceedings of the 1979 International Workshop on Managing Requirements Knowledge (MARK), New York, NY, USA, 4–7 June 1979; pp. 313–318.
47. Ito, M.; Saito, A.; Nishizeki, T. Secret sharing scheme realizing general access structure. *Electron. Commun. Jpn. Part III Fundam. Electron. Sci.* **1989**, *72*, 56–64. [CrossRef]
48. Beaver, D. Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. *J. Cryptol.* **1991**, *4*, 75–122. [CrossRef]
49. Boneh, D.; Gentry, C.; Halevi, S.; Wang, F.; Wu, D.J. Private database queries using somewhat homomorphic encryption. In *Applied Cryptography and Network Security, Proceedings of the 11th International Conference, ACNS 2013, Banff, AB, Canada, 25–28 June 2013*; Proceedings 11; Springer: Berlin/Heidelberg, Germany, 2013; pp. 102–118.
50. Yang, Y.; Wei, L.; Wu, J.; Long, C. Block-SMPC: A blockchain-based secure multi-party computation for privacy-protected data sharing. In Proceedings of the 2nd International Conference on Blockchain Technology, Hilo, HI, USA, 12–14 March 2020; pp. 46–51.
51. Van Dijk, M.; Gentry, C.; Halevi, S.; Vaikuntanathan, V. Fully homomorphic encryption over the integers. In *Advances in Cryptology—EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Riviera, French 30 May–3 June 2010*; Proceedings 29; Springer: Berlin/Heidelberg, Germany, 2010; pp. 24–43.
52. Zhou, L.; Diro, A.; Saini, A.; Kaisar, S.; Hiep, P.C. Leveraging zero knowledge proofs for blockchain-based identity sharing: A survey of advancements, challenges and opportunities. *J. Inf. Secur. Appl.* **2024**, *80*, 103678. [CrossRef]