State-of-the-Art and Development of Privacy-Enhancing Technologies

Yulliwas Ameur and Samia Bouzefrane

March 2, 2023

State-of-the-art and development of Privacy-Enhancing Technologies

Yulliwas Ameur[1][0000−0003−1435−2982] and Samia Bouzefrane[1][0000−0002−0979−1289]

CEDRIC Lab, Cnam, 292 rue Saint Martin, Paris, 75141, France
yulliwas.ameur@lecnam.net

Abstract. Privacy-Enhancing Technologies (PETs) have been developed to securely process and confidently share sensitive data. The two main categories of PETs are those that focus on input privacy and those that focus on output privacy. Input privacy concerns how parties can manipulate data to prevent it from being used outside of a defined context, while output privacy focuses on modifying calculation results so that output data cannot be used to trace back to original inputs.
In this document, we will provide an introduction and overview of PETs by exploring various approaches in the literature. We will conclude this document by discussing the challenges in the adoption of these privacy-preserving technologies. Our aim throughout this paper is to provide a comprehensive understanding of PETs, their application, and the hurdles that must be overcome for widespread adoption.
The implementation of Privacy-Enhancing Technologies (PETs) enables the creation of secure data life cycles, promoting collaboration, trust, and security among those with a stake in the data.

Keywords: Privacy-Enhancing Technologies · Zero-Knowledge proofs · Private Set Intersection · Threshold Secret Sharing · Multiparty Computation · Fully-Homomorphic Encryption · Differential Privacy

1 Introduction to privacy-enhancing-technologies

Information and data are widely recognized as valuable resources for individuals, businesses, and governments. However, data breaches and misuses present significant security risks and privacy concerns. The cybersecurity landscape is full of dangers, and the collection, processing, and transfer of data across different entities and authorities pose unique privacy risks. The balance between the benefits and risks of data collection and use is a global policy debate. Despite evolving governance and legal frameworks, entities continue to use data with fragmented guidelines and oversight, especially concerning security, privacy, and appropriate use. For a long time, technical systems and tools such as data anonymization and encryption have been utilized to protect data while leveraging it to create value. However, with the increasing number of data breaches, these tools no longer provide the same level of protection. Privacy-enhancing technologies (PETs) have been developed to assist in the protection of data security and privacy. This report aims to define and categorize these PETs, offering a comprehensive overview to help regulators make well-informed decisions regarding data protection.

For each privacy-preserving technology, a brief overview is provided, and whenever possible, a reference to a survey is given systematically.

1.1 Zero-knowledge proof of knowledge:

Zero-knowledge proof of knowledge is a fundamental tool in the field of cryptography that plays a critical role in authentication and identification. This term refers to a secure protocol between two entities: the "proof provider" and the "verifier". The proof provider demonstrates, through mathematical proof, that a particular proposition is true while keeping all other information confidential, except for the truth of the proposition.

To further illustrate, imagine we want to attend a play in a theater. In this case, we would like to purchase a ticket without disclosing our personal information, such as our name or social security number. However, when we arrive at the theater as a customer, it is essential to verify the validity of our ticket without revealing any additional information to the ticket checker. This can be achieved using a zero-knowledge proof of knowledge protocol, in which the proof provider (ticket holder) proves to the verifier (ticket checker) the validity of the ticket without disclosing any personal information beyond the ticket's authenticity.

Goldwasser, Micali, and Rackoff [32] laid the foundation for zero-knowledge proofs, a cryptographic primitive that has garnered significant interest in the past few decades and has been widely applied across various domains. Typical statements are membership statements to an NP-language: the prover should demonstrate that a public word x belongs to a given language L from the class NP. Informally, the proof must satisfy three properties:

1. correctness: if the statement is true, and the prover knows a proof of this, he will succeed in convincing the verifier;
2. soundness: if the statement is false, no prover can convince the verifier of the truth of the statement;
3. zero-knowledge: the interaction yields nothing beyond the fact that the statement is true. This is captured by requiring the existence of a simulator that can produce an honest-looking transcript for the protocol, without knowing anything about the statement.

Zero-knowledge proof systems are typically interactive, with the prover and verifier exchanging several messages. Interactions are undesirable in some situations. In these cases, the conventional approach is to utilise a subset of zero-knowledge proofs known as non-interactive zero-knowledge proofs, which consist of a single flow from the prover to the verifier (after some one-time trusted setup). Although generic methods have been developed to convert classical zero-knowledge proofs into non-interactive ones, they only provide heuristic security guarantees [8].

Zero-knowledge proof schemes

There are currently several ZKP constructions in use in the wild, with each having its own set of advantages and disadvantages. As illustrated in the table 1 from [46].

Table 1. Comparison of different Zero-knowledge proof schemes, in regards to their asymptotic efficiency, where n is the number of gates of the circuit and d its depth, trusted setup, post-quantum secure and Security assumption, q-PKE for q-Power Knowledge of Exponent ,AGM for the algebraic group model,q-SBDH for q-Strong Bilinear Diffie-Hellman,DLP for Discrete Logarithm Problem,CRHF for Collision Resistant Hash Functions

Scheme	Trusted setup	Post -quantum secure	Prove	Verify	proof size	Security Assumption
BSCTV [11]	per-statement	no	O(n log n)	O(1)	O(1)	q-PKE
Groth[34]	per-statement	no	O(n log n)	O(1)	O(1)	q-PKE
Sonic [39]	updatable	no	O(n log n)	O(1)	O(1)	AGM
Libra [53]	updatable	no	O(n)	O(d log n)	O(d log n)	q-SBDH, q-PKE
Bulletproofs [18]	no	no	O(n)	O(n)	O(log n)	DLP
zk-STARKs [10]	no	yes	O(n log^{2} n)	O(log^2 n)	O(log^2 n)	CRHF

Zero-knowledge proof libraries

Libraries like the one provided in this paper are required to develop Zero-knowledge proof applications. One of the main libraries for this purpose is libsnark[1], a C++ library for constructing zk-SNARKs that was used for a while by Zcash [35], based on the specific zk-SNARK construction introduced in [11], but also supporting [34]. Even though this library provides excellent benchmarks, one of its main drawbacks, as stated by the authors, is that it is not well-optimized for ARM architectures.

Bellman3[2], a Rust-based library for building zk-SNARKs that was developed and is currently used by Zcash, has similar benchmarks. Furthermore, we previously stated that a version coded in Solidity is required when developing DApps for the Ethereum blockchain. ZoKrates[3] is a Python toolbox for zk-SNARKs that is designed to generate Solidity versions for deployment on the Ethereum blockchain. A similar approach is taken by snarkjs[4], a JavaScript library for

[1] https://github.com/scipr-lab/libsnark
[2] https://github.com/zkcrypto/bellman/
[3] https://github.com/Zokrates/ZoKrates
[4] https://github.com/iden3/snarkjs

creating zk-SNARKs. It includes a clear API for generating trusted setups, generating proofs, and verifying them using a fairly secure MPC protocol. It also makes it simple to export the version in Solidity and deploy it to the Ethereum blockchain.

For further details on zero knowledge proofs, please refer to [41].

1.2 Secret Sharing

Secret sharing consists in distributing a secret - for example a key or a password - among several or a password - between several custodians. In the simplest scheme the secret can only be discovered if a sufficient number of depositories share the information of depositories share the information they have received and, on the other hand, a and, on the other hand, a smaller number of depositories do not provide any information about the secret. Example: access to a room containing highly sensitive material can only be granted if at least two be authorized only if at least two people present themselves with their keys, in order to prevent an isolated person from manipulating or stealing them without control.Such a secret sharing scheme was invented by by both Adi Shamir [50] and Blakley [15].

Secret Sharing Schemes

There are several types of Secret Sharing Schemes (SSS), including polynomial-based SSS, Chinese Remainder Theorem (CRT)-based SSS, Matrix Projection-based SSS, and Image Secret Sharing [68]. This classification is mainly based on the method used in the process of sharing and reconstruction:

Polynomial secret sharing scheme

Adi Shamir [50] proposed the Polynomial Secret Sharing scheme in 1979, demonstrating a method to share a secret integer a_0 using a 't-1' degree polynomial. The secret is the constant term of the polynomial. All secret sharing schemes construct the shares using random bits. T-1 random bits are required to distribute a one-bit secret share with a threshold of t shares. Entropy of (t-1) b bits is required to distribute a secret of b bits.

To encrypt and decrypt the secret message, MM. Sathik et al. [49] used the Polynomial Secret Sharing scheme. Two levels of encryption are used: the first level generates the sharing polynomial, and the second level computes the mean value of the coefficients. This mean value is then multiplied by the ASCII value of the plain text letters to be encrypted. The above method generates ciphertext. The function value for a specific identity value is distributed to participants as shares. Using Lagrange's interpolation formula, these shares are then used to reconstruct the secret.

Chinese remainder theorem (CRT)-based SSS

Asmuth and Bloom developed secret sharing schemes based on the Chinese Remainder Theorem (CRT) in 1983, in [4]. CRT is used to generate the shares distributed to participants. The secret value can be reconstructed by solving a system of congruence equations. The basic CRT can be used to solve simultaneous linear congruences with coprime moduli in a unique way.

The matrix projection-based SSS

Li Bai et alMatrix Projection SSS can share multiple secrets [6]. Wanmeng Ding et al [23] used matrix theory to analyze Shamir's Image Secret Sharing Method in 2018. Then, based on the additional capabilities such as share renewal or cheating detection capability, we have many variants such as Dynamic Secret Sharing, Proactive Secret Sharing, Secret Sharing with Veto Capability, Secret Sharing with Cheating Detection Capability, Verifiable Secret Sharing, and Robust Secret Sharing Schemes [47]. Apart from accomplishing the basic task of secret sharing, these variants also give participants additional control over the secret-sharing process.

Dynamic secret sharing

A threshold is not predefined in some secret sharing schemes, and any qualified subset of participants can be used to disclose the secret information. Such schemes are referred to as access structure-based secret sharing schemes. The dynamic secret sharing enables the dealer to change a specific access structure and allows various qualified subsets of participants to reconstruct different secrets. Researchers such as Harn et al. and Blakely et al. made early efforts to develop dynamic threshold or dynamic access structure-based systems, but Blundo et al. provided a model for the fully dynamic secret sharing scheme [14, 16]

For further details on Secret Sharing, please refer to [48].

1.3 Multiparty Computation

MPC is a mechanism that allows for distributed privacy-preserving computations in a collaborative setting. MPC enables the computation of mathematical functions based on input data from multiple parties, while providing formal guarantees of data confidentiality and the correctness of the computation result [22]. Yao [54] first proposed MPC to solve the millionaires problem, in which two millionaires want to know who is richer without disclosing the size of their fortunes. This technique can be applied to existing data sets, which are then divided and distributed to various service providers or analysts, or it can be applied to data that already exists across multiple organizations. The use of multi-party computation across already distributed data has the additional benefit of never combining it in a single centralized database, reducing risk even further. This technology is especially promising for activities that require large amounts of data, which, if aggregated together, could pose additional risks. At a smaller scale, a form of multi-party computation can be performed by querying or requesting confirmations from databases without revealing underlying information. To further protect the confidentiality, these systems can hide the query itself from the entity holding the database or dataset. A relevant use case could be confirming that someone possesses a specific financial or physical asset, or a degree from an issuing institution. A system can allow that query to be sent between entities, an analysis on a database is performed, and the binary answer is returned. Secure multiparty computation (SMC) is a powerful technique for preserving the privacy and security of shared data. However, it is not without limitations. One of the main drawbacks of SMC is computational overhead. To ensure the security of the computation, random numbers must be generated, which can slow down run time due to the computational resources required. Additionally, SMC involves secret sharing, which requires communication and connectivity between all participants. This results in higher communication costs as compared to plaintext compute. While SMC is an important tool for protecting sensitive data, its limitations must be carefully considered in any implementation. NIST has recently released NISTIR 8214C entitled "NIST First Call for Multi-Party Threshold Schemes (Initial Public Draft)" which is aimed at soliciting feedback from the cryptographic community. This call for comments is divided into two categories, the first of which pertains to NIST-specified primitives (cat1), while the second category is reserved for primitives not specified by NIST (cat2). The public comment period will remain open until April 10, 2023. Given the critical importance of the cryptographic community's input in advancing this initiative, its active participation is highly encouraged.

For further details on Multiparty Computation, please refer to [38].

1.4 Private Set Intersection

Private Set Intersection (PSI) is a potent cryptographic methodology that enables two parties to determine the intersection of their datasets, all while safeguarding the confidentiality of their raw data from the other party. This technique has been employed by Apple in its Password Monitoring feature[5], and has been suggested as a potential solution for its recently announced Expanded Protections for Children.

In general, PSI (Private Set Intersection) protocols can be classified into two types, namely traditional PSI and delegated PSI. In the traditional PSI category, data owners engage in direct interaction with each other and require a copy of their set during computation, as illustrated in [29]. In contrast, the delegated PSI category enables delegation of the PSI computation and/or set storage to a third-party server, which may potentially be a passive or active adversary. The delegated PSI category can be further divided into two subcategories, namely one-off delegation and repeated delegation. One-off delegation protocols require

[5] https://support.apple.com/fr-fr/guide/security/sec78e79fc3b/web

the data owner to re-encode their data and send the encoded data to the server for each computation, as exemplified in [36]. Conversely, repeated delegation protocols permit data owners to upload their encrypted data to the server once and reuse it for multiple computations performed by the server, as demonstrated in [1]. Recently, new variants of the PSI protocol that support data update have been proposed by researchers in both the traditional and delegated categories, as described in [1] and [5]. These types of PSI protocols enable data owners to insert or delete set elements from their data while preserving privacy.

For further details on Multiparty Computation, please refer to [30].

1.5 Fully-Homomorphic Encryption

Fully homomorphic encryption enables any entity (for example, a cloud provider) to operate on encrypted private data without ever decrypting it. HE's goal is to perform operations on plain text while only manipulating ciphertexts. Typically, we must decrypt them before performing the desired processing on encrypted data.

Some operations are possible under certain cryptosystems with algebraic structures. For example, two RSA ciphertexts [44] can be multiplied to obtain the multiplication of the two corresponding plain texts. This is known as the multiplicative homomorphic property of the "textbook RSA" cryptosystem. On ciphertexts, another operation can be performed. For example, in the Paillier cryptosystem [42], we can add two ciphertexts to obtain the addition of the two corresponding plain texts. This is referred to as the "Paillier" cryptosystem's additive property. For example, if we are interested in e-voting applications, we can use this to add encrypted votes without knowing the initial vote.

Rivest, Adelman, and Dertouzos first proposed the concept of homomorphic encryption in [44]. Until Gentry's work [31], building a cryptosystem with both multiplicative and additive properties was a significant problem in cryptography. Gentry proposed the first fully homomorphic encryption algorithm based on ideal lattices. The HE is classified into three types based on the number of mathematical operations performed on the encrypted message: Partially Homomorphic Encryption (PHE), Somewhat Homomorphic Encryption (SHE), and Fully Homomorphic Encryption (FHE) (FHE) (FHE).

Homomorphic Encryption schemes

The following steps are shared by all HE schemes: key generation, encryption, decryption, and homomorphic operations on the ciphertexts.

Table 1 from [3] summarizes the most widely used and researched schemes in the cryptographic community, and Figure reffig:my labelA depicts the Homomorphic Encryption timeline.

Table 2. Comparison of HE schemes: SIMD for single-instruction-multiple-data

Operation	SCHEMES				
	BFV [28]	BGV [17]	CKKS [20]	FHEW[25]	TFHE[21]
Native Add/Sub	✓	✓	✓	✗	✗
Native Mult	✓	✓	✓	✗	✗
Boolean Logic	✗	✗	✗	✓	✓
SIMD	✓	✓	✓	✓	✓

Homomorphic Encryption Librairies

There are several open-source libraries available for implementing the HE scheme. For each scheme, they provide key generation, encryption, decryption, and homomorphic operations; library APIs frequently include features for maintaining and manipulating ciphertexts. Despite a lack of technical interoperability, there is also a lack of conceptual interoperability; for example, libraries that use the same scheme can produce surprisingly different results. Ongoing standardization efforts aim to create a unified view of the most popular schemes. Current HE methods are constrained. They are unable to easily support division operations and comparisons, such as the equality/inequality test.

Table 3. Overview of existing FHE librairies: CPU-targeting (top) and GPU-targeting (bottom)

Name	Input language	Supported schemes				
		BFV[28]	BGV [17]	CKKS[20]	FHEW[25]	TFHE[21]
HE-CPU-TARGETING						
Concrete	Rust	✗	✗	✗	✗	✓
FHEW	C++	✗	✗	✗	✓	✗
FV-NFlib	C++	✓	✗	✗	✗	✗
HEAAN	C++	✗	✗	✓	✗	✗
HElib	C++	✓	✓	✓	✗	✗
lattigo	Go	✓	✗	✓	✗	✗
PALISADE	C++	✓	✓	✓	✓	✓
SEAL	C++, .NET	✓	✓	✓	✗	✗
TFHE	C++	✗	✗	✗	✗	✓
HE-GPU-TARGETING						
cuFHE	C++, Python	✗	✗	✗	✗	✓
nuFHE	C++, Python	✗	✗	✗	✗	✓

Fully Homomorphic Encryption schemes may have limitations right now, but computer researchers are working hard to overcome them. There is no doubt that FHE will become the encryption standard of the future. You should keep an eye on its progress, and as it becomes more and more practically applicable to your industry, you should consider using it as soon as possible. In recent years, there have been several exciting developments in the field of Fully Homomorphic Encryption (FHE). One area of progress has been the updates to FHE libraries, which have made it easier for developers to work with this technology. Additionally, there have been advancements in FHE accelerations and compilers, which have helped to improve the efficiency and performance of FHE systems. As a result, FHE is becoming more accessible and practical for a wider range of applications and requirements, such as secure data sharing and privacy-preserving machine learning. In response to this growth, there has also been work towards establishing FHE standards and regulations, which will help ensure the safety and security of FHE systems. Finally, there has been a surge of interest in the FHE industry, with new companies and startups emerging to develop and promote this technology. Overall, these developments indicate a bright future for the field of FHE, and suggest that we can expect to see even more exciting advancements in the years to come.

For further details on Fully Homomorphic Encryption, please refer to [2].

1.6 Group signature

A group signature scheme is a method that allows a member of a group to sign a message anonymously on behalf of the group. David Chaum [19] and Eugene van Heyst pioneered the concept in 1991. A group signature scheme, for example, could be used by an employee of a large company where a verifier only needs to know that a message was signed by an employee but not which employee signed it. Another application is keycard access to restricted areas where tracking individual employee movements is inappropriate but it is necessary to secure areas to only employees in the group.

For further details on group signature schemes, please refer to [56].

1.7 Differential Privacy

Many definitions of privacy have been proposed in the literature, but finding a formal definition of privacy in the context of data analysis proved to be extremely difficult. The need to find a definition that meets the expectations of users while also being practical gives rise to the concept of differential privacy. "Differential privacy," as defined by cite [27], is a promise made by a data holder or curator to a data subject: "You will not be affected, adversely or otherwise, by allowing your data to be used in any study or analysis, regardless of what other studies, data sets, or information sources are available." In other words, an adversary with access to the data cannot determine whether or not an individual's information is contained in the data. Differential privacy is a mathematical framework for protecting individuals' privacy in a dataset. This framework operates by introducing noise into the input data, models, or outputs. The goal is to maximize data utility as well as privacy. It is critical to understand that DP is a property of an algorithm, not of the data itself. More formally, we can define DP as follows for two datasets that differ in only one individual (two neighboring datasets): [26].

Definition 1. *A randomized algorithm M gives (ϵ, δ)-differential privacy if for all datasets D_1 and D_2 differing on at most one element, and all $S \subset Range(M)$,*

$$Pr[M(D_1) \in S] \leq e^{\epsilon} \times Pr[M(D_2) \in S] + \delta \tag{1}$$

The privacy budget is commonly referred to as *epsilon*. It determines how much privacy protection the differential privacy algorithm provides. The privacy budget is expressed as a real value, with a lower epsilon value representing a stronger privacy guarantee. The failure probability of the differential privacy mechanism, on the other hand, is represented by *delta*. When delta equals zero, the mechanism is referred to as pure differential privacy (pure-DP), which means it provides the strongest privacy guarantee possible.

This definition yields three major properties: composition, post-processing, and group privacy. These properties will assist us in designing useful algorithms that satisfy DP while also ensuring that the algorithms provide accurate results.

– Composition: it provides a way to limit the privacy cost of answering multiple queries on the same data.
– Postprocessing : it ensures that the privacy guarantees of a differential privacy mechanism remain unchanged even if the output is further processed or analyzed
– Group privacy : this definition can be extended to group privacy by considering two datasets that differ on at most k rows rather than one row.

As previously stated, DP is a property of an algorithm or mechanism. The literature contains several basic mechanisms that meet the definition of differential privacy, including the Laplace mechanism, the Gaussian mechanism, and the exponential mechanism. The principles of these mechanisms are explained below:

– **The Laplace mechanism :**
It adds noise to the output of a Laplace distribution algorithm $Lap(\frac{\Delta_s}{\epsilon})$, where ϵ is the privacy budget and Δ_s is the sensitivity of the applied function. The sensitivity is a measure of how much the output can change in response to a small change in the input data. This is referred to as ϵ-differential privacy.
– **The Gaussian mechanism :**
It is a different mechanism than the Laplace mechanism. Instead of a Laplace distribution, it adds noise sampled from a Gaussian distribution. The Gaussian mechanism is not pure ϵ-differential privacy, but it is (ϵ, δ)-differential privacy.
– **The exponential mechanism :** It is a randomized algorithm that chooses an output from a set of input data with a probability proportional to the exponential of a score function. Each data item is assigned a score by the

score function based on its quality or relevance. The exponential mechanism is well suited for applications where data item quality is important, such as recommendation or voting systems.

These are only the basic mechanisms that ensure differential privacy. Many other mechanisms and algorithms have been proposed in the literature, each with its own set of advantages and disadvantages. The mechanism chosen is determined by the application's specific privacy requirements.

1.8 Attribute-based encryption

In 2005, Sahai and Waters [45] proposed Attribute Based Cryptography (ABC) as a new technique for encrypted access control. Ciphertexts are not necessarily encrypted to a single user in ABC, as they are in traditional public key cryptography. Instead, both users' private keys and ciphertexts are associated with a set of attributes or an attribute policy. If there is a match between the user's private key and the ciphertext, the user can decrypt it. This section provides a brief overview of ABC:
Sahai and Waters presented a threshold Attribute Based Encryption (ABE) scheme in [45]. In other words, ciphertexts are labeled with a set of attributes S, whereas a user private key is associated with both a threshold parameter t and a different set of attributes S. Enciphered data can only be decrypted if at least t attributes match between the ciphertext and the user private key. One of the primary motivations for this work was to create an error-tolerant (Fuzzy) identity-based encryption scheme that could be used with biometric identities.

Goyal et al. proposed a key-policy attribute based encryption (KP-ABE) scheme [33] in 2006 that integrated the access policy into the user private key. In other words, ciphertexts are

Private keys are associated with access structures that control which ciphertexts a user is able to decrypt and are labeled with sets of attributes. KP-ABE schemes provide fine-grained access control. In fact, data is outsourced in encrypted form, with different users still permitted to decrypt different pieces of data in accordance with security policy. This removes the need to rely on the storage server to prevent unauthorized data access. The disadvantage of KP-ABE is that the access policy is based on a single user's private key. As a result, the data owner has no choice but to select a set of attributes that can describe the data.

the outsourced data. Furthermore, the sender must trust that the key-issuer will grant or deny access to the appropriate users. Bethencourt et al. [12] presented the first implementation of a ciphertext policy attribute based encryption (CP-ABE) scheme. The user secret key is associated with a set of attributes in their scheme, and the ciphertext is associated with an access policy over attributes. The user can decrypt the ciphertext if and only if his secret key's attribute set matches the access policy specified in the ciphertext. [12] is conceptually superior. Role Based Access Control is an alternative to traditional access control methods (RBAC). Because of the computation properties on attributes, attribute-based cryptography (ABC) is referred to as an innovative concept and one of the most appealing ways to manage and control file sharing in the cloud. Traditional access control architectures, in fact, generally assume that remote servers storing data are completely trusted by their clients. As a result, they are frequently in charge of developing and enforcing access control policies. However, this is not the case. Because of the abstract nature of this business model, this statement does not usually hold true in multi-tenant cloud data storage environments. As a result, cloud clients are still hesitant to outsource their data file contents. ABC is regarded as a promotional solution for ensuring fine-grained access control to data stored on untrusted storage servers. First, ABC allows you to search through encrypted data. In other words, the ciphertext is given a set of descriptive attributes. In such a system, viewing these attributes as keywords results in a keyword-based search on encrypted data. Second, despite the fact that data is outsourced in encrypted form, each authorized user is able to decrypt different pieces of enciphered content due to the security policy included in the ciphertext and a required match with the decrypting key of the recipient. This removes the need to rely on the cloud storage server to prevent unauthorized data access. There are several common drawbacks to the works mentioned above. First, they usually assume that the system will be run by a single trusted authority. This may not only result in a load bottleneck, but it also has the key escrow issue. In fact, this entity has access to all encrypted files, potentially exposing personal information. Furthermore, delegating all attribute management tasks to a single entity is impractical, including certifying all user attributes or roles and generating secret keys. Different organizations, for example, usually define their domains. A professional association, for example, would be in charge of certifying medical specialties, whereas a regional health provider would certify the job ranks of its employees. Second, an efficient and on-demand user revocation mechanism for attribute-based schemes with support for dynamic policy updates and changes, which are critical components of secure sharing use cases, is still lacking.

For further details on Differential Privacy, please refer to [37].

1.9 Format preserving encryption

FPE (Format-preserving Encryption) is a technique of transforming plaintext into ciphertext that has the same format as the original text. Many applications exist for FPE for character data, such as database encryption and sensitive data protection in network transmission. It has some advantages over traditional block ciphers, such as increased data size and format change. When a 16-digit credit card number is encrypted, the ciphertext generated by AES is a block of binary data, but the ciphertext generated by FPE is another 16-digit number. The FPE problem was revealed in 1981 [43]. Ballare [7] developed the formal definition and security goals until 2009. Prefix, cycle-walking, and generalized-Feistel are the fundamental methods for integer domain provided by Black and Rogaway [13]. FFSEM [51], RtE [7], FFX [9], and other FPE methods are available. RtE and FFX are appropriate for the character domain described as, a string set where C is a finite character set and n is the number of characters in the string. By establishing a bijective relationship between string and integer, these modes convert string encryption into integer FPE. The FPE described above is known as NPE (Number-preserving Encryption), and it requires that the ciphertext character count be equal to the plaintext character count. NPE is only suitable for fixed-width character encoding. The National Institute of Standards and Technology (NIST) has recommended two methods for implementing format-preserving encryption (FF1 and FF3-1). Both FF1 and FF3-1 are intended to take a 128-bit block cipher (such as AES) and process its input and output to ensure that data is stored in the desired format while remaining decryptable to the original value. However, implementing an effective data protection strategy is more complicated than simply downloading a popular encryption algorithm and following basic implementation instructions. To avoid business disruptions, FPE algorithms must be carefully implemented to ensure the confidentiality, integrity, and availability of encrypted data while maintaining critical format requirements. For further details on Format preserving encryption, please refer to [52].

1.10 Federated learning

FL (Federated Learning) is a decentralized framework that enables collaborative training of a shared global model by multiple clients while keeping training data distributed on client devices, under the orchestration of a central server. The FL process is composed of several rounds, during which the server randomly initializes a global model as a starting point. At the beginning of each round, the server selects a subset of clients who participate in the training and distributes the current global model to them. Subsequently, each selected client trains the model locally on their own data and communicates only the updates to the model back to the server. Finally, the server aggregates the model updates received from the clients and incorporates them into the global model, concluding the current round. The most common approach to optimization for FL is the Federated Averaging algorithm [40] In this case, each client performs several epochs of minibatch stochastic gradient descent (SGD) minimizing a local loss function, and the central server then performs a weighted averaging of the updated local models to form the updated global model.

For further details on Federated learning, please refer to [55].

1.11 Data anonymization

Data anonymization is a form of information sanitization designed to safeguard individuals' privacy. Specifically, it entails the removal of personally identifiable information from datasets, thereby ensuring that the individuals represented within the data remain anonymous.

The anonymization of personal data is a process aimed at altering such data in a manner that would prevent any direct or indirect identification of a data subject by the data controller alone or in collaboration with any other party. This process can enable the transfer of information across boundaries, such as between departments within a single agency or between multiple agencies, while mitigating the risk of inadvertent disclosure. Additionally, in certain contexts, data anonymization can facilitate post-anonymization evaluation and analytics.

De-anonymization is the process of re-identifying an anonymous data source by cross-referencing it with other data sources[24]. The two most common anonymization approaches for relational data are generalization and perturbation.The process of obscuring data with the ability to re-identify it later is known as pseudonymization, and it is one way for businesses to store data in a HIPAA-compliant manner.

2 Barriers to the widespread adoption of privacy-enhancing technologies

Despite their growing demand and potential benefits, Privacy Enhancing Technologies (PETs) have significant drawbacks, some of which are more apparent than others. One of the most common criticisms of PETs is that they can be complex and difficult to use, leading to user errors that seriously compromise individual privacy and security. Additionally, many PETs are expensive and require significant computational resources that are not readily available to all market participants, not to mention the environmental impact of such resources.

The complexity and resource constraints of PETs can also make them difficult to audit or govern, potentially creating a false sense of security that undermines core data protection principles, such as data minimization. Due to usability and accessibility issues, as well as a lack of accountability, the use of PETs may actually encourage more data collection and sharing, further eroding privacy.

PETs present challenges in terms of increased architectural and operational complexity, as well as performance-related constraints that must be carefully considered when selecting a PET. Moreover, the limitations of PETs must be taken into account; for instance, homomorphic encryption and secure multi-party computation cannot handle large amounts of data, and do not satisfy anonymization requirements for secondary data processing. Notably, k-anonymity does not provide a formal guarantee of privacy, unlike differential privacy.

Despite these challenges, PETs will become increasingly important, particularly as the need for collaboration across organizational boundaries grows. In the future, there will be an effort to improve and expand the classification of PETs to include additional application categories and domains. Furthermore, there will be a focus on implementing and testing concrete PETs and use cases, such as differential privacy and secure multiparty computation.

References

1. Aydin Abadi, Sotirios Terzis, and Changyu Dong. Vd-psi: verifiable delegated private set intersection on outsourced private datasets. In Jens Grossklags and Bart Preneel, editors, *Financial Cryptography and Data Security*, volume 9603 of

Lecture Notes in Computer Science, pages 149–168. Springer Verlag, June 2016. Financial Cryptography and Data Security ; Conference date: 22-02-2016 Through 26-02-2016.

2. Bechir Alaya, Lamri Laouamer, and Nihel Msilini. Homomorphic encryption systems statement: Trends and challenges. *Computer Science Review*, 36:100235, 2020.
3. Yulliwas Ameur, Samia Bouzefrane, and Vincent Audigier. Application of homomorphic encryption in machine learning. In *Emerging Trends in Cybersecurity Applications*, pages 391–410. Springer, 2022.
4. Charles Asmuth and John Bloom. A modular approach to key safeguarding. *IEEE transactions on information theory*, 29(2):208–210, 1983.
5. Saikrishna Badrinarayanan, Peihan Miao, and Tiancheng Xie. Updatable private set intersection. *Proceedings on Privacy Enhancing Technologies*, 2022:378–406, 04 2022.
6. li Bai and Xukai Zou. A proactive secret sharing scheme in matrix projection method. *IJSN*, 4:201–209, 01 2009.
7. Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In *Selected Areas in Cryptography: 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009, Revised Selected Papers 16*, pages 295–312. Springer, 2009.
8. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, page 62–73, New York, NY, USA, 1993. Association for Computing Machinery.
9. Mihir Bellare, Phillip Rogaway, and Terence Spies. The ffx mode of operation for format-preserving encryption. *NIST submission*, 20(19):1–18, 2010.
10. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. *Cryptology ePrint Archive*, 2018.
11. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pages 781–796, 2014.
12. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *2007 IEEE symposium on security and privacy (SP'07)*, pages 321–334. IEEE, 2007.
13. John Black and Phillip Rogaway. Ciphers with arbitrary finite domains. In *Topics in Cryptology—CT-RSA 2002: The Cryptographers' Track at the RSA Conference 2002 San Jose, CA, USA, February 18–22, 2002 Proceedings*, pages 114–130. Springer, 2002.
14. Bob Blakley, GR Blakley, Agnes Hui Chan, and James L Massey. Threshold schemes with disenrollment. In *Advances in Cryptology—CRYPTO'92: 12th Annual International Cryptology Conference Santa Barbara, California, USA August 16–20, 1992 Proceedings 12*, pages 540–548. Springer, 1993.
15. G. R. Blakley. Safeguarding cryptographic keys. In *Managing Requirements Knowledge, International Workshop on*, page 313, Los Alamitos, CA, USA, jun 1979. IEEE Computer Society.
16. Carlo Blundo, Antonella Cresti, Alfredo De Santis, and Ugo Vaccaro. Fully dynamic secret sharing schemes. *Theoretical Computer Science*, 165(2):407–440, 1996.
17. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3), jul 2014.

18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE symposium on security and privacy (SP)*, pages 315–334. IEEE, 2018.
19. David Chaum and Eugène Van Heyst. Group signatures. In *Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10*, pages 257–265. Springer, 1991.
20. Jung Cheon, Han Kyoohyung, Andrey Kim, Miran Kim, and Yongsoo Song. *Bootstrapping for Approximate Homomorphic Encryption*, pages 360–384. 01 2018.
21. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Tfhe: fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, 2020.
22. Ronald Cramer, Ivan Damgård, Dario Catalano, Giovanni Crescenzo, Ivan Darmgård, David Pointcheval, and Tsuyoshi Takagi. *Multiparty Computation, an Introduction*, pages 41–87. 03 2006.
23. Wanmeng Ding, Kesheng Liu, Xuehu Yan, Huaixi Wang, Lintao Liu, and Qinghong Gong. An image secret sharing method based on matrix theory. *Symmetry*, 10(10):530, 2018.
24. Xuan Ding, Lan Zhang, Zhiguo Wan, and Ming Gu. A brief survey on de-anonymization attacks in online social networks. In *2010 international conference on computational aspects of social networks*, pages 611–615. IEEE, 2010.
25. Léo Ducas and Daniele Micciancio. Fhew: bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology–EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I 34*, pages 617–640. Springer, 2015.
26. Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 486–503. Springer, 2006.
27. Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.
28. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. *IACR Cryptol. ePrint Arch.*, 2012:144, 2012.
29. Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2004.
30. Ying GAO and Wei WANG. A survey of multi-party private set intersection. , 44:1–14, 2022.
31. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178. ACM, 2009.
32. S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, page 291–304, New York, NY, USA, 1985. Association for Computing Machinery.

33. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98, 2006.
34. Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 305–326. Springer, 2016.
35. Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. Zcash protocol specification. *GitHub: San Francisco, CA, USA*, 4:220, 2016.
36. Seny Kamara, Payman Mohassel, Mariana Raykova, and Saeed Sadeghian. Scaling private set intersection to billion-element sets. volume 8437, pages 195–215, 03 2014.
37. Ninghui Li. Differentially private data synthesis: State of the art and challenges. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, page 815, New York, NY, USA, 2022. Association for Computing Machinery.
38. Yehuda Lindell. Secure multiparty computation (mpc). Cryptology ePrint Archive, Paper 2020/300, 2020. https://eprint.iacr.org/2020/300.
39. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2111–2128, 2019.
40. H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, 2016.
41. Eduardo Morais, Tommy Koens, Cees Van Wijk, and Aleksei Koren. A survey on zero knowledge range proofs and applications. *SN Applied Sciences*, 1:1–17, 2019.
42. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.
43. FIPS PUB. Implementing and using the nbs data encryption standard. 1981.
44. R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation, Academia Press*, pages 169–179, 1978.
45. Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings 24*, pages 457–473. Springer, 2005.
46. Xavier Salleras and Vanesa Daza. Zpie: Zero-knowledge proofs in embedded systems. *IACR Cryptol. ePrint Arch.*, 2021:1382, 2021.
47. K.N. Sarma, Hemraj Lamkuche, and S. Umamaheswa. A review of secret sharing schemes. *Research Journal of Information Technology*, 5:67–72, 02 2013.
48. Parsa Sarosh, Shabir A Parah, and Ghulam Mohiuddin Bhat. Utilization of secret sharing technology for secure communication: a state-of-the-art review. *Multimedia Tools and Applications*, 80:517–541, 2021.
49. M. Mohamed Sathik and A. Kalai Selvi. Secret sharing scheme for data encryption based on polynomial coefficient. In *2010 Second International conference on Computing, Communication and Networking Technologies*, pages 1–5, 2010.
50. Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, nov 1979.

51. Terence Spies. Feistel finite set encryption mode. *NIST Proposed Encryption Mode. Available online at http://csrc. nist. gov/groups/ST/toolkit/BCM/documents/proposedmodes/ffsem/ffsem-spec. pdf*, 2008.
52. Kirill Denisovich Tsaregorodtsev. Format-preserving encryption: a survey. , 13(2):133–153, 2022.
53. Tiacheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III 39*, pages 733–764. Springer, 2019.
54. Andrew C. Yao. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 160–164, 1982.
55. Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.
56. Lei Zhang, Zhiyong Zheng, and Wei Wang. Survey of lattice-based group signature. In Zhiyong Zheng, editor, *Proceedings of the First International Forum on Financial Mathematics and Financial Technology*, pages 79–92, Singapore, 2021. Springer Singapore.