Privacy-preserving Cloud Computing on Sensitive Data: a Survey of Methods, Products and Challenges

Josep Domingo-Ferrer, Oriol Farràs, Jordi Ribes-González and David Sánchez

Universitat Rovira i Virgili, UNESCO Chair in Data Privacy, CYBERCAT-Center for Cybersecurity Research of Catalonia, Av. Països Catalans 26, E-43007 Tarragona, Catalonia
E-mail {josep.domingo, oriol.farras, jordi.ribes, david.sanchez}@urv.cat

Abstract

The increasing volume of personal and sensitive data being harvested by data controllers makes it increasingly necessary to use the cloud not just to store the data, but also to *process* them on cloud premises. However, security concerns on frequent data breaches, together with recently upgraded legal data protection requirements (like the European Union's General Data Protection Regulation), advise against outsourcing unprotected sensitive data to public clouds. To tackle this issue, this survey covers technologies that allow privacy-aware outsourcing of *storage and processing* of sensitive data to public clouds. Specifically and as a novelty, we review masking methods for outsourced data based on data splitting and anonymization, in addition to cryptographic methods covered in other surveys. We then compare these methods in terms of operations supported on the masked outsourced data, overhead, accuracy preservation, and impact on data management. Furthermore, we list several research projects and available products that have materialized some of the surveyed solutions. Finally, we identify outstanding research challenges.

Key words: Privacy, Data confidentiality, Public clouds, Sensitive data, Cloud computing

1 Introduction

Many companies are outsourcing at least some of their information technology to the cloud, from mere data storage to e-mail and other productivity applications. Reduced costs, no need for maintenance, virtually unlimited computational resources and increased availability are the main forces driving this

change. Yet, security and privacy misgivings are still cardinal barriers hindering a franker migration to the cloud.

Security is defined as achieving confidentiality, integrity and availability of the data outsourced to the cloud. Users want to be assured that no intruder can hack the cloud and/or impersonate them to steal or alter their sensitive data, and that no denial of service will occur. In the E.U., 57% of large enterprises using the cloud reported the risk of a security breach as the main limiting factor in the use of cloud computing services [1]; in a survey by the cloud Security Alliance to over 165 information technology and security professionals in the U.S., most of the respondents considered cloud storage as high risk [2]; the European Network and Information Security Agency identified "loss of governance" over the data outsourced to the cloud as a critically deterring factor [3]. Security breaches are, in fact, very real threats. Some well-known examples include the Sony PlayStation Network outage [1] as a result of an external intrusion, in which personal details from approximately 77 million accounts were stolen, the multi-day outage in Dropbox [2] that temporarily allowed visitors to log into any of its 25 million customer accounts as a result of a misconfiguration, or the leakage of private pictures of a number of celebrities from the Apple iCloud storage service due to weakly protected login credentials [3].

Regarding privacy, its most widely accepted definition in the information society is in terms of informational self-determination, that is, "the claim of individuals, groups or institutions to determine for themselves when, how, and to what extent information about them is communicated to others" [4]. Hence, for a cloud user to store and/or process sensitive data in the cloud, she needs the guarantee that no one other than herself —not even the CSP— will be able to see or infer her data. Thus, cloud computing needs to increase the user's control on her data, which will decrease the need for users blindly trusting the CSP. Otherwise, a user might be reluctant to outsource sensitive data to the cloud.

Furthermore, when data are personal, the individuals to whom the data refer —a.k.a. subjects— have privacy rights that have recently been enshrined in the new European Union's General Data Protection Regulation (GDPR [4]). To stay GDPR-compliant, a controller —an entity that has obtained consent from subjects to collect, store and process their data— can only outsource subject data to the cloud if she can obtain full control and confidentiality for the outsourced data. The above is a relevant issue because GDPR is also

[1] http://www.telegraph.co.uk/technology/news/8475728/Millions-of-internet-users-hit-by-massive-Sony-PlayStation-data-theft.html

[2] http://money.cnn.com/2011/06/22/technology/dropbox_passwords/

[3] http://www.bbc.com/news/technology-29237469

[4] http://www.gdpr-info.eu

becoming a *de facto* legal standard outside the European Union, specifically in the USA, Australia, Canada and Japan, and any company wishing to sell information technology solutions to those markets must take it into account.

Notice that privacy is even more challenging than security, because it must hold also with respect to (public and, therefore, untrusted) CSPs. In this respect, the cloud has given CSPs the opportunity to analyze and exploit large amounts of personal data. In fact, a report by the U.S. Federal Trade Commission [5] states that public CSPs regularly collect and analyze the data of their users without the latter's knowledge, and that those analyses could yield sensitive inferences; for example, a CSP could detect individuals that suffer from diabetes because of their interest in sugar-free products and share this information with an insurance company that could use that clue to classify a person as higher-risk (and possibly higher-premium).

One might argue that sensitive data handling in the cloud would be much simpler if the CSP could be assumed to be trusted. However, there are several legal issues here. On the one hand, in many scenarios the data subjects entrust the data controller with their personal data (for example, healthcare data), but this does not mean they allow the controller to further transfer their data to whomever the controller chooses to trust. On the other hand, the CSP may be under a jurisdiction different from the controller's. If, say, the CSP is under U.S. law whereas controller and subjects are under E.U. law, the latter law may be violated. Finally, many public CSPs offer their services free of charge in return for the possibility of monetizing users' data. For example, a recent privacy policy in Google [5] specifies that whatever information a user decides to outsource to any Google service can be used, reproduced, modified or distributed by Google with the aim of improving or promoting its services, but also to conduct targeted advertising (e.g., the Gmail filtering system scans the content of our emails to serve personalized ads).

To assuage the above issues and restore the user's control and trust on the protection of the data outsourced to the cloud, several solutions have been proposed in recent years. All of them involve masking sensitive data so that only protected values are stored in the cloud and only the user/controller owning the data is able to unmask the protected values retrieved from the cloud. However, if the user wants to use not only the cloud's storage but also the cloud's computational power, the challenge is even harder, because data protection should be made compatible with outsourced computations on cloud premises on masked data.

In this paper we survey the state of the art on security and privacy-enabling solutions towards the cloud, with a focus on those that preserve cloud service

[5] `http://www.google.com/policies/privacy`

functionalities, such as the ability to outsource queries and calculations on protected data to the cloud. In comparison with recent surveys on this area, ours offers the following contributions:

- Most surveys focus on data security vs external attackers [6,7,8,9] rather than on privacy versus the cloud. Therefore, they center their analysis on security attacks and on mechanisms to prevent, detect and mitigate them. In contrast, our survey considers mechanisms that protect outsourced data not only against third-party attackers, but also against insiders and *honest-but-curious clouds*[6] storing and managing such data.
- Many surveys concentrate on outsourced data storage [8,9,10]. Our paper goes a step beyond and puts the spotlight on the preservation of cloud service functionalities (e.g., queries, calculations, etc.) on the protected data outsourced to the cloud. Taking advantage of the cloud's computing power on protected data is significantly more challenging than merely using the cloud to store protected data.
- All surveys covering privacy-enabling service preservation mechanisms are limited to cryptographic solutions [11,12]. Even though these methods are powerful to secure data, they also result in significant calculation overheads (which partly neutralize the cost-saving benefits of cloud computing), they require key management, and they severely limit cloud functionalities on the outsourced data because they need tailored solutions for each type of outsourced calculation [12]. To be self-contained, we also discuss cryptographic solutions but, for the *first time*, we comprehensively *survey non-cryptographic methods* (based on data splitting and anonymization) that can be used to efficiently protect data outsourced to the cloud while preserving a variety of cloud services.
- In addition to the security and privacy-enabling solutions proposed in the literature, we survey research projects and products that implement some of these methods in the cloud scenario and discuss the outsourced functionalities they support.

In Section 2 we characterize the scenario we consider by identifying the actors and the entities involved. A common feature of privacy-enabling technologies towards the cloud is the use of a local proxy by the controller to mask the data before outsourcing them to the cloud. In this section we also present this general architecture and the assumptions and security models on which available solutions rely. We conclude the section with the identification and the description of the requirements that privacy-enabling technologies should ideally fulfill. In Section 3 we review data protection techniques, non-cryptographic or cryptographic, that have been or can be implemented in the aforementioned proxies to enable privacy and security towards the cloud while preserving (at

[6] An honest-but-curious cloud honestly fulfills the storage and processing services it offers, but may inspect the information that users store or process.

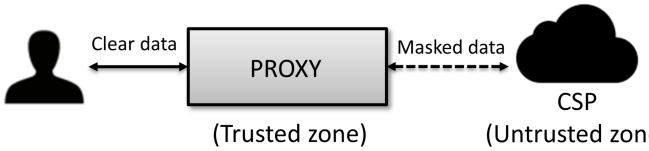

Fig. 1. General architecture of a trusted proxy that protects a user's data outsourced to the cloud

least some of) its functionalities. In Section 4 we present a critical comparison of the techniques covered in Section 3 with respect to the requirements identified in Section 2. In Section 5 we survey research projects and products that offer security- and privacy-enabling solutions towards the cloud, and classify them according to the type of data protection technique they employ. Finally, in Section 6 we gather the conclusions and several research challenges derived from the gaps identified in the survey.

2 Data protection towards the cloud: assumptions and requirements

Data protection towards untrusted clouds is usually accomplished by deploying a trusted local proxy. This proxy is a logical entity that can be located on the client side (for instance, the user's personal computer or smartphone) or, at least, in a location trusted by the user (such as a server or a router within the user's corporate intranet) [13]. Proxies can be implemented as additional modules or services for existing applications (with dedicated APIs), as browser plugins (that intercept and sanitize the content of http requests) or as standalone servers (which may handle and protect the data of several end users). A variety of security and privacy-enhancing masking transformations can be performed by these proxies to protect the sensitive data of users before outsourcing them to the cloud. When protected data are retrieved from the cloud as a result of the user's searches or requests for computations, the proxy unmasks the retrieved data before forwarding them to the user. This makes masking and the very existence of the proxy transparent to the user. Depending on the design of the proxy, the user may have more or less control on which data are protected and how. As discussed below, empowering cloud customers with control of their data is in fact crucial to cement trust on cloud computing.

Figure 1 shows the general architecture of a data protection proxy. Figure 2 depicts the workflows corresponding to protecting and outsourcing data to the cloud (Steps 1 and 2) and searching and retrieving data from the cloud (Steps 3-6).

The local data masking implemented by the proxy makes the outsourced data

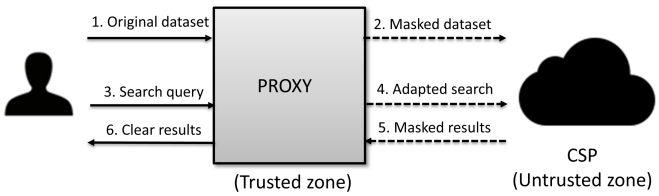

Fig. 2. Data workflows for protecting and outsourcing data to the cloud, and for searching and retrieving data from the cloud

robust against attacks that might disrupt the security and privacy of all cloud customers. For example, if an attacker acquires the encryption keys of a cloud customer, this affects the security of that customer, but not the security of other customers. On the other hand, insider attacks against the cloud provider, accidental data leakage or data mining analyses executed by the CSP on the customers' data will only reveal as much as each cloud customer allows.

The data protection architecture employed by current solutions (surveyed in Section 5) relies on some or all of the following assumptions:

- CSPs are *honest-but-curious*, which is the most common model in the cloud computing literature [14]. CSPs, as commercial providers of services, *honestly* fulfill their role; that is, they do not act maliciously against the user/customer (for example, by consciously altering or deleting data) and they follow the protocols as expected. However, even if honest, the cloud might be vulnerable to attacks either by outsiders or insiders, who might gain access to sensitive user data outsourced to the cloud. Furthermore, CSPs may be *curious* to analyze the data they store and the queries received from the users in order to acquire additional information. Being curious is usual in the case of CSPs that offer their services for free (like Google and others), who expect to derive income that sustains their businesses by analyzing and monetizing the data of users.
- In some scenarios, one can expect CSPs to be willing to collaborate with user-side data protection proxies, because these help building trust in the cloud among users. Since enhanced trust might motivate more users to migrate to the cloud, CSPs may see this collaboration as a market opportunity. In the most basic scenario, collaboration may imply that CSPs should provide a stable interface to their services (such as data storage, searches, or data operations). In other cases, it might imply that the CSP should alter the way the data are processed on the CSP premises to make some outsourced functionalities compatible with the masking technique implemented by the proxy on the data. In fact, making the cloud provider aware of data masking may increase the number of data protection techniques that can be applied on the data and the privacy-enabled services that can be supported.
- Some approaches rely on the increasingly popular notion of a *multi-cloud*,

which consists in the use of multiple cloud computing services in a single heterogeneous architecture [15]. Specifically, a multi-cloud offers a pool of locations in which a user's data can be stored. Multi-clouds add several advantages, such as reducing reliance on any single vendor, increasing flexibility or mitigating disasters. However, here we are specifically interested in the distributed and unconnected nature of multi-cloud services; this means that individual CSPs do not share information with each other and, in particular, they do not collude or pool together the data they hold on a certain user in the hope of breaching the user's privacy.

To accomplish their goal, the technologies employed by data protection proxies should fulfill most (if not all) of the following requirements:

- *User empowerment.* Since one of the main objectives of trusted proxies is to increase user trust in the cloud, it is crucial to empower users with control on which data are protected and how. Specifically, the user must be able to specify which of her data are sensitive, or even define different sensitivity degrees for individual data pieces. Additionally, or alternatively, the user may be allowed to select and configure the protection of her data in a way consistent with their sensitivity and, also, with the functionalities (e.g., searches, computations, etc.) that she desires to outsource to the cloud. It is important to note that most data masking techniques (especially the most secure ones) impair the outsourced functionalities to some extent [13], and the user should consider this circumstance to balance her privacy and functionality requirements. Also, proxies may implement trust-enhancing mechanisms, like auditing and monitoring tools, whereby users can check how their data are being protected and how they are stored and managed in the cloud.
- *Low overhead.* Cost savings and large computational power are fundamental benefits of cloud computing, and therefore performance should not be significantly penalized by the data protection. Specifically, privacy-enabled services should be implemented so that most of the computational load can be effectively outsourced to the cloud, thereby maintaining low workload and low hardware requirements on the trusted proxy. Likewise, response times of storage, search, retrieval and calculations on the cloud should not be significantly higher when using data protection proxies than when using the cloud services directly.
- *Transparency.* End users or applications using cloud services should not need to modify their behavior (for instance, function calls, parameters, protocols) due to the presence or absence of the trusted proxy. In other words, this means that the masking and unmasking actions implemented by the proxy should be made compatible and transparent with existing communication protocols, services and data formats.
- *Preservation of cloud functionalities.* Beyond data storage, cloud computing offers the possibility to outsource expensive computations on large data

sets to the cloud. However, making locally masked data compatible with outsourced computations on untrusted clouds is very challenging [16]. In fact, many privacy-enabling solutions available in the literature limit or reduce the flexibility of some functionalities on protected data, such as searches or computations [13]. In this sense, a combination of data protection techniques may be an adequate solution to compensate the limitations of individual methods and, therefore, preserve a wider spectrum of functionalities. The functionalities we consider in this survey are:

(1) Storage: the ability to outsource data storage to the cloud.
(2) Update: the ability to modify an individual data piece of an already outsourced data set (e.g., a record value in a remote database) without requiring to re-upload the whole data set or a significant part of it.
(3) Search and retrieval: the possibility of searching for data fulfilling one or several search criteria, and to retrieve the data matching the search, which will be unmasked by the trusted proxy prior to presenting them to the user. Search criteria include keywords and value ranges, which can be combined with logical operators (Boolean search).
(4) Computation: the possibility of outsourcing any type of computation (e.g., statistical) on data sets stored in the cloud, and to obtain an unmasked and accurate result on local premises. Depending on the complexity of the computation and the way the data have been masked before outsourcing them to the cloud, different data protection techniques and computation protocols may be designed, which imply more or less involvement of the trusted proxy to orchestrate the calculation and obtain the unmasked result. In any case, the goal should be to outsource as much computation workload as possible to the cloud.

- *Interoperability.* Many cloud services, such as those supporting collaborative edition of documents, require users from *different* organizations (using separate proxies with potentially different security configurations) to access the same data. Proxies should thus be able to interoperate in order to make these collaborative tasks possible. Ideally, no encryption keys or other key material should be exchanged/shared among proxies because doing so may weaken security.

In Section 4 we use the above requirements as *criteria to compare the data protection techniques* we survey in the next section.

3 Data protection techniques

As discussed in the previous section, local proxies may use different techniques to protect the user's data before outsourcing them to the cloud. Due to the variety of data protection scenarios, the research community has proposed a plethora of masking techniques. Although this diversity may seem bewildering,

it also has the advantage of making it possible to cope with heterogeneous privacy and functionality needs.

This section surveys the state of the art on functionality-preserving data protection techniques that have been or can be used as privacy-enabling mechanisms towards the cloud. Our analysis focuses on their ability to preserve the benefits of cloud computing: functionality, cost-effectiveness and efficiency. We will categorize techniques as follows: i) data splitting mechanisms, ii) data anonymization methods and iii) cryptographic techniques. As far as we know, this survey is the first one that covers the first two —non-cryptographic— categories in the cloud scenario. As it will be seen in the comparative analysis of Section 4, data splitting, anonymization and cryptographic techniques significantly differ in i) the level of protection they offer, ii) the amount of data accuracy they retain, iii) the overhead they introduce due to masking and unmasking operations, iv) the functionalities they allow on the masked data and v) the key material they require to mask and unmask data.

3.1 Data splitting

Data splitting is a protection technique based on fragmenting sensitive data [17] and storing the fragments *in clear form* in separate locations. Fragments should be such that a single fragment neither allows re-identifying the subject to whom it corresponds nor reveals confidential information that can be linked to a certain subject. For example, if a fragment consists of the values of an attribute 'Diagnosis', then clearly just knowing a list of diagnoses is useless to an intruder, because he cannot associate them with the corresponding subjects. Within the cloud scenario, data may be outsourced by a local proxy performing data splitting to either separate cloud accounts within the same CSP, or to different clouds, each one run by a different CSP providing the same kind of service (i.e., a multi-cloud [15]). For splitting-based data protection to be effective, storage locations should remain independent and unlinkable, so that CSPs cannot collude to aggregate partial data in the hope of breaking data protection.

In the following we detail the general workflow of data splitting as data protection method for the cloud (Section 3.1.1). In Section 3.1.2, we survey works on non-cryptographic data splitting, and we classify them according to the splitting criteria they use. Finally, in Section 3.1.3, we discuss how outsourced computations can be conducted on split data. Table 1 offers a summary of the surveyed methods.

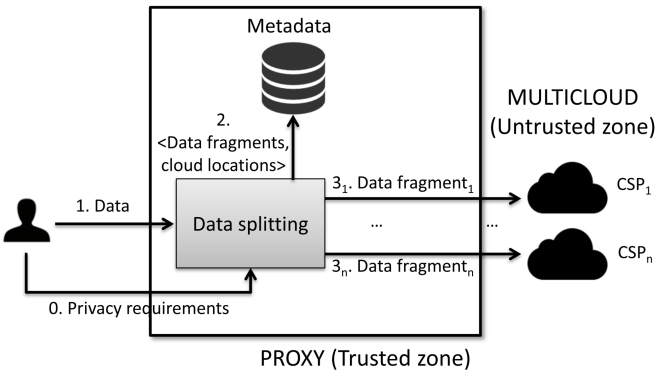

Fig. 3. Workflow of data storage and splitting to a multi-cloud

3.1.1 Data splitting workflow

The workflow of the data splitting and storage process is depicted in Figure 3. First, the proxy receives from the user the data to be outsourced, and assesses the disclosure risks. To do so, the proxy relies on the privacy requirements defined by the user (Step 0 in Figure 3), which state the set of attributes that may cause re-identification; that is, which attributes are identifiers or quasi-identifiers (see Section 3.2 for a definition of quasi-identifier). The proxy then decides how data are split and how many storage locations are needed to prevent disclosure; each piece of data that can be safely stored together constitutes a data fragment. It also stores the splitting criterion and the storage locations in a local database (Step 2) so that the system can properly process future queries on split data and aggregate the partial results. Finally, it forwards each data fragment to a separate CSP (Steps 3_1 to 3_n). In this manner, the splitting process is *lossless* (that is, it is possible to reconstruct the results that would be obtained on the original data set without having to store this data set on local premises) and privacy-preserving (since the fragments stored in each separate CSP are disclosure-free according to the privacy requirements) [18].

Even though some storage space is needed at the proxy for the splitting metadata, these only account for the correspondence between individual attributes and CSP locations. Since the number of attributes is usually many orders or magnitude smaller than the number of records, keeping metadata does not incur unacceptable storage requirements.

The fact that split data are stored in clear form makes it possible to seamlessly support a number of cloud functionalities (like keyword searches, conjunctive and range queries, univalued statistical calculations, etc.) by just broadcasting user queries to each storage location. Even though these functionalities provide partial results because they have been applied to data fragments, it is possible for the proxy, who knows the location of each fragment, to reconstruct the complete result by combining and aggregating partial results [18].

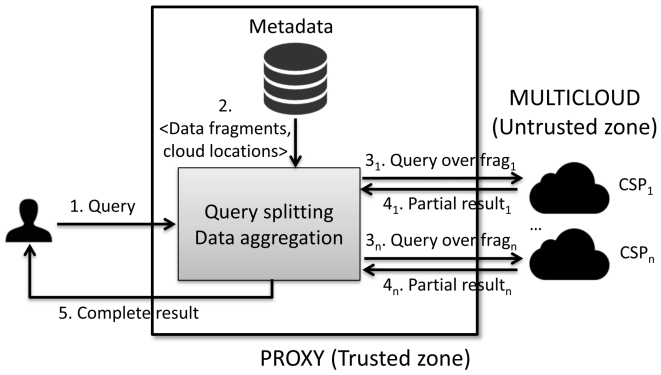

Fig. 4. Workflow of query answering on split data

The workflow of the execution of a user query over the outsourced split data is depicted in Figure 4. First, the query is processed and the storage locations of the data the query refers to are retrieved from the local database (Step 2). Then, the query is replicated as many times as the number of data chunks the data have been split into, and these queries are forwarded to the corresponding CSPs (Steps 3_1 to 3_n). As a result, each CSP provides a set of results that represents a partial view of the complete result (Steps 4_1 to 4_n). Finally, the proxy aggregates the partial results according to the criterion used to split the data, and provides the complete result to the user (Step 5).

Since fragments are stored in the clear, computations on individual fragments can be efficiently supported. Even so, computations involving data stored in different clouds will require specific algorithms by which the local proxy splits and orchestrates the computation among the CSPs in a privacy-preserving way (see Section 3.1.3). Also, the CSPs storing data fragments retain full utility on the fragment they store, which is a part of the utility of the whole data set.

Notice that, if several users employ the same local proxy and, therefore, they share the same pool of cloud storage locations, each location may store fragments of data of different users; that is, the split data of different users are *merged* at each cloud location. The fact that the proxy is the only one capable of interpreting which fragment corresponds to which user provides a degree of protection against the collusion of CSPs: even though CSPs may collude and pool the partial data they store, they will not be able to unequivocally ascertain the user to whom each fragment belongs. However, this also adds a single point of failure because, without the metadata stored at the proxy, data in the cloud cannot be accessed and fragments cannot be reconstructed. Therefore, redundancy of the proxy metadata and backup policies are crucial to ensure the robustness of the mechanism.

3.1.2 Data splitting mechanisms

Several mechanisms for privacy-preserving data splitting have been proposed in the literature. In [19], the authors split sensitive files into bytes, which are shifted and recombined into a fixed set of parts that are finally stored in different clouds. In [20], each file to be outsourced is associated a privacy level, that is chosen by the user according to the sensitivity of its contents. Then, file fragments are created based on the standard RAID storage mechanism, and those with higher privacy levels are stored at locations the user trusts more. In [21], byte-level data splitting is combined with data replication in order to improve both security and performance. The authors also propose an algorithm to store fragments in a multi-cloud so that they are at a certain distance; this minimizes the chance of confabulation attacks that may aggregate the split fragments. In [22,23] byte-level splitting of user files is combined with the encryption of the split fragments in order to improve security at the cost of efficiency. These mechanisms, in which data partition is performed according to a fixed criterion and bytes are alternatively picked and recombined, are well-suited for binary or even multimedia files, whose contents are usually stored but not processed by the cloud. However, they are not suitable for files whose contents should be processed by the cloud (e.g., structured databases or even textual documents such as e-mails or Word files). On the one hand, it is not possible to ensure that chunks of bytes do not contain enough data to cause disclosure if they are too large; on the other hand, it is not possible to preserve outsourced functionalities if chunks are too small or arbitrarily combined.

At an attribute level, most data splitting approaches focus on structured databases. In this scenario, data splitting can be *horizontal* (each fragment contains the values of all the attributes for a subset of records), *vertical* (each fragment contains the values of a subset of attributes for all the records) or *mixed* (each fragment contains the values of a subset of attributes for a subset of records). As stated in [18], horizontal partitioning is of limited use in enabling privacy-preserving decomposition because disclosure often arises from the correlation between attributes rather than from different records (records are usually independent).

On the other hand, vertical splitting is proposed in [17,18,24,25] to achieve confidentiality at the record level. Vertical splitting methods rely on predefined constraints describing risky attribute combinations. The goal is to partition the original data set into two vertical fragments in such a way that, if some risky attributes are stored together in a fragment, some of them need to be encoded/encrypted. In [24] the authors illustrate a similar approach to [18] and [25] but using an arbitrary number of non-linkable data fragments, which can be stored at an arbitrary number of providers. Also, [26] presents a solution for splitting data into two vertical fragments without requiring the use of encryption, but rather resorting to a trusted party (the owner) to store a

portion of the data and perform part of the computation.

Regarding mixed splitting, in general, it does not improve on vertical splitting in terms of privacy and it complicates distributed computation, because no local computation on entire attributes is possible any more at the locations holding fragments [16].

In summary, vertical splitting is the most suitable splitting for data protection, since in statistical databases it is the joint distribution of several attributes that may lead to re-identification of the subject behind a tuple of values. Moreover, when storing dynamically changing sensitive data in the cloud, vertical splitting is very convenient: each time one wants to add/update a record, this can be done separately in each fragment, which preserves the isolation inherent to splitting. Also, additions/updates are fast, because nothing needs to be done to the rest of records (those that do not change).

The methods above rely on the structure of the data and on predefined privacy rules (i.e., attribute combinations that are quasi-identifying) to guide the splitting process. A recent method [13] proposes a semantically-grounded splitting mechanism that is well suited for unstructured textual data, such as documents or emails. Its main innovation is to automatically detect (and split) the sets of textual entities that may disclose sensitive information by assessing the semantics they convey and their semantic dependencies. The method exploits the state of the art in document sanitization or redaction [27], which strives to automatically identify and protect sensitive linguistic entities in textual documents.

3.1.3 Outsourcing computation on split data

In vertical splitting, computations that involve single attributes (e.g., univalued statistics like the mean or the variance) or attributes stored within a single fragment are fast and straightforward: the cloud storing the fragment can independently compute and send the output of the analysis to the local proxy. However, analyses assessing the relationship (e.g., correlation) between attributes may involve fragments stored in different locations, and thus, communication between several clouds. As shown in [16], performing calculations on data split among multiple clouds can be decomposed into several secure scalar products to be conducted between pairs of clouds. Scalar products can be made secure with or without cryptography. Cryptographic approaches use a variety of techniques; for instance, the protocol in [28] involves homomorphic encryption. Non-cryptographic approaches are rather based on modifying the data before sharing them, in such a way that the original data cannot be inferred from the shared data but the final results are preserved [29].

Table 1. Summary of privacy-preserving data splitting methods.

Splitting criterion	*References*	*Supported operations*	*Usability*
Byte-level	[19,20,21,22,23]	Storage and retrieval	Straightforward to implement. Useful for binary files (e.g., images). No guarantees on data protection. No outsourced operations are supported
Attribute-level	[17,18,24,25,16]	Storage. Retrieval. Search queries. Computations	Useful for structured data bases. Privacy rules on attribute combinations should be carefully set. Operations on individual attributes are seamlessly supported. Multivariate analyses require ad hoc calculation protocols and introduce communication overhead
Semantic-level	[13]	Storage. Retrieval. Keyword search	Useful for textual documents. Flexible keyword search supported. Non-supervised splitting is not 100% accurate. Requires a large pool of independent cloud storage locations

In contrast with data splitting (and also data encryption), data anonymization *irreversibly* masks data in a privacy-preserving way. Anonymization techniques have been developed within the areas of Statistical Disclosure Control [30,31] and Privacy Preserving Data Publishing [32] as a means of releasing sensitive data to an untrusted third party so that they stay analytically useful for secondary uses but do not disclose information that can be linked to specific individuals.

In our setting, the untrusted party is the CSP and the secondary uses are the outsourced computations the users (e.g. data analysts) want to carry out on the masked data [33]. Whereas encrypted data are useless to whoever cannot decrypt them, anonymized data retain some utility for everyone. In particular, CSPs that offer their services free of charge in return for monetizing the analysis of the information they store may object against encrypted data (unless encryption is performed by themselves) but not against anonymized data (in most cases, their tools for automated analysis will not even be able to detect that anonymization has taken place).

Yet the main advantage of anonymized data over encrypted data and data splitting relates to the former's ease of processing. With data anonymization, masking is only performed at the storage stage, and can be implemented with linear or quasi-linear algorithms; after that, any query on the anonymized data (search, retrieval, calculation) is transparent and does not incur any overhead either for the proxy or the CSP. Due to the irreversibility of data anonymization, the retrieved data and the calculation results do not have to be unmasked (in fact they cannot), which means that the local proxy is only required at the storage stage, and no key materials need to be stored or managed.

A downside of anonymized data is that in general the results of computations on them are approximate rather than exact. A second downside is that keyword searches on attributes that have been anonymized are not supported. In fact, since data anonymization modifies individual values but tries to retain the utility of the data set as a whole, it is suited for aggregate calculations (e.g., statistics) on outsourced data sets, but not for queries or analyses on specific individuals. There would be little privacy if inferences on specific individuals were easy to make from the anonymized data. In contrast, some aggregate statistics are exactly preserved by certain anonymization methods, such as the mean [34] or the variance [35].

Anonymization can be performed independently for each record or by groups of records. In the first case, updates on anonymized data can be straightforwardly supported, whereas, in the latter case, updates would require re-anonymizing

the affected group of records or even the entire data set. Although anonymization can be applied to all the attributes of the data set to be outsourced, in most situations anonymizing the subset of attributes that can actually originate a disclosure is enough. According to their disclosure potential, attributes can be classified as:

- *Identifiers*. They individually disclose the identity of a subject (e.g., social security numbers). These attributes must either be removed from the anonymized data set or be replaced with random values.
- *Quasi-identifier or key attributes*. Unlike identifiers, these attributes do not identify subjects when considered separately, but their combination may. For example, neither age, gender, job or zipcode separately identify a single individual, but there may be a single, easy-to-spot individual for a certain combination of age, gender, job and zipcode (say a 95-year old female doctor living a bad neighbourhood). Quasi-identifier attributes should therefore be anonymized.
- *Confidential attributes*. They are those that convey sensitive features of an individual (income, religion, health condition, etc.). As long as they cannot be associated with an identity, confidential attributes can be outsourced unaltered, which is beneficial for analytical utility. However, in some cases they are also anonymized (in addition to quasi-identifiers) in order to achieve stronger protection.
- *Non-confidential attributes*. Attributes that do not belong to any of the previous categories can be outsourced unaltered.

There are two kinds of disclosure that can occur when outsourcing anonymized data. On the one hand, there is *identity disclosure* when observation of the anonymized data allows an attacker to deduce the identity of the subject to whom a record refers. As a result, the attacker can link the (typically unaltered) confidential attribute values in the record to the subject's identity, which clearly violates the subject's privacy. On the other hand, there is *attribute disclosure* when the attacker can determine a subject's confidential attribute value more accurately after seeing the anonymized data than before seeing them. For example, if the attacker cannot exactly determine the target subject's record but all records whose quasi-identifiers match the attacker's background knowledge on the target subject have high values for the confidential attribute "Income", the attacker can conclude that the target subject is wealthy. This shows that attribute disclosure can occur without identity disclosure. Conversely, identity disclosure can occur without attribute disclosure if confidential attributes in the target subject's record are missing or heavily masked.

In the sequel we survey the masking methods that can be employed to anonymize data and discuss their benefits and drawbacks. We have classified them according to the principle they employ to mask the data: Section 3.2.1 deals with

non-perturbative methods and Section 3.2.2 with perturbative methods. After that, in Section 3.2.3 we describe the privacy models that can be enforced by means of data masking in order to obtain formal anonymization guarantees. Table 2 summarizes the masking methods discussed and the privacy models they can achieve.

3.2.1 Non-perturbative masking

Non-perturbative masking does not alter the truthfulness of the data, although it reduces their accuracy. The masked data are derived from the original data through partial suppressions or reductions of detail, but they stay truthful [36].

The main methods that can be used to outsource data via non-perturbative masking are:

- *Sampling.* The masked data are a sample of the original data, which is construed as the population. Thus, if an intruder identifies a unique record in the released data (sample), he cannot be sure it was unique in the original data (population), which thwarts re-identification. The smaller the sampling fraction, the more protection [37], although the probability of releasing population uniques is never zero.
- *Local suppression.* If a combination of quasi-identifier attribute values is rare (that is, shared by too few records), this may lead to re-identification (identity disclosure). Local suppression is a countermeasure that operates by suppressing certain attribute values (i.e. replacing them with missing values) in order to increase the number of records sharing the combination, which reduces the risk of identity disclosure [38].
- *Generalization (a.k.a. global recoding).* This is an alternative to local suppression to handle rare combinations: values of the quasi-identifier attributes are recoded into new (more general) categories so as to reduce detail and thereby make combinations less rare and re-identification more difficult [38]. For categorical attributes (e.g., job), generalizations are obtained from value generalization hierarchies, taxonomies or ontologies [39]. For numerical attributes (e.g., age), generalizations correspond to increasingly larger intervals (e.g., $25 \rightarrow [20-30] \rightarrow [0-50]$).

3.2.2 Perturbative masking

Perturbative masking consists in generating a modified version of the data set such that the masked values can be viewed as original values plus some noise. Hence, the individual masked values are not truthful in general. However, perturbative masking may preserve the statistical properties of the original data better than non-perturbative masking (see below).

The best-known perturbative masking methods are:

- *Noise addition.* Each record in the original data set is added a noise vector. For example, if the noise vector is drawn from a $N(0, \alpha\Sigma)$ distribution, where Σ is the variance-covariance matrix of the data set and α is a parameter that defines the amount of noise to be added, the anonymized data can be expected to preserve the original means and correlations [40]. Even though noise addition was in principle only suitable for continuous attributes, it has been recently adapted to handle categorical attributes [41]. It can be applied to both quasi-identifiers and confidential attributes, in order to avoid identity and attribute disclosure at the same time. Since noise is added to individual records independently, noise addition has linear complexity in the number of records, which makes it an efficient option for large amounts of data.
- *Data swapping.* It randomly exchanges the values of attributes among different records [35]. In this way, univariate distributions are exactly preserved. In the case of numerical attributes, value swaps are restricted within a certain range so that the variance-covariance matrix is not altered too much. For categorical attributes that can be ranked, only swaps between categories whose ranks are not too different are made (this variant is known as rank swapping). Extensions of swapping for nominal categorical attributes that have no natural ranking can be found in [42,43]. As noted by [44], data swapping has a high level of user acceptance as the values themselves do not suffer any modification. It can be applied to quasi-identifiers and/or confidential attributes; in any case, the link between identities and confidential values is altered, which prevents attribute disclosure.
- *Microaggregation.* It clusters records into groups containing each at least k similar records and releases the average record value of each group [45]. The more similar the records in a group, the more data utility is preserved. Furthermore, the clustering can be univariate [46] or multivariate [34], and there exist numerical [45] and categorical versions [47]. Microaggregation is normally used on quasi-identifiers to prevent re-identifications, but it has recently been extended so that attribute disclosure is also minimized during the clustering of records [48].

Although perturbative methods may produce untruthful values at a record level, they have several advantages over the non-perturbative methods discussed above regarding data utility preservation. Such advantages make them preferable if we are mainly interested in outsourcing aggregate computations on large data sets to the cloud. Specifically, due to the removal of values and records, sampling and local suppression and sampling usually result in a large loss of data utility [30]. Also, generalization may recode some records that do not need it, thus causing extra utility loss; and generalization usually results in a significant loss of data granularity because values become less specific. In contrast, perturbative methods do not reduce the granularity of the values;

in the particular case of data swapping, all the original attribute values appear in the anonymized data set. For continuous attributes, generalization discretizes input numbers to numerical ranges and thereby changes the nature of data from continuous to discrete. In contrast, perturbative methods maintain the continuous nature of numbers. Finally, another advantage of perturbative methods is that they are able to perfectly preserve some statistical features of the data, which are of the utmost importance in data analysis. Specifically, if the parameters of noise are suitably chosen, noise addition can preserve attribute means and correlations. Also, data swapping preserves all univariate statistics (attribute means, variances, frequency distributions and min-max values). On the other hand, microaggregation preserves attribute means.

3.2.3 Privacy models

Privacy models address the trade-off between privacy and utility by putting privacy first and stating an *ex ante* privacy guarantee that the anonymized data set must satisfy, regardless of the original data set and the method used to mask it [36]. Normally, there are several possible masking methods that can satisfy a certain privacy model. The specific masking choice must be made to maximize data utility (because satisfying the model already ensures privacy). By adhering to a privacy model, users (for example, the security managers in charge of the local proxy) may obtain a clearer idea on the type and level of protection achieved for the data outsourced to the cloud, no matter their size, type or structure.

k-Anonymity [38] is the oldest among the so-called syntactic privacy models. It seeks to make record re-identification unfeasible by hiding each subject within a group of k subjects. To this end, k-anonymity requires each record in the anonymized data set to be indistinguishable from another $k-1$ records as far as the quasi-identifier attributes are concerned. Therefore, in a k-anonymous data set, no subject's identity can be linked (based on the quasi-identifiers) to less than k records; that is, the probability of correct re-identification is, at most, $1/k$.

Note that for $k > 1$, this probability is less than 1 for all records, thereby ensuring zero unequivocal re-identifications [49]. Moreover, by tuning k, one can also tune the level of exposure of each individual.

A method enforcing k-anonymity should modify the quasi-identifier values of records to attain the indistinguishability condition of the model. Masking methods typically used to enforce k-anonymity are:

- *Local supression*, which replaces the rarest attribute values by missing values so that the number of records sharing a combination of quasi-identifier attribute values increases;

- *Generalization*, which groups similar records in clusters of k or more and replaces their quasi-identifier attribute values by a generalization common to all cluster records;
- *Microaggregation*, which groups similar records in clusters of k or more and replaces their quasi-identifier attribute values by their cluster-level centroid/average.

In the latter two methods, records are grouped according to the similarity of their quasi-identifier attribute values, in order to minimize the information (and utility) loss incurred when replacing those quasi-identifier values by common values.

Although k-anonymity protects against identity disclosure (the subject to whom a record corresponds can be successfully re-identified with probability at most $1/k$), attribute disclosure can happen if the variability of the confidential attribute values (e.g., diagnosis) in a cluster of k records is small (e.g., all individuals in the cluster have *cancer*). Refinements of the k-anonymity model protecting attribute disclosure are l-diversity [50] and t-closeness [51]. l-Diversity requires that the confidential attribute values within each cluster of (at least k) indistinguishable records are diverse enough to avoid unequivocal disclosure of confidential values. t-Closeness reinforces this notion by requiring that the distribution of the confidential attribute values of each cluster be similar to the distribution of the confidential attribute values in the entire data set. This privacy guarantee is probably the strictest among k-anonymity-like models. t-Closeness has been enforced by means of *generalization* [52] and *microaggregation* [48]. In both cases, the grouping of records is done so that i) the distribution of confidential values in the cluster is diverse enough to fulfill t-closeness, and ii) the similarity among the records in a cluster fulfilling t-closeness is maximal with respect to their quasi-identifier attributes, so that information loss is minimized.

Differential privacy [53] is another privacy model that has become very popular in recent years due to its strong privacy guarantees. In a nutshell, differential privacy requires that the presence or absence of any individual in a data set does not significantly influence the results of analyses on the data set. Differential privacy is usually achieved by adding a special type of noise to the attribute values or the value counts [54]. However, since this privacy model has been conceived for interactive settings, where the anonymization mechanism sits between the users performing queries and the data controller storing the original data set and answering the queries, it does not fit well in the cloud scenario, because one cannot outsource storage and computation on the original data to untrusted third parties (the CSPs). To increase its flexibility, some authors have studied the application of differential privacy to (non-interactive) data releases: state-of-the-art proposals are based on histograms [55], microaggregation [56], Bayesian networks [57] or synthetic data [58,59]. However, the

strict privacy guarantees of differential privacy usually entail severely perturbing the data [54], hence diminishing the benefits of outsourcing differentially private data to the cloud and in general of releasing differentially private microdata [60] (that is, data sets where each record corresponds to one subject). Current research on improving the utility of differentially private data releases focuses on designing mechanisms that reduce the amount of distortion to be added to the data [61] or on proposing reasonable relaxations of its privacy guarantees [62].

3.3 Cryptographic techniques

In this section, we survey the state-of-the-art of cryptographic techniques oriented to cloud computing. Indeed, cryptography can be viewed as one of the strongest and most critical building blocks in many of the security plans deployed to address privacy concerns in the cloud.

Modern cryptography has evolved by addressing very diverse real-world security needs, many of which apply to cloud computing. As a field, cryptography considers a wide range of information security objectives, user architectures and functionalities. Real-world cloud computing scenarios may require various security objectives, such as data confidentiality or data integrity. Furthermore, one may wish to carry out functionalities richer than simple secure storage in the cloud, such as searching and retrieving data from the cloud, or computing on the protected data.

Due to the data breach outbreak witnessed in the last few years, many cloud computing services nowadays use cryptographic solutions to encrypt the data they store, particularly by using symmetric encryption schemes such as AES-256 [63]. However, in those services the CSP often manages all the keying material, and thus the privacy of the outsourced data ultimately rests on the trust relationship between users and the CSP.

A natural cryptographic approach to help preserve data privacy against untrusted CSPs is to encrypt the data using traditional encryption schemes prior to outsourcing them. Nevertheless, this approach prevents CSPs from executing any complex functionality, and forces users to download the entire data set if they want to exploit the outsourced data. Hence, traditional encryption schemes fail to support the delegation of most functionalities over the outsourced encrypted data to the cloud, and this clearly degrades the performance and cost-saving benefits of cloud architectures. To address this issue, recent cryptographic research focuses on developing encryption techniques that empower the client to securely delegate to the cloud particular functionalities on outsourced encrypted data.

Table 2. Summary of data anonymization methods

Method	*References*	*Perturbative*	*Usability*	*Achievable privacy models*
Sampling	[37]	N	True values released. Large utility loss. Population uniques may remain. Efficient	
Local suppression	[30]	N	True values released. Yields missing values. Large utility loss. Needs combination with generalization	k-anonymity and extensions (combined with generalization)
Generalization (global recoding)	[30,38]	N	True values released. Granularity and utility loss. Needs generalization hierarchy. Discretizes numerical data	k-Anonymity. l-Diversity. t-Closeness
Noise addition	[40]	Y	Efficient. Mostly for numerical attributes. Can preserve statistical features	Differential privacy
Data swapping	[35]	Y	Preserves univariate distributions. Applicable to any type of attributes	
Microaggregation	[45,34]	Y	Preserves attribute means. Moderate damage to covariances. Quadratic complexity. Usable on numerical and categorical attributes	k-Anonymity. l-Diversity. t-Closeness. Can help reduce utility loss of differential privacy

In the following, we give an overview of the main cryptographic approaches to address the data security and privacy issues raised in cloud computing. We look at techniques allowing clients to outsource an encrypted data set to the cloud, in such a way that the cloud can still perform useful functionalities on the outsourced data set. The surveyed solutions are *user-centered*, in the sense that they protect information in the user's trusted domain before outsourcing it. We also require them to provide *end-to-end* security, meaning that the data stay in a protected form whenever they leave the client trusted zone. Finally, the surveyed techniques efficiently support functionalities arising in cloud computing. We must note that, while all the surveyed techniques are very promising, many of them are in their early stages and thus still largely undeployed.

We focus on techniques aimed at delegating four main functionalities needed in cloud computing that are hard to outsource with traditional encryption schemes:

- *Searching on encrypted data*, which refers to retrieving a segment of records of an outsourced encrypted data set that satisfies certain query conditions. Since remote search capabilities take a central role in the business logic of many commercial and industrial applications, solutions to securely search over outsourced data can bring huge security benefits.
- *Computing on outsourced data*, which is a central functionality in cloud computing. Due to the major performance advantage and the data ubiquity and availability features provided by cloud infrastructures, performing computations in the cloud is very convenient in a wide range of applications. Relevant examples include e-voting, accounting and financial applications as well as computing statistical indicators.
- *Access control on encrypted data*, which refers to restricting the visibility of the data to a designated set of authorized recipients. Controlling access to data is a required functionality in many cloud computing applications involving access policies, such as file sharing or audit log systems. However, entrusting the cloud with the access control functionality is prone to many privacy threats. Therefore, we survey existing techniques to achieve user-centered access control on the outsourced data by cryptographic means.
- *Storage control on outsourced data*, which refers to the user being able to check the integrity and the location of her outsourced data.

3.3.1 Searching on encrypted data

Retrieving outsourced data selectively is a basic functionality of cloud storage servers, and it is required in a wide range of applications such as file storage systems, outsourced backup services, accounting systems or e-mail servers. However, delegating the search execution usually requires entrusting cloud

service providers with the data set and with the query data, both of which may contain sensitive information. If users want to remotely query their data, this compels them to completely trust cloud service providers. Therefore, recent cryptographic research has sought techniques to protect the outsourced data, while also enabling users to delegate the search execution to the CSP. We survey here some of the existing cryptographic constructions aimed at providing users with remote search capabilities.

We can classify potentially sensitive information in the context of searching over encrypted data into *outsourced data*, *query data* and *search information*. *Outsourced data* refer to information enclosed in the data set owned by the user, such as the data content or the number of data items. In case users delegate data of sensitive nature, data items may indeed need to be protected. *Query data* refer to data enclosed in queries. Note that queries themselves can be of a sensitive nature, for instance, if they contain personal information details. Finally, *search information* consists of all the data generated in the searching process, as can be the number of data items that match any given query.

The primary cryptographic tools designed to provide secure delegated search are Searchable Encryption, Oblivious RAM and Private Information Retrieval. Other techniques such as Public-Key Encryption with Equality Test and Order-Preserving Encryption are also useful in some applications involving secure search, although they are not as relevant to the searching case. All these solutions differ both in their security guarantees and in the supported queries. Since Searchable Encryption is arguably the most practical technique among all the cryptographic tools aimed at providing secure delegated search, we dedicate most of this subsection to this technique.

Searchable Encryption (SE) schemes [64] protect the data prior to outsourcing by encrypting them in a way to enable secure search over the outsourced data. In order to increase the efficiency of the searching process with respect to alternative constructions, SE schemes typically lower their security guarantees to allow a bounded amount of information leakage. For an extensive survey on the subject of SE, see [65].

Oblivious Random Access Machines (ORAMs) [66] were originally formulated in the context of protecting memory access patterns in program executions. However, they are also applicable to secure delegated search. Here, ORAM achieve the highest available security features and protect most outsourced data and all query and search information. Moreover, ORAM solutions support dynamic data sets. However, most ORAM instances require a great amount of interactivity and client computations, and this restricts their practical applicability in many situations.

Private Information Retrieval (PIR) [67] aims at protecting the knowledge of which data elements are retrieved from the CSP at every query. Therefore, PIR schemes allow retrieving data items from a (possibly unprotected) outsourced data set, while the CSP is oblivious to which items are retrieved. In this sense, PIR is related to other cryptographic primitives, particularly to oblivious transfer, which satisfies a stronger security definition. Despite recent improvements in PIR constructions [68], their practical use is restricted because of their significant costs.

Order Preserving Encryption (OPE) schemes [69] allow encrypting numerical data in a way that makes it possible to compare the encrypted data, and thus to order the ciphertexts according to their underlying plaintext values. OPE schemes provide a very efficient solution for one-dimensional range queries, as well as maximum, minimum and top queries, and they have found applications in existing cryptographic suites, such as CryptDB [70]. However, the security notion of OPE schemes is weak, and recent studies show that the leakage they impose can be significant [71].

Generally, the searching process involves three distinct types of actors: writers, readers and the CSP. Writers are cloud users in charge of outsourcing the data (i.e., data providers), and readers hold the ability to remotely search over the outsourced data (i.e., data consumers). The architectural setting can be single or multi-reader and single or multi-writer depending on the number of actors taking part in the process, and it can also be single or multi-cloud according to the number of CSPs involved.

One of the most frequent architectures found in applications involving delegated search is the single-user, single-cloud setting, where a single client uploads a data set to a cloud server and then wants to query on it. Searchable Encryption schemes in this setting are called Searchable Symmetric Encryption (SSE) schemes. A relevant foundational work in the SSE literature is the paper by Curtmola et al. [72].Since their foundational work, intensive research efforts have extended the field in many directions, by enhancing the security and the efficiency of the schemes [73], improving the expressiveness of the queries [74], providing verifiability of the search results [75], supporting dynamic data sets [76], and supporting various other functionalities [65].

In [73], Cash et al. test the performance of their SSE scheme over a multi-million, terabyte-scale webpage collection that includes Wikipedia as a subset. Their performance analysis shows that queries in SSE with 10,000 results can be executed in under one second, and that the search execution time is roughly logarithmic in the number of data items of the outsourced data set.

We also note here that Searchable Encryption has been explored in other architectural settings, and out of them the single-reader, multi-writer case

has received most attention. In this setting, Searchable Encryption receives the name Public-Key Searchable Encryption with Keyword Search (PEKS). The first PEKS scheme was proposed by Boneh, Di Crescenzo, Ostrovsky and Persiano in [77], and after their work PEKS has been extended with sublinear search [78], or expressiveness [79,80] and security [81] enhancements (see [65] for a survey about PEKS). However, SSE remains the most efficient and popular Searchable Encryption flavor.

In a client-server setting, SSE schemes are used as depicted in Figure 5. First, as in all symmetric encryption schemes (e.g. [63]), a set-up algorithm is executed locally to provide the client with a key. Then, in an Outsource Phase (step 1 in Figure 5), the client uses this key to generate an encrypted version of the original data set, consisting of a collection of ciphertexts and an encrypted index, and outsources it to the cloud. Later, in the Query Phase (step 2), the client may want to retrieve the segment of the outsourced data set that satisfies a particular query. To do so, the client uses the SSE scheme to generate an encrypted version of this query, commonly called a trapdoor, and then sends it to the server. Once the trapdoor has been issued, the cloud can enter the Computation Phase (step 3) and combine this trapdoor with the encrypted index, so as to filter the items in the encrypted data set that satisfy the query condition. When these encrypted results are sent back to the client, she executes the Decryption Phase (step 4) and finally obtains the intended query outcome.

Searchable symmetric encryption schemes can be classified according to query expressiveness, i.e., according to the class of supported queries. If the SSE scheme supports only single keywords as queries, the scheme is called *single-keyword*. Alternatively, if the scheme supports arbitrary conjunctions of single-keyword queries, it is called *conjunctive* [73]. If the scheme supports arbitrary Boolean formulas on keywords, it is called a *Boolean* scheme. Recent research in the field of SE has expanded schemes to support several other predicates, such as range [82] and substring [83] queries, among many others.

Typically, SSE schemes leak the number of outsourced data items, the *search pattern* and the *access pattern*. The search pattern is the query data of whether any two given queries are identical or not, and the access pattern corresponds to the search information of which outsourced data items match any given query. By learning this information, the CSP is able to carry out the search process very efficiently, often in sublinear time in the size of the data set.

It has recently been shown that encryption schemes that reveal the volume of retrieved data, search patterns or access patterns may be insecure. Kellaris et al. [84] presented generic reconstruction attacks on any system supporting range queries where either the access pattern or the communication volume is leaked. In particular, these attacks affect searchable encryption schemes

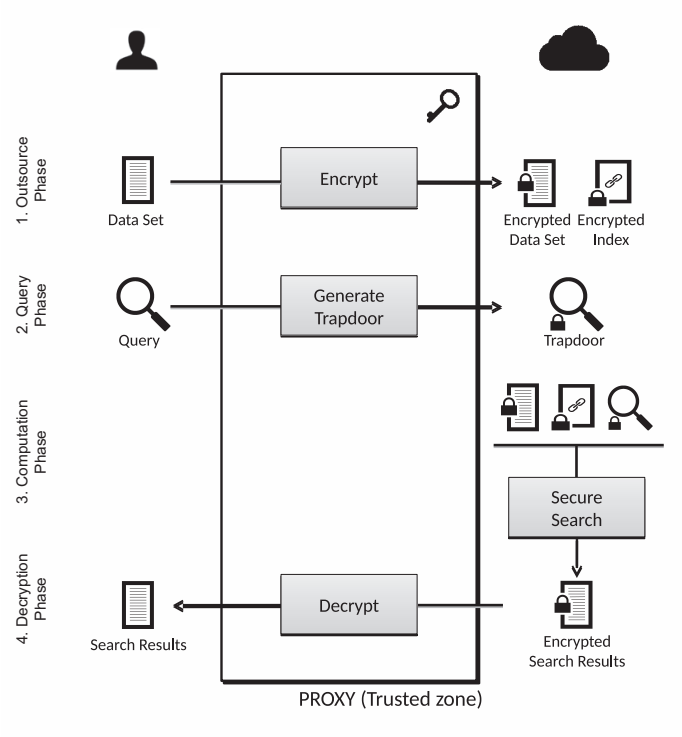

Fig. 5. Workflow of the delegated encrypted search process using Searchable Symmetric Encryption schemes

supporting range queries. Subsequent works improved the aforementioned attacks under different assumptions [85,86]. Also, generic single-keyword and conjunctive searchable encryption schemes were found vulnerable against file-injection attacks [87]. These vulnerabilities have motivated the construction of new searchable encryption schemes that adopt additional security measures [88,89,90].

3.3.2 Computing on encrypted data

Several cloud computing applications involve delegating computing capabilities on outsourced data to a cloud service provider. Frequent examples of this arise in the areas of finance, enterprise resource planning, decision support systems, computation of statistical indicators, e-voting systems or machine learning.

As in the setting of searching over outsourced data, the architectural framework in which computing over outsourced data takes place involves three distinct types of actors: the cloud service provider, the writers (or data providers) and the readers. Here, the writers provide the data to be outsourced, and the readers specify the computation to be outsourced and then receive the computation outcome.

Outsourcing data and computations in the clear to an untrusted cloud service provider poses a threat to data confidentiality. In this context, we can classify potentially sensitive information into *outsourced data*, *outsourced computation* and *computation outcome*. The outsourced data refer to information enclosed in the data set owned by the user. The outsourced computation refers to information about the particular operations to be carried out: for example, whether the outsourced computational task is to compute the mean of the outsourced data set, or to perform a linear regression. The computation outcome refers to information about the computation results, which can reveal information both about the outsourced data and the outsourced computation.

Among cryptographic approaches, there exist two main techniques aimed at tackling data confidentiality issues in outsourced computation: homomorphic encryption and secure multi-party computation.

3.3.2.1 Homomorphic Encryption. These are cryptosystems that enable computing on encrypted data. Homomorphic encryption (HE) comes under single-reader architectures, and it allows users to encrypt their data so that operations on the generated ciphertexts can be carried out in such a way that they translate to arithmetic operations on the underlying plaintexts.

In a client-server setting, HE schemes are used as depicted in Figure 6. First, a set-up algorithm is executed locally to provide the client with a key. Then, in an Outsource Phase (step 1 in Figure 6), the client is able to outsource her data in an encrypted form to a non-trusted party. Subsequently, in the Query Phase (step 2), the same user can request arithmetic circuits to be evaluated on those encrypted data. Upon receiving this query, the cloud server enters a Computation Phase (step 3) and securely computes an encrypted result. Finally, in the Decryption Phase (step 4) the client receives this encrypted result and decrypts it. This usage of HE schemes is non-interactive, and it protects both the outsourced data contents and the computation outcome from the cloud service provider.

In the literature, homomorphic encryption schemes are classified in three categories according to their homomorphic properties:

- *Partially Homomorphic Encryption* (PHE) schemes support just a single arithmetic operation on ciphertexts. If the operation on ciphertexts yields the encrypted version of the sum of the corresponding plaintexts, then the homomorphic scheme is called *additive*; if it yields the encrypted version of the product of the plaintexts, then the homomorphic scheme is called *multiplicative*. Common examples of multiplicative homomorphic encryption schemes are the RSA [91] and the ElGamal [92] encryption schemes; on the other hand, the Paillier [93] cryptosystem is additively homomorphic.

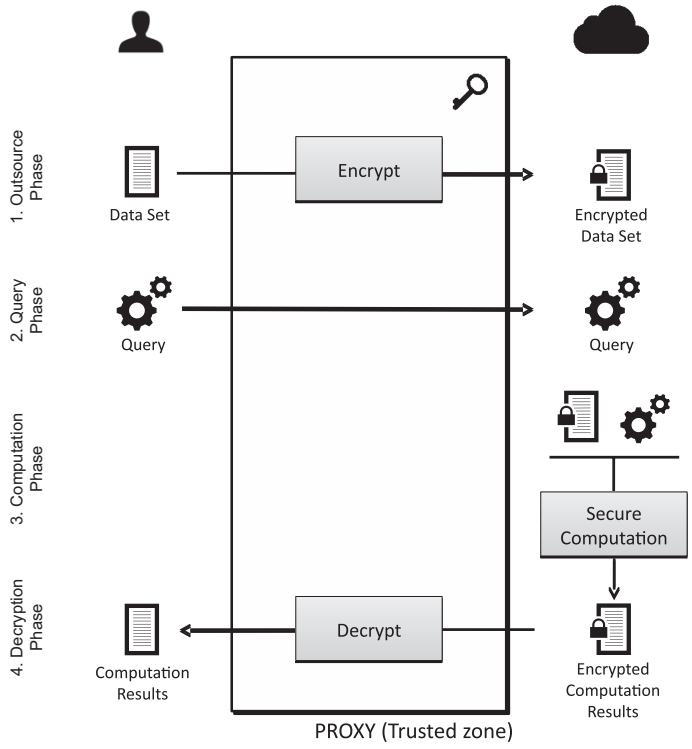

Fig. 6. Workflow of the delegated computing process using homomorphic encryption schemes

- *Somewhat Homomorphic Encryption* (SHE) schemes support both addition and multiplication of ciphertexts, but in practice they allow a limited number of operations. Most SHE schemes admit a large number of additions and a small number of products on ciphertexts. This limits the computations that can actually be outsourced to the cloud, and so it restricts the suitability of SHE schemes for certain applications.

  Some of SHE schemes examples are the BGN scheme [94], which admits many additions and a single multiplication, and the Catalano-Fiore schemes [95,96], whose ciphertext size and computation and decryption times increase linearly with the multiplicative depth, and which can compute arbitrary quadratic multivariate polynomials on the input data.
- *Fully Homomorphic Encryption* (FHE) schemes support an unlimited number of both additions and multiplications on ciphertexts. By encrypting information in a bit-by-bit fashion, we note that addition and multiplication can be interpreted as XOR and AND gates [97], which are complete for the class of Boolean circuits. Hence, FHE allows the computation of any Boolean circuit on the encrypted inputs.

  Unfortunately, the computational overhead of FHE schemes is currently too large to make them practical for many applications. Most existing FHE schemes follow the blueprint by Gentry [97] for FHE, which consists in using ideal lattice-based SHE schemes with an additional bootstrapping algorithm. When too many multiplications are performed, the expensive boot-

strapping algorithm is used to process the encrypted data so that additional multiplications are allowed. Other subsequent approaches to FHE include schemes over the integers based on the approximate GCD assumption [98] and schemes based on the learning with errors (LWE) and ring learning with errors (RLWE) assumptions [99]. Most notable examples of FHE schemes are the BGV [100], Fan-Vercauteren [101], Brakerski-Vaikuntanathan [99] and YASHE [102] schemes. For a recent survey on FHE, see [103].

In general, existing HE schemes show a trade-off between functionality and efficiency. While FHE schemes are very convenient from the functionality point of view, the solutions known up to date have a massive overhead in computational and memory costs when evaluating functions of high multiplicative depth. The current technological status and these efficiency issues still restrict the applicability of FHE in practice. However, recent improvements in schemes [101] and FHE for certain arithmetic functions [104] allow the use of this primitive in particular applications. Some examples of these applications are the secure evaluation of neural networks [105,106] and the secure retrieval of data from encrypted databases [107].

A widely adopted strategy to outsource computational tasks over encrypted data is to modify the protocol and the data set and then employ additional cryptographic primitives so that PHE suffices to securely delegate computations. This strategy can be applied to delegate computations such as Gaussian elimination [108], linear regression [109], interpolation [110], statistical analysis [111], e-voting [112], analysis of genomic data [113] or private information retrieval [114]. Other custom techniques can be employed to privately delegate various matrix operations, such as multiplication [115], factorization [116], inversion [117], solving systems of linear equations [118] and linear programming [119]. See [12] for a survey on the secure outsourcing of these and other functions.

Known homomorphic encryption schemes satisfy several security definitions. One of the most common is semantic security, which states that, given any two adversarially chosen messages and an encryption of one of them, an adversary is not able to figure out which message corresponds to the ciphertext. Semantic security ensures data confidentiality, in the sense that no adversary is able to recover any partial information about the data set from just the encrypted data. Additionally, the verifiability of the results returned by the cloud is a very important issue that has been studied in many works, like [120,121,122,123]. See [124] for a recent survey on verifiability in outsourced computation.

3.3.2.2 Secure Multi-Party Computation. This is an area of cryptography that deals with the case where two or more parties hold some private input data, and they want to jointly evaluate a function on their inputs with-

out revealing them to other parties. Recently, secure Multi-Party Computation (MPC) has also been used to build cryptographic solutions for cloud computing. In this new setting, the players are the data controller and the CSPs, who run an MPC protocol to compute the output of a function of the data they hold. The output of the function will only be received by the data controller.

From an efficiency perspective, the main difference of the approach to compute on encrypted data provided by HE and by MPC lies in the inherent *interactivity* of MPC protocols. That is, unlike HE schemes as described in the previous section, MPC protocols may require various rounds of communication between all parties until the data controller finally receives the computation result.

Yao [125] provided a 2PC protocol (i.e., an MPC protocol for two parties) for arbitrary functions, and Goldreich et al. [126] generalized it to the general MPC case (i.e., with an arbitrary number of parties). The 2PC protocol proposed by Yao provides efficient MPC with low interactivity, and experiments [127] show that it is possible to evaluate circuits with 1.2 billion gates in 18 minutes. Additionally, there exist very efficient tailor-made MPC protocols for specific operations, such as for e-voting [128,129], electronic auctions [130], data mining [131] and set operations [132,133], and MPC can be used to delegate computations on encrypted data from various users [134]. However, the general multi-party setting proposed in [126] requires many rounds of communication, and the required bandwidth blows up with the number of parties. Large interactivity, along with the hardness of coordinating interactions with a large number of parties, remains the main bottleneck for the deployment of general MPC protocols. For an introduction to MPC, see [135].

In the cloud computing setting, there are several options to use MPC protocols for secure computation. For example, a data controller can split her data among several CSPs using a secret sharing scheme to ensure that no single CSP has information about the data. If the controller needs to perform a computation on the outsourced data, she starts an MPC protocol with the CSPs to compute the desired function, and the output is just received by the data controller [136,137]. Also, secret sharing can be used to protect cryptographic keys, by storing shares of the keys in different CSPs [138]. In the seminal works on data splitting [18] the one-time-pad scheme, that can be viewed as a secret sharing scheme, was also proposed as an alternative to splitting when data require extra protection. In other cases in which the users have low computational power, secure two-party computation protocols are run between the data controller and a server using garbled circuits [139,140,141].

In general and regarding security guarantees, the only information that parties can learn in an MPC protocol is the one that can be deduced from their own input data, the function being computed and the result of the computation. Hence, MPC protects the individual inputs and discloses both the outsourced

computation and the computation outcome.

Adversaries are *static* if they corrupt players before the protocol begins; and *dynamic* if they corrupt players on-the-fly during the protocol execution. Schemes that are secure against static adversaries are significantly more efficient. Furthermore, the literature defines three different adversary models: (i) *semi-honest adversaries*, who follow the protocol honestly but try to learn additional information; (ii) *malicious adversaries*, who do not necessarily follow the protocol; (iii) *covert adversaries*, who do not necessarily follow the protocol, but do not wish to be caught deviating from it. Semi-honest adversaries are the most relevant in our context.

The literature considers additional security properties, such as *fairness* and *guaranteed output delivery*. The fairness property ensures that all the involved parties receive the outcome of the computation simultaneously. Guaranteed output delivery states that no corrupted party is able to prevent honest parties from receiving the correct computation output.

Private Function Evaluation (PFE) [142] is a special case of two-party MPC that allows users to hide the outsourced computation. In PFE, both parties hold private input data and one of them additionally holds a private function. Their goal is to learn the outcome of the joint function evaluation, while keeping their inputs private. Typically, PFE schemes leak the size of the private function as an arithmetic circuit.

3.3.3 Access control on encrypted data

Data sharing is a required feature in many cloud computing applications, for instance in audit log systems, file storage services and messaging services. As a consequence, access control on outsourced data is a common functionality in cloud computing, since data owners may want to control which data consumers are allowed to access any given data item. This may be the case, for instance, when a medical institution wants to restrict access to patient history to only the authorized personnel. In the typical cloud computing service, access control is enforced by the CSP in the cloud premises. However, when outsourcing sensitive data this strategy poses serious confidentiality threats.

The quest for cryptographic approaches to provide user-centered access control can be classified into four main areas:

- *Public-Key Encryption:* it realizes one-way encrypted communication to a target recipient;
- *Identity-Based Encryption:* it enables access control based on identities;
- *Attribute-Based Encryption:* it enables access control based on attributes;
- *Functional Encryption:* it lets private key holders learn some function of the

underlying plaintext.

In Public-Key Encryption (PKE) schemes [91], the receiver generates a pair of keys: a public key, which can be posted publicly, and a private key, which is kept secret. The security properties of PKE schemes guarantee that, given the knowledge of the public key, the sender is able to generate ciphertexts that are meant to be sent to the receiver, and further that it is unfeasible to extract partial information about the underlying plaintext from these ciphertexts without the knowledge of the secret key. Therefore, given that two parties know the public key of each other, PKE schemes enable both of them to communicate privately even if they do not share common secret information.

By choosing the public key corresponding to the intended recipient, PKE schemes provide targeted access control for single recipients. Hence, in practice, public keys are either distributed to many different senders privately, or publicly disseminated through a Public-Key Infrastructure (PKI). PKIs are used to avoid tampering attacks, by certifying that public keys correspond to the intended recipients. They use digital signatures and they rely on a number of trusted certification authorities for this verification. Thus, the introduction of PKIs entails an efficiency penalty, and it increases the attack surface of the scheme. These shortcomings motivated the introduction of Identity-Based Encryption schemes.

Identity-Based Encryption (IBE) schemes [143] rely on a central authority. They are used concurrently by a set of users, each of them being assigned a particular identity by the central authority. Identities in this setting are represented as strings, such as a name or an e-mail address. Similarly to PKE, IBE schemes use a pair of public and private keys for each identity. However, unlike in PKE, the receiver does not generate the keying material. Instead, private keys associated with each identity are created by the central authority, and they are then privately distributed to the corresponding users.

The main strength of IBE comes from the fact that anybody can generate the public key corresponding to any given identity by using the authority's public information, without having to interact with the receiver or with the central authority. Thus, IBE schemes effectively remove the need for PKIs in asymmetric encrypted communication, allowing users to encrypt under the public key of any recipient without having to request any keying material to the authority. The cryptography literature has extended IBE schemes to achieve strong security notions [144] and high efficiency [145], to deal with fuzzy [146] and hierarchical [147] identities, to include revocation mechanisms [148], or even to realize searchable encryption in the single-reader and multiple-writers setting [77]. For a survey of Identity-Based Encryption, see [149].

Just as PKE does, IBE schemes provide targeted access control for one-to-one

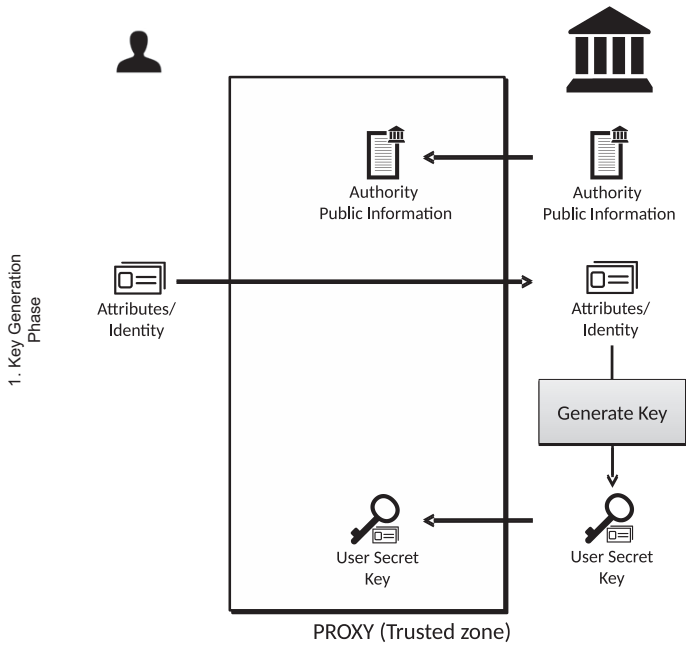

Fig. 7. Workflow of the Key Generation Phase of an access control mechanism using ABE or IBE schemes, in the context of cloud storage systems

communication. This process is depicted in Figures 7, 8 and 9, and we follow it through an example.

Imagine a healthcare institution that uses a cloud file sharing system for convenience. The institution consists of a central administration authority and a medical board, the latter being formed by several doctors. Further assume that each doctor is publicly identified by its ID card number. Upon hiring a doctor, the central administration can execute a Key Generation Phase (step 1 in Figure 7) and use an IBE scheme to give her a secret key corresponding to her ID card number.

Now, suppose that a general practitioner wants to send a patient over to a certain specialist, and hence she wants to share the patient's file with this specialist. To do so, in an Outsource Phase (step 2 in Figure 8), she can use IBE to encrypt the patient's file according to the master public information of the central authority and to the ID card number of the specialist, and then upload the encrypted file to the cloud server.

Finally, in a Decryption Phase (step 3 in Figure 9), the specialist can retrieve the encrypted file from the cloud server and, provided she is authorized to, decrypt it. Assuming that the central authority is trusted, this use of IBE protects the patient's file, and it ensures that the encrypted file stored in the cloud can only be decrypted by the intended recipient.

In the context of applications requiring one-to-many communication, one often resorts to Attribute-Based Access Control (ABAC) in order to facilitate

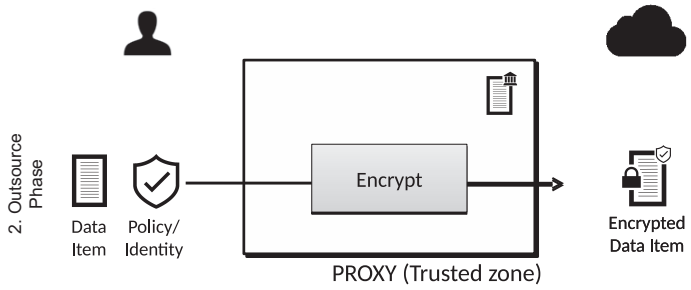

Fig. 8. Workflow of the Outsource Phase of an access control mechanism using ABE or IBE schemes, in the context of cloud storage systems

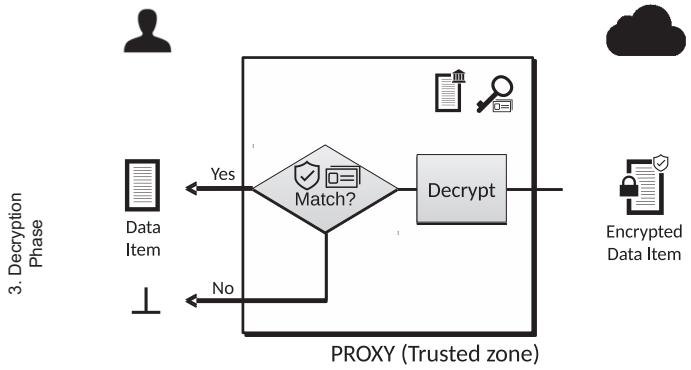

Fig. 9. Workflow of the Decryption Phase of an access control mechanism using ABE or IBE schemes, in the context of cloud storage systems

the administration of access control and increase efficiency. ABAC is an approach in which identities are not necessarily atomic, but can consist of several attributes that are assigned to users. Access to information is then restricted according to access control policies, which are specified as Boolean rules evaluated on sets of attributes. To provide user-centered means to support ABAC, the cryptographic research generalized IBE into Attribute-Based Encryption.

Attribute-Based Encryption (ABE) schemes [146] are cryptographic schemes that enable fine-grained access control to encrypted data, handling the case of a large number of recipients in an efficient way. As in IBE, ABE schemes also rely on a central authority to distribute private keys individually to each user. However, in ABE, users are able to encrypt messages while designating a set of authorized recipients, instead of just a single user. This is done, as is common in large systems, by assigning permissions in terms of attributes and access control policies. In this respect, there exist two types of ABE schemes: Ciphertext-Policy ABE (CP-ABE) and Key-Policy ABE (KP-ABE). The only difference between CP-ABE and KP-ABE is whether policies are attached to ciphertexts or to keys. In CP-ABE [150], the central authority assigns a set of attributes to each user by embedding those attributes into the keying material provided to them, and ciphertexts are associated with a policy on the total set of attributes. Conversely, in KP-ABE [151] keys held by users enclose policies, and ciphertexts are embedded with a set of attributes.

Attribute-Based Encryption schemes provide access control based on attributes, and the workflow of the process follows exactly that in IBE (see Figures 7, 8 and 9). As an example, consider the medical setting previously mentioned in the case of IBE. Using CP-ABE, in a Key Generation Phase (step 1 in Figure 7), the central administration authority can assign keys to the personnel by associating a list of attributes with each of them. For instance, a cardiac surgeon may be assigned the attributes "cardiology", "surgery" and "doctor", and an administrative clerk in the cardiology department may be assigned the attributes "administration" and "cardiology".

Now, resume the medical example above, and assume that a general practitioner wants to transfer a patient over to the cardiology department. Moreover, suppose that the patient file consists of very sensitive information, which should only be revealed to cardiac surgeons, as well as administrative personal data required for the admission process. In an Outsourcing Phase (step 2 in Figure 8), using CP-ABE, the practitioner can encrypt the sensitive information under the policy "cardiology" AND "surgery" (meaning that at least both attributes are needed for decryption), and the administrative information under the policy "cardiology" AND ("administration" OR "doctor"). Then the practitioner can upload the resulting ciphertexts to the cloud server, and any recipients will be able to retrieve it from the cloud server and (if authorized) decrypt the sensitive information (step 3 in Figure 9). This choice of policies ensures that the sensitive information can only be read by cardiac surgeons, and that the administrative information can be read by either administrative or medical personnel of the cardiology department. Note that non-cardiac surgeons would not have access to the sensitive information, and clerks from other departments would not be able to retrieve the administrative information either. Thus, this use of ABE effectively administers access control in terms of attributes, and it protects the information against users with unauthorized credentials.

A relevant issue in ABE relates to *revocation* mechanisms. Following the example above, one may want to revoke the access to a given patient file once her treatment has ended, or perhaps restrict it to a given period of time. In order to achieve this, the usual approach consists in embedding the validity period in policies and in the authority master public information. This problem has been extensively studied in the ABE literature, see for instance [152].

Known provably secure ABE schemes satisfy semantic security-like definitions. In general, the security guarantees of ABE and IBE state that ciphertexts encrypted under a certain attribute or identity can only reveal information about the underlying plaintext by using the private key attached to that attribute or identity. The ABE literature has investigated further security notions and settings, such as policy hiding [153], revocation mechanisms [152], different forms of policy [151] and multi-authority settings [154]. For a survey of Attribute-

Based Encryption in the context of cloud computing, see [155].

A further step to generalize PKE, IBE and ABE schemes altogether is brought by Functional Encryption (FE) [156,157]. While CP-ABE schemes let private key holders learn the decryption of ciphertexts they are authorized to receive, FE schemes let private key holders learn an arbitrary function of the underlying plaintext. That is, in FE schemes, every private key is associated with a particular function, and the decryption of a given ciphertext results in the function evaluated at the plaintext. FE is a very powerful cryptographic primitive, and its potential applications range from access control to encrypted search and predicate encryption [156].

In the context of cloud computing, *proxy re-encryption* (PRE) is an additional primitive that can be used in order to add access control to outsourced data. In a proxy re-encryption scheme [158,159,160], a semi-honest proxy can convert a ciphertext encrypted under a key to a ciphertext encrypted under another key. For instance, if the proxy receives a ciphertext encrypted under the public key of a user Alice, the proxy can convert it into a ciphertext encrypted under the public key of a user Bob. This conversion is made using a special key created by Alice and Bob, and the proxy does not have to know the underlying plaintext. Indeed, PRE schemes guarantee that the proxy cannot learn anything about the plaintext. Those PRE schemes tailored for identity-based encryption [161,162,163] are specially useful for cloud access control.

3.3.4 Storage controls on outsourced encrypted data

As mentioned above, the user may wish to control how her outsourced data are stored. That is, the user may want methods to check the integrity of the stored data, the data location, and the software that is used [164]. To that end, cryptographic protocols can be built around zero-knowledge proofs (ZKPs) [165] and similar primitives.

A ZKP involves two parties, the prover and the verifier. Initially, the prover holds some private knowledge. This knowledge can consist, for instance, of an integer value, a string or a solution to a particular mathematical problem. The prover wants to prove to the verifier that she has this private knowledge, but without revealing it to the verifier. The security guarantees of ZKP are completeness, soundness and zero-knowledge. Completeness states that the verifier must always become convinced that the prover holds the private knowledge if she actually does. Dually, soundness ensures that no cheating prover can convince the verifier that she holds the private knowledge in case she does not. Finally, zero-knowledge refers to the fact that the verifier learns no information on the private knowledge held by the prover.

The need for *proofs of data storage* arises when a cloud user first stores

some data in the cloud, and at a later moment he wants to make sure that the uploaded data are still intact. In this setting, remote data possession proofs [166,167,168] or proofs of retrievability [169,170] can be used to carry out this verification efficiently. Proofs of retrievability tolerate some leakage, because they are used in a context in which the verifier is the data owner himself.

Proofs of data location are motivated by the case in which cloud users want their data to be stored at a particular geographic location, be it because of the sensitive nature of the data or for legal reasons. By using IP geolocation techniques along with proofs of retrievability, it is possible [171,172] for the user to verify the location of the data. *Direct anonymous attestation* [173,174] is a privacy-preserving primitive that is used to remotely authenticate a trusted computer.

4 Comparison of data protection techniques

In Table 4 we compare the types of protection methods surveyed above in terms of the requirements identified in Section 2: supported operations, overhead at the local proxy, preservation of the accuracy of the original data, transparency and interoperability. Operations on protected data not listed in the table may still be supported but at a prohibitive cost (e.g. to update searchable encrypted data, the whole data set should be downloaded, decrypted, updated, encrypted and re-uploaded). In rest of this section, we elaborate on this comparison.

First, thanks to the very nature of the proxy-based solution, the user can be empowered with control on which data are protected and how. Specifically, the user can configure the trusted local proxy to specify which of her data are sensitive and select and configure the protection mechanism to be applied consistently with the functionalities she desires to outsource to the cloud.

4.1 Supported operations

All the methods allow generating protected data that can be stored in the cloud. The differences lie in the additional supported operations. Whereas plain encryption precludes any operation by the CSP who receives the encrypted data, the other methods are more flexible.

Anonymization yields a cleartext anonymized data set on which the CSP can conduct any calculation that the user could conduct on her original data set.

Table 3. Summary of cryptographic techniques for data protection in cloud computing

Technique	*References*	*Allowed operations*	*Usability*
Symmetric-Key Encryption	[63]	Encrypted storage. Encryption and retrieval in bulk	Enables no other functionalities over outsourced data
Searchable Encryption	[64] [72] [73] [83] [74] [78] [75] [76] [82] [77] [79] [81] [80]	Encrypted storage. Preserves efficient and expressive keyword search	Can impose significant leakage
Order-Preserving Encryption	[69]	Sort quantitative encrypted data by ascending or descending order	Imposes significant leakage, since encryption is deterministic
Partially HE	[91] [92]	Computations under a single operation on encrypted data	Limited range of computations
Somewhat HE	[94] [95] [96]	Computations with limited multiplicative depth over encrypted data	Limited range of possible computations
Fully HE	[97] [98] [99] [100] [101] [99] [102]	Arbitrary computations on encrypted data	Efficiency bottleneck in the computation stage
Two-Party Computation	[125] [142]	Arbitrary computations on distributed inputs between two parties	Costs increases with the function complexity
Multi-Party Computation	[126] [128] [129] [130] [131] [133] [132] [134] [136] [137] [139] [140] [141]	Arbitrary computations on distributed inputs	Communication complexity increases with the function complexity
Public-Key Encryption	[91] [92]	Access control with targeted recipient	Requires a PKI to bind public keys to recipients
Identity-Based Encryption	[143] [144] [145] [146] [147] [148]	Access control with targeted recipient	Requires the designation of a single recipient per ciphertext
Attribute-Based Encryption	[146] [150] [151] [153] [152] [154]	Attribute-based access control	Less efficient than Symmetric-Key Encryption
Functional Encryption	[156,157]	Lets recipients learn a designated function of the underlying plaintext	Non-standard security definitions, which may depend on the particular application.
Proxy Re-Encryption	[158] [159] [160] [161] [162] [163]	Switch the public key a ciphertext is encrypted under	Access control must be administered by a semi-honest proxy
Proofs of Storage	[166] [167] [169]	Verify that outsourced data are intact	Protects only static archived files

Table 4. Comparison of data protection technologies

Method	*Supported operations*	*Overhead at the local proxy*	*Data accuracy preservation*	*Data protection level*	*Transparency*	*Interoperability*
Anonymization	Storage. Queries on non-masked data. Any computation	Quasi-linear w.r.t. data size during storage. Zero in all other operations	May be partial. Depends on the calculation requested and the masking method. Same for end users and CSP	Data still in clear, but coarsened/distorted in an irreversible way	Transparent both for the proxy and CSP	Straightforwardly interoperable with external systems
Splitting	Storage. Updates. Queries. Computation on marginals. Joint computations require specific solutions	Constant for all operations	Full for end users. Full for CSP on the *partial* data it stores	Partial data in its original form. Could be broken via confabulation and/or compromised metadata on the splitting process	A multi-cloud or several cloud accounts are needed. Operations are transparent for CSP	Requires exchanging metadata on the splitting process (data splitting criteria and storage locations)
Encryption	Storage	Linear w.r.t. data size on all operations	Full for end users. None for CSP	Fully encrypted data. Could be broken via compromised keys	Key management at the local proxy	Requires exchanging keys
Homomorphic encryption	Storage. Computation of arithmetic circuits	Linear w.r.t. data size but involving very expensive primitives	Full for end users. None for CSP	Fully encrypted data. Could be broken via compromised keys	Key management at the local proxy. Special modules for ciphertext arithmetic to be installed at CSP	Requires exchanging keys
Searchable encryption	Storage. Queries (keyword, ranges, conjunctive)	Linear w.r.t. data size on all operations. More expensive primitives than plain encryption but less than homomorphic encryption	Full for end users. None for CSP	Fully encrypted data. Could be broken via compromised keys	Key management at the local proxy. Special modules for ciphertext search to be installed at CSP	Requires exchanging keys

Splitting also yields cleartext data, but each CSP only gets a fragment of the original data set; therefore the CSP can run any calculation on its fragment (e.g., marginal statistics), but operations involving fragments held by other CSPs require *ad hoc* protocols that orchestrate the calculations; that is, splitting the operation on the data fragments and aggregating the partial results, as discussed in Section 3.1.3. Additions/updates are fast and immediate because they can be done separately for each fragment. In contrast, if data stored in the cloud are encrypted or anonymized rather than split, any record addition/ update requires re-encrypting or re-anonymizing the original data set including the added/updated record and re-uploading the entire re-encrypted or re-anonymized data set. This is may be prohibitively costly.

Homomorphic encryption results in ciphertext being uploaded to the CSP, but the CSP can still carry out some operations on that ciphertext (the precise operations depend on the particular homomorphic cryptosystem in use). Finally, searchable encryption produces a ciphertext on which the CSP can conduct searches and hence perform a variety of queries (keyword search, range queries, conjunctive queries) and return the results to the user.

4.2 Overhead at the local proxy

The amount of work that the user's local proxy must carry out varies with the protection method used.

Most anonymization methods require quasi-linear (or even quadratic) computation in the number of records to generate the anonymized data set. Once the anonymized data set has been generated and uploaded to the cloud, no further intervention by the proxy is required.

Splitting takes constant work to split the original data set into fragments; after that, within-fragment operations by CSPs require no help from the proxy, but operations involving several fragments need constant computation by the local proxy (typically the proxy participates at the end of the *ad hoc* protocol run among the CSPs to decrypt the computation result or remove the noise from it).

As to plain encryption, it takes linear effort from the proxy to generate the encrypted data set (and to decrypt it if necessary). Homomorphic encryption also takes linear effort to generate the ciphertext and decrypt the results of operations conducted by the CSP on encrypted data; however, the complexity of encryption and decryption are typically much higher than with plain encryption and, together with the fact that certain computations cannot be efficiently outsourced, this renders (fully) homomorphic encryption impractical for many applications. Searchable encryption normally lies between plain

encryption and homomorphic encryption in what regards the amount of work required from the local proxy.

4.3 Accuracy preservation

All methods deliver full accuracy to the user except anonymization. The reason is that both splitting and encryption are *reversible* transformations. Since anonymization entails some irreversible modification, a certain computation conducted on the anonymized data set may yield results that are not exactly the ones that would be obtained on the original data set. Nonetheless, depending on the particular anonymization method, some statistics may be exactly preserved by the anonymized data set.

Among the methods delivering full accuracy, there are differences regarding the accuracy given to the CSP: whereas the CSP cannot learn any result if it stores encrypted data, it can learn the results of within-fragment computations when it stores split data.

4.4 Data protection level

Non-cryptographic methods produce less protected data than cryptographic counterparts. Whereas data anonymization mechanisms outsource the whole data in a coarsened or distorted way (according to the specific anonymization mechanism), data splitting produces data fragments of *original* data. In both cases, the configuration of the protection mechanism at the proxy is crucial to avoid disclosure risks. This implies deciding which attributes are direct identifiers (which should be either removed or split), and which are quasi-identifiers (which should be protected or stored in separate locations). If this characterization of attribute types is wrong or incomplete, the achieved protection may be ineffective.

Splitting also assumes that CSPs do not collude, e.g., in order to aggregate the data fragments they store. Also, the protection may be broken if the splitting metadata stored at the proxy are compromised, because such metadata allow reconstructing the complete original data. Anonymization is immune to leakages at the proxy because the anonymization process is irreversible and no metadata are stored. However, with dynamic data (i.e., records that are successively added/updated in the cloud) splitting offers more protection than anonymization: with the latter the CSP might infer the value of some original records by comparing the successive anonymized versions of the data set.

On the other hand, all cryptographic methods produce encrypted data that

are inherently safer than clear data. However, encryption can be undone if encryption keys stored in the proxy are leaked or by brute-force attacks if weak cryptographic primitives are used.

4.5 Transparency

Anonymization is entirely transparent: the proxy does not need to store any metadata or key material on the anonymized data, and the CSP can operate on them as if they were the original data.

For data splitting, a multi-cloud or several cloud accounts are needed to store data fragments, and the proxy needs to keep track of where each fragment has been stored; in contrast, the CSPs storing fragments do not need to perform any management or do any extra work. Thus splitting is transparent for the CSPs but not for the proxy.

Encryption solutions share the need for key management at the proxy, who needs to store all keys used; thus there is no transparency for the proxy. Furthermore, homomorphic encryption and searchable encryption also have a management impact on the CSP: operations on encrypted data being non-standard, special modules to implement them are needed at the CSP. So homomorphic encryption is neither transparent for the proxy nor for the CSP.

4.6 Interoperability

Anonymization is straightforwardly interoperable between several systems (proxies or clouds): the interoperating systems do not need to be aware of the anonymization methods applied to the data (they can treat the anonymized data as though they were original).

In data splitting, interoperating proxies need to share the metadata showing where each fragment has been stored.

Regarding encryption, interoperability requires the encryption keys to be shared among the interoperating proxies: this is a clear weakness, because if a key gets compromised during the transmission, the outsourced data will no longer be secure.

5 Research projects and products on privacy-aware computation outsourcing

In this section we review research projects and products that materialize some of the data protection techniques described above in the cloud scenario.

5.1 Research projects

The general concern about the security and privacy of outsourced data has motivated increased research on secure clouds. In the last decade the public sector has funded different initiatives. Next, we present some of the most relevant ones, among those that have been publicized.

In the United States of America, the National Science Foundation (NSF) and the Defense Advanced Research Projects Agency (DARPA) have supported several projects dedicated to cryptographic techniques for the cloud and secure cloud infrastructures. DARPA funded PROCEED [7], a project dedicated to computation on encrypted data with FHE and MPC, with the aim of building efficient solutions.

The Modular Approach to Cloud Security (MACS) project [8] is dedicated to the development of cryptographic techniques and tools for building information systems with multi-layered security guarantees. MACS was funded by a Frontier grant from the National Science Foundation's Secure and Trustworthy Cyberspace (SaTC) program. The results of this project were tested in the Mass Open Cloud [9]. Other related projects funded by NSF about secure cloud computing were NEBULA: A Future Internet That Supports Trustworthy Cloud Computing [10], Self Protecting Electronic Medical Records [11], Enabling Large-Scale, Privacy-Preserving Genomic Computing with a Hardware-Assisted Secure Big-Data Analytics Framework [12], Trustworthy Computing over Protected Datasets [13], and Trustworthy Virtual Cloud Computing [14].

A number of calls in Horizon 2020 (H2020), the European Union's current research and innovation programme, have been focused on cloud technologies.

[7] https://www.darpa.mil/program/programming-computation-on-encrypted-data
[8] https://www.bu.edu/macs/
[9] https://massopen.cloud/
[10] http://www.cs.cornell.edu/Projects/nebula/
[11] https://www.nsf.gov/awardsearch/showAward?AWD_ID=1010928
[12] https://www.nsf.gov/awardsearch/showAward?AWD_ID=1838083
[13] https://www.nsf.gov/awardsearch/showAward?AWD$_$ID=1739000
[14] https://www.nsf.gov/awardsearch/showAward?AWD_ID=0910767

Among the projects funded via these H2020 calls, the following are aligned with the topic of this survey, in the sense that they introduce cryptographic, anonymization or data splitting techniques in order to protect data in public clouds:

- *A Framework for User Centred Privacy and Security in the Cloud (CLARUS*[15] *)* provides a transparent solution for users to ensure control and privacy of their sensitive data outsourced in the cloud without losing the benefits of the cloud computing services. CLARUS is based on a proxy that supports privacy-preserving techniques based on searchable encryption, homomorphic encryption, data anonymization (via k-anonymity and t-closeness), and data splitting.
- *Enforceable Security in the Cloud to Uphold Data Ownership (ESCUDO-CLOUD*[16] *)* offers a solution that allows data owners to outsource their data while maintaining control on them. In particular, owners retain the ability to regulate access to data and share them with other users in a selective way. At the same time, owners can still rely on CSPs for data storage, processing and management.
- *Multi-cloud Secure Applications (MUSA*[17] *)* supports security-intelligent lifecycle management of distributed applications over heterogeneous cloud resources using a security framework. This framework includes security-by-design mechanisms to allow application self-protection at runtime, and methods and tools for integrated security assurance in both the engineering and operation of multi-cloud applications.
- *A Holistic Data Privacy and Security by Design Platform-as-a Service Framework (PaaSword*[18] *)* PaaSword introduces a data privacy and security-by-design framework based on distributed and encrypted data persistence and context-aware access control mechanisms in cloud-based services and applications.
- *Collaborative Software Development Framework based on Trusted, Secure Cloud-based Pool of Users (CloudTeams,* [19] *)* built a cloud-based platform that transforms software development for cloud services into an easier, faster and targeted process. CloudTeams is the intersection of crowd-sourcing platforms, collaborative software development tools and delivery of trusted cloud services.

[15] `http://www.clarussecure.eu`
[16] `http://www.escudocloud.eu/`
[17] `http://www.musa-project.eu`
[18] `https://www.paasword.eu/`
[19] `https://cordis.europa.eu/project/rcn/194223_en.html`

We have reviewed three categories of products that focus on the cloud scenario or that can be used to implement data protection proxies:

- Encryption products;
- Secure multi-party computation products;
- Anonymization products.

Each of the above categories is treated in the next sections. Furthermore, Table 5 compares the products based on encryption and secure multi-party computation, and Table 6 compares anonymization products.

Many of the reviewed products come from start-ups, but also multinational companies have their own solutions. This market has increased a lot in the last decade and is very dynamic.

5.2.1 Encryption products

Most current cloud services protect data by using cryptographic techniques. Public clouds like Google Drive or Dropbox encrypt the stored data using symmetric encryption schemes like AES, but usually the cryptographic keys are held by the cloud service provider. Hence, data are encrypted but users delegate the management of the keys to the cloud, and therefore implicitly trust the CSP and its data management policies.

In contrast, the products listed below allow users to manage their own data and manage their cryptographic keys. In this case, the CSP does not have access to the outsourced data, because they are uploaded to the cloud as ciphertext. Admittedly, hosting data that are already encrypted when uploaded is not very attractive for CSPs offering free accounts in public clouds, because they cannot inspect the uploaded content, for example in order to personalize ads sent to the user or detect storage of illegal items. However, from the user's point of view it is very attractive to ensure confidentiality vs the CSP, which is what the following products offer, while allowing some funcionalities on the encrypted data:

- *Skyhigh Security Cloud* provides protection for enterprise data and users across all cloud services. It combines different tokenization and encryption techniques in order to cover different levels of protection and utility requirements. In particular, it uses searchable encryption, order-preserving encryption and format-preserving encryption.
- *Bitglass* is a cloud access security broker that protects data with symmetric-key encryption and searchable encryption. It is a proxy-based solution that is

Table 5. Comparison of products based on cryptographic techniques

Product	*Supported operations*	*Techniques*	*Other features*
Skyhigh Security Cloud	Storage. Search Queries	Searchable encryption. Order-preserving encryption. Format-preserving encryption	Protection across different cloud services
Bitglass	Storage. Search and Sort Queries	Searchable encryption	Proxy-based solution for any cloud platform
CryptDB	Storage. Search and SQL Queries	Order-preserving encryption. Homomorphic encryption	Open source
CipherCloud CASB+	Storage	Encryption. Tokenization. Dynamic access control	
BoxCryptor	Storage and secure computation	Encryption	Data sharing. Supporting many cloud platforms
Sookasa	Storage	Encryption	Data sharing. Supporting Dropbox and Google Drive
nCrypted Cloud	Storage	Encryption	File sharing
Partisia	Secure computation	Encryption. Secure Multiparty Computation	Secure auctions. Statistics. Customized solutions
Sepior	Storage. Secure computation	Encryption. Secure Multiparty Computation	Amazon S3 plugin
Sharemind	Secure Computation	Encryption. Secure Multiparty Computation	Secure statistics. Customized solutions

supported by any cloud platform. Users can take control of the cryptographic keys, and encrypt data prior to uploading them to a cloud application. Its main feature is the Bitglass' split-index technology, which enables AES-256 encryption while supporting search queries and sort queries.

- *CryptDB* is an OpenSource product, with MIT license, that provides data confidentiality for applications that use database management systems. One of the main features of CryptDB is that it supports SQL queries on encrypted data. Data are encrypted in different ways in order to enable various types of queries, and SQL queries are translated by a proxy into the corresponding queries on encrypted data. The main cryptographic techniques that allow the execution of these queries are order-preserving encryption and homomorphic encryption.
- *CipherCloud CASB+* is a platform that combines a cloud access security broker with CipherCloud data protection techniques. It includes solutions for encryption, tokenization, dynamic access control, and cloud discovery and analysis. CipherCloud adds an interface between the client and the CSP, where protection technologies are applied. It is FIPS compliant, and uses AES as symmetric-key encryption scheme.
- *BoxCryptor* is a solution for encrypting data. It is supported by Dropbox, Google Drive, OneDrive and most of the commercial public clouds. BoxCryptor encrypts data on the user's device before transferring anything to the cloud storage provider. It combines RSA with AES-256 encryption.
- *Sookasa* has an encryption-based solution for Dropbox and Google Drive. Sookasa allows users to encrypt data with AES-256 and store the encrypted data in these cloud platforms, keeping the cloud advantages and the sharing options.
- *nCrypted Cloud* works with Dropbox, GoogleDrive, OneDrive, Box, and Egnyte. It uses AES 256-bit encryption. It offers encryption of cloud users' data, and also private file sharing, watermarking, access log and access notification features.

5.2.2 Secure multi-party computation-based solutions

Next we present a list of companies dedicated to the creation of privacy-preserving products that are based on secure multiparty computation techniques:

- *Partisia* was founded by the Alexandra Institute and the Partisia Holding and is dedicated to the design of secure multiparty computation systems. They provide different tailored products like secure surveys, privacy-preserving auctions, confidential benchmarking, loan brokers, and statistics.
- *Sepior* is dedicated to key management for cloud services. The key feature of their solutions is that keys are shared among different servers using secure multiparty computation. It allows server-side encryption and reencryption

of data. A specific plugin for Amazon S3 services is also available.
- *Sharemind* developed a secure computing platform that allows privacy-preserving computation of statistics and functions, in general. Data owners encrypt their data and upload them to the computing platform; the computations are performed on encrypted data, and the output is also encrypted.

5.2.3 Anonymization products

Unlike encryption, anonymization cannot be applied off-the-shelf. The data set to be anonymized needs to be carefully analyzed before anonymizing it. Decisions to be taken include: (i) determining the data uses for which the anonymized data have to stay useful (utility features to be preserved); (ii) determining the maximum tolerable disclosure risk; (iii) choosing a privacy model, an anonymization method and a parameterization according to the desired risk-utility trade-off; (iv) for most privacy models, determining which attributes should be regarded as quasi-identifiers given the background information available to a reasonable type of attacker (usual types include a nosy neighbor, a colleague, a journalist, etc.); (v) determining which attributes are confidential ones.

Keeping the above caveats in mind, several anonymization software packages are available that can be used to implement data protection proxies. We first review freeware products, that are all aimed at producing analytically useful anonymized data:

- *ARX*. This freeware package [175], developed by German academics, allows anonymizing an input data set according to a variety of privacy models, including k-anonymity, l-diversity, t-closeness and differential privacy.
- *UTD Toolbox*. This package is also freely available [176] and it offers k-anonymity, l-diversity and t-closeness based on the Incognito and Anatomy algorithms.
- *ARGUS*. This is an open-source freeware developed within the European official statistics community, with the financial support of the European Commission through a number of successive projects. It consists of two packages: μ-ARGUS [177] is devoted to protecting microdata (that is, data sets where each record corresponds to one subject) and τ-ARGUS [178] provides methods for anonymizing statistical tables. Both packages offer a choice of masking methods and risk assessment metrics. As usual in the official statistics community, *ex post* anonymization rather than *ex ante* is offered: instead of using privacy models, the idea is to mask data with a certain utility preservation in mind and then check whether the disclosure risk is low enough.
- *sdcMicro/sdcTable*. sdcMicro [179] is an open-source freeware package developed in R by Austrian statisticians that allows anonymizing microdata,

whereas its twin package sdcTable [180] is oriented to anonymization of statistical tables. Similar to the ARGUS packages, these packages are mostly used in the official statistics community.

Then there are a few commercial products whose purpose is different from that of the previous freeware products. Indeed, commercial solutions aim at producing fictitious data having the same format and appearance as (sensitive) exploitation data. These fictitious data are unlikely to preserve the analytical value of exploitation data, but they can safely be transferred to development environments for software debugging. Examples of such products are:

- *Data Masking and Subsetting Pack.* This product by Oracle offers a library of masking techniques and format transformations such as shuffling, noise addition, randomization or encryption.
- *IBM InfoSphere Optim Data Privacy.* This is also a commercial product that protects privacy by anonymizing sensitive exploitation data across applications, databases and operating systems. It provides a variety of masking transformations.
- *IBM InfoSphere Guardium.* This is a commercial database security and monitoring software that offers data masking features for cloud and big data platforms, including structured and unstructured data. Masking techniques in these products range from substrings to random number generation and concatenation.
- *Persistent Data Masking.* This product, developed by Informatica, shields confidential data from unintended exposure. The main masking techniques implemented in this product are data anonymization methods such as shuffling, randomization, nullification, hashing or substitution. It can be used for outsourcing data to the cloud, and also when using Hadoop and collaborative systems.
- *Dynamic Data Masking.* Also developed by Informatica, it is similar to Persistent Data Masking, but it allows setting flexible data masking rules based on a user's authentication level.

6 Conclusions and research challenges

In this survey, we have presented in a systematic way technologies that allow privacy-aware outsourcing of storage *and processing* of sensitive data to the cloud. This topic is especially hot given the coincidence of two phenomena: first, the sheer volume of personal or otherwise sensitive data being collected makes it increasingly necessary not only to store them in the cloud, but also to process them there; second, the upgrade of personal data protection that the new European General Data Protection Regulation has brought to Europe and beyond prevents storing and processing unprotected personal data in the

Table 6. Comparison of anonymization products. "Data format" indicates whether the product can work on individual data records (microdata), tables or other formats. Freeware products yield analytically useful data, whereas commercial products yield format-preserving data for use in software debugging.

Product	*Data format*	*Techniques*	*Other features*
ARX	Microdata	Privacy models (k-anonymity, l-diversity, t-closeness, diff. privacy, β-likeness, population and sample uniqueness)	Utility metrics. Freeware
UTD Toolbox	Microdata	k-Anonymity, l-diversity, t-closeness	Freeware
μ-ARGUS	Microdata	Local suppression, global recoding, PRAM, microaggregation, swapping, synthetic data	Utility and disclosure risk metrics. JAVA and SPSS on Windows. Freeware
τ-ARGUS	Tables	Cell suppression, controlled tabular adjustment, controlled rounding	JAVA on Windows. Freeware
sdcMicro	Microdata	Same as μ-Argus plus noise addition	Utility and disclosure risk metrics. R package. Freeware
sdcTable	Tables	Same as τ-Argus	R package. Freeware
Oracle Data Masking and Subsetting Pack	Microdata	Shuffling, noise addition, randomization or encryption	Format-preserving test data for software debugging. Commercial.
IBM InfoSphere Optim Data Privacy	Microdata	Same as previous	Same as previous
IBM InfoSphere Guardium	Microdata, unstructured data	Encryption, redaction, format-preserving recoding	Same as previous
Persistent Data Masking	Microdata	Substitution, noise addition, format-preserving recoding	Same as previous
Dynamic Data Masking	Microdata	Same as previous, but in real time	Same as previous

cloud.

Our analysis complements other surveys in the field by: i) considering the privacy issues related to data outsourcing to (untrusted) public clouds; ii) focusing on the preservation of cloud functionalities on outsourced data (beyond mere data storage); iii) surveying functionality-preserving data protection techniques not only based on (expensive) encryption, but also on (more efficient and flexible) anonymization and splitting; iv) comparing the surveyed techniques; and v) identifying research projects and products that offer privacy-enhancing and functionality-preserving solutions for the cloud or that can be employed to implement data protection proxies.

Each of the three examined data protection approaches (data splitting, anonymization and encryption) has its pros and cons, that have been analyzed above. Challenges for future research include mitigating some of the identified limitations, especially when it comes to outsourcing big data or at any rate large volumes of data:

- In [181], several requirements for big data anonymization were identified, namely composability, (quasi-)linear complexity, linkability and utility preservation for exploratory analyses. A privacy model is composable if, when satisfied by several data sets, the result of merging these data sets still satisfies the model. On the other hand, linkability refers to the ability to link similar subjects across several anonymized data sets, which is desirable when constructing big data by merging data sets with overlapping subjects. Finding privacy models and masking methods satisfying those requirements is an open research issue.
- Splitting raises the need to find *ad hoc* protocols among several CSPs to perform computations that involve several fragments. Using secure multiparty computation is a natural solution, but the resulting protocols are not yet efficient enough for huge data volumes. Non-cryptographic solutions are faster but they may not offer the same level of formal security.
- The design of cryptographic schemes supporting searches and computations on encrypted data is one of the most important research areas in cryptography. Such schemes offer high protection, but they currently reduce utility and add an important computation overhead. Their application in cloud computing is at an early stage, but is being explored by many companies and research institutions, and we expect cryptographic solutions for other functionalities to come out in the next few years.
- In the time of big data analytics, it would be interesting to be able to conduct exploratory analyses on encrypted data. Some progress in that direction is reported in [182,183], but certainly more research is needed to be able to conduct analyses on encrypted data that are as sophisticated as the ones that can be run on plain data.

In conclusion, combining big data of personal/sensitive nature with cloud processing for exploratory analytics under the new privacy regulations is an important issue that is likely to receive a lot of attention in the near future.

Disclaimer and acknowledgments

This work was partly supported by the European Commission (project H2020-700540 "CANVAS"), the Government of Catalonia (ICREA Acadèmia Prize to J. Domingo-Ferrer and grant 2017 SGR 705) and the Spanish Government (projects TIN2014-57364-C2-1-R "SmartGlacis", TIN2015-70054-REDC and TIN2016-80250-R "Sec-MCloud"). While the authors are with the UNESCO Chair in Data Privacy, the opinions expressed in this paper are the authors' own and do not necessarily reflect the views of UNESCO.

References

[1] Eurostat, Cloud computing - statistics on the use by enterprises (Dec. 2016 (Accessed 14 February 2019)).
URL http://ec.europa.eu/eurostat/statistics-explained/index.php/Cloud_computing_-_statistics_on_the_use_by_enterprises

[2] C. S. Alliance, Cloud usage: Risks and opportunities report (Sep. 2014 (Accessed 14 February 2019)).
URL https://downloads.cloudsecurityalliance.org/initiatives/collaborate/netskope/Cloud_Usage_Risks_and_Opportunities_Survey_Report.pdf

[3] T. Haeberlen, L. Dupré, Cloud computing. benefits, risks and recommendations for information security (rev. b), European Network and Information Security Agency (Dec. 2012).

[4] A. Westin, Privacy and Freedom, Atheneum, 1967.

[5] E. Ramirez, J. Brill, M. K. Ohlhausen, J. D. Wright, T. McSweeny, Data brokers: A call for transparency and accountability, U.S. Federal Trade Commission (May 2014).

[6] M. A. Khan, A survey of security issues for cloud computing, Journal of Network and Computer Applications 71 (2016) 11–29.

[7] S. Singh, Y.-S. Jeong, J. Park, A survey on cloud computing security: Issues, threats, and solutions, Journal of Network and Computer Applications 75 (2016) 200–222.

[8] A. Singh, K. Chatterjee, Cloud security issues and challenges: A survey, Journal of Network and Computer Applications 79 (2017) 88–115.

[9] P. Praveen-Kumar, P. Syam-Kumar, P. Alphonse, Attribute based encryption in cloud computing: A survey, gap analysis, and future directions, Journal of Network and Computer Applications 108 (2018) 37–52.

[10] N. Kaaniche, M. Laurent, Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms, Computer Communications 111 (2017) 120–141.

[11] J. Tang, Y. Cui, Q. Li, K. Ren, J. Liu, R. Buyya, Ensuring security and privacy preservation for cloud data services, ACM Computing Surveys 49 (1) (2016) Article No. 13.

[12] Z. Shan, K. Ren, M. Blanton, C. Wang, Practical secure computation outsourcing: A survey, ACM Computing Surveys 51 (2) (2018) Article No. 31.

[13] D. Sánchez, M. Batet, Privacy-preserving data outsourcing in the cloud via semantic data splitting, Computer Communications 110 (2017) 187–201.

[14] N. Cao, C. Wang, M. Li, K. Ren, W. Lou, Privacy-preserving multi-keyword ranked search over encrypted cloud data, IEEE Trans. Parallel Distrib. Syst. 25 (1) (2014) 222–233.

[15] M. Rouse, What is a multi-cloud strategy?, whatIs.com (Jul. 2014).
URL `https://searchcloudcomputing.techtarget.com`

[16] J. Domingo-Ferrer, S. Ricci, C. Domingo-Enrich, Outsourcing scalar products and matrix products on privacy-protected unencrypted data stored in untrusted clouds, Information Sciences 436–437 (2018) 320–342.

[17] J. Yang, J. Li, Y. Niu, A hybrid solution for privacy preserving medical data sharing in the cloud environment, Fut. Gen. Comp. Syst. 43–44 (2015) 74–86.

[18] G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, R. Motwani, U. Srivastava, D. Thomas, Y. Xu, Two can keep A secret: A distributed architecture for secure database services, in: Proceedings of the Conference on Innovative Data Systems Research (CIDR 2005), www.cidrdb.org, 2005, pp. 186–199.

[19] Z. Wei, S. Xinwei, Data privacy protection using multiple cloud storages, in: Proceedings of the International Conference on Mechatronic Sciences, 2013, pp. 1768–1772.

[20] H. Dev, T. Sen, M. Basak, M. E. Ali, An approach to protect the privacy of cloud data from data mining based attacks, in: Proceedings of 2012 SC Companion: High Performance Computing, Networking Storage and Analysis, 2012, pp. 1106–1115.

[21] M. Ali, K. Bilal, S. U. Khan, B. Veeravalli, K. Li, A. Y. Aomaya, Drops: Division and replication of data in cloud for optimal performance and security, IEEE Transactions on Cloud Computing 6 (2) (2018) 303–315.

[22] K. Gai, M. Qiu, H. Zhao, Security-aware efficient mass distributed storage approach for cloud systems in big data, in: 2nd International Conference on Bid Data Security on Cloud, 2016.

[23] H. Alqahtani, P. Sant, A multi-cloud approach for secure data storage on smart device, in: Sixth International Conference on Digital Information and Communication Technology and its Applications, 2016.

[24] V. Ciriani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, Fragmentation and encryption to enforce privacy in data storage, in: Proceedings of the 12th European Symposium On Research in Computer Security, 2007, pp. 171–186.

[25] V. Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, R. Motwani, Distributing data for secure database services, Trans. Data Priv. 5 (1) (2012) 253–272.

[26] V. Ciriani, S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, Selective data outsourcing for enforcing privacy, J. Comp. Sec. 19 (3) (2011) 531–566.

[27] D. Sánchez, M. Batet, C-sanitized: A privacy model for document redaction and sanitization, J. Assoc. Inf. Sci. Technol. 67 (1) (2016) 148–163.

[28] B. Goethals, S. Laur, H. Lipmaa, T. Mielikäinen, On private scalar product computation for privacy-preserving data mining, in: Proceedings of the International Conference on Information Security and Cryptology (ICISC 2004), 2004, pp. 104–120.

[29] A. F. Karr, X. Lin, A. P. Sanil, J. P. Reiter, Privacy-preserving analysis of vertically partitioned data using secure matrix products, J. Off. Stat. 25 (1) (2009) 125–138.

[30] A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, E. S. Nordholt, K. Spicer, P. de Wolf, Statistical Disclosure Control, Wiley, 2012.

[31] M. Elliot, J. Domingo-Ferrer, The future of statistical disclosure control, Tech. rep., National Statistician's Quality Review - Government Statistical Service, UK (Dec. 2018).
URL `https://gss.civilservice.gov.uk/wp-content/uploads/2018/12/12-12-18_FINAL_Mark_Elliot_Josep_Domingo-Ferrer.pdf`

[32] B. C. M. Fung, K. Wang, R. Chen, P. S. Yu, Privacy-preserving data publishing: A survey of recent developments, ACM Computing Surveys 42 (4) (2010) 14:1–14:53.

[33] N. Singh, A. K. Singh, Data privacy protection mechanisms in cloud, Data Science and Engineering 3 (1) (2018) 24–39.

[34] J. Domingo-Ferrer, V. Torra, Ordinal, continuous and heterogeneous k-anonymity through microaggregation, Data Mining and Knowledge Discovery 11 (2) (2005) 195–212.

[35] R. A. Moore, Controlled data swapping techniques for masking public use microdata sets, u.S. Bureau of the Census Research Report RR96/04 (1996). URL `http://www.census.gov/srd/papers/pdf/rr96-4.pdf`

[36] J. Domingo-Ferrer, D. Sánchez, J. Soria-Comas, Database Anonymization: Privacy Models, Data Utility, and Microaggregation-Based Inter-Model Connections, Synthesis Lectures on Information Security, Privacy, & Trust, Morgan & Claypool Publishers, 2016.

[37] C. Skinner, C. Marsh, S. Openshaw, C. Wymer, Disclosure avoidance for census microdata in great britain, in: Proceedings of the 1990 U.S. Bureau of the Census Annual Research Conference, 2004, pp. 131–196.

[38] P. Samarati, L. Sweeney, Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression, Tech. rep., SRI International Report (Aug. 1998).

[39] S. Martínez, D. Sánchez, A. Valls, M. Batet, Privacy protection of textual attributes through a semantic-based masking method, Inf. Fusion 13 (4) (2012) 304–314.

[40] R. Brand, Microdata protection through noise addition, in: Inference Control in Statistical Databases, Vol. 2316 of Lecture Notes in Computer Science, Springer, 2002, pp. 97–116.

[41] M. Rodríguez-Garcia, M. Batet, D. Sánchez, A semantic framework for noise addition with nominal data, Knowledge-Based Systems 122 (2017) 103–118.

[42] V. Torra, Rank swapping for partial orders and continuous variables, in: Proceedings of the Intl. Conf. on Availability, Reliability and Security (ARES 2009), IEEE Comp. Soc., 2009, pp. 888–893.

[43] M. Rodriguez-Garcia, M. Batet, D. Sánchez, Utility-preserving privacy protection of nominal data sets via semantic rank, Inf. Fusion 45 (2019) 282–295.

[44] K. Muralidhar, R. Sarathy, J. Domingo-Ferrer, Reverse mapping to preserve the marginal distributions of attributes in masked microdata, in: Proceedings of Privacy in Statistical Databases - UNESCO Chair in Data Privacy, Intl. Conference (PSD 2014), 2014, pp. 105–116.

[45] J. Domingo-Ferrer, J. M. Mateo-Sanz, Practical data-oriented microaggregation for statistical disclosure control, IEEE Trans. Knowl. Data Eng. 14 (1) (2002) 189–201.

[46] D. Defays, P. Nanopoulos, Panels of enterprises and confidentiality: the small aggregates method, in: Proceedings of the 92 Symp. on Design and Analysis of Longitudinal Surveys, 1993, pp. 195–204.

[47] S. Martínez, D. Sánchez, A. Valls, Semantic adaptive microaggregation of categorical microdata, Computers & Security 31 (5) (2012) 653–672.

[48] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, S. Martínez, t-closeness through microaggregation: Strict privacy with enhanced utility preservation, in: IEEE Trans. Knowl. Data Eng., IEEE Computer Society, 2016, pp. 1464–1465.

[49] D. Sánchez, S. Martínez, J. Domingo-Ferrer, Comment on 'unique in the shopping mall: On the reidentifiability of credit card metadata', Science 351 (2016) 1274.

[50] A. Machanavajjhala, D. Kifera, J. Gehrke, M. Venkitasubramaniam, L-diversity: Privacy beyond k-anonymity, ACM Transactions on Knowledge Discovery from Data 1 (1) (2007) 3.

[51] N. Li, T. Li, S. Venkatasubramanian, t-closeness: Privacy beyond k-anonymity and l-diversity, in: Proceedings of the International Conference on Data Engineering (ICDE 2007), IEEE Computer Society, 2007, pp. 106–115.

[52] N. Li, T. Li, S. Venkatasubramanian, Closeness: A new privacy measure for data publishing, IEEE Trans. Knowl. Data Eng. 22 (7) (2010) 943–956.

[53] C. Dwork, Differential privacy, in: Proceedings of Automata, Languages and Programming, Vol. 4052 of Lecture Notes in Computer Science, Springer, 2006, pp. 1–12.

[54] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, S. Martínez, Enhancing data utility in differential privacy via microaggregation-based k-anonymity, VLDB Journal 23 (5) (2014) 771–794.

[55] M. Hardt, K. Ligett, F. McSherry, A simple and practical algorithm for differentially private data release, in: Advances in Neural Information Processing Systems (NIPS 2012), 2012, pp. 2339–2347.

[56] J. Soria-Comas, J. Domingo-Ferrer, Differentially private data publishing via optimal univariate microaggregation and record perturbation, Knowledge-Based Systems 125 (2018) 13–23.

[57] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, X. Xiao, PrivBayes: private data release via Bayesian networks, ACM Transactions on Database Systems 42 (2017) 4.

[58] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, L. Vilhuber, Privacy: theory meets practice on the map, in: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE 2008), 2008, pp. 277–286.

[59] J. Snoke, A. Slavković, p-MSE mechanism: differentially private synthetic data with maximal distributional similarity, in: Proceedings of Privacy in Statistical Databases - UNESCO Chair in Data Privacy, Intl. Conference (PSD 2018), 2018, pp. 138–159.

[60] S. L. Garfinkel, J. M. Abowd, S. Powazek, Issues encountered deploying differential privacy, in: Proceedings of the 2018 Workshop on Privacy in the Electronic Society (WPES 2018), ACM, 2018, pp. 133–137.

[61] D. Sánchez, J. Domingo-Ferrer, S. Martínez, J. Soria-Comas, Utility-preserving differentially private data releases via individual ranking microaggregation, Inf. Fusion 30 (2016) 1–14.

[62] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, D. Megías, Individual differential privacy: A utility-preserving formulation of differential privacy guarantees, IEEE Transactions on Information Forensics and Security 12 (6) (2017) 1418–1429.

[63] J. Daemen, V. Rijmen, The Design of Rijndael, Springer-Verlag, Berlin, Heidelberg, 2002.

[64] D. X. Song, D. A. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in: IEEE Symposium on Security and Privacy, 2000, pp. 44–55.

[65] C. Bösch, P. H. Hartel, W. Jonker, A. Peter, A survey of provably secure searchable encryption, ACM Computing Surveys 47 (2) (2014) 18:1–18:51.

[66] O. Goldreich, Towards a theory of software protection and simulation by oblivious rams, in: Proceedings of the 19th Annual ACM Symp. on Theory of Computing (STOC 1987), 1987, pp. 182–194.

[67] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, Private information retrieval, in: Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1995), IEEE Computer Society, 1995, pp. 41–50.

[68] S. Patel, G. Persiano, K. Yeo, Private stateful information retrieval, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, New York, NY, USA, 2018, pp. 1002–1019. doi:10.1145/3243734.3243821.
URL http://doi.acm.org/10.1145/3243734.3243821

[69] R. Agrawal, J. Kiernan, R. Srikant, Y. Xu, Order-preserving encryption for numeric data, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2004, pp. 563–574.

[70] R. A. Popa, C. M. S. Redfield, N. Zeldovich, H. Balakrishnan, Cryptdb: Protecting confidentiality with encrypted query processing, in: Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011), 2011, pp. 85–100.

[71] M. Naveed, S. Kamara, C. V. Wright, Inference attacks on property-preserving encrypted databases, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 644–655.

[72] R. Curtmola, J. A. Garay, S. Kamara, R. Ostrovsky, Searchable symmetric encryption: Improved definitions and efficient constructions, J. Comp. 19 (5) (2011) 895–934.

[73] D. Cash, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, M. Steiner, Highly-scalable searchable symmetric encryption with support for boolean queries, in:

Proceedings of CRYPTO 2013 - 33rd Annual Cryptology Conference, 2013, pp. 353–373.

[74] S. Kamara, T. Moataz, SQL on structurally-encrypted databases, IACR Cryptology ePrint Archive (2016) 453.

[75] Q. Zheng, S. Xu, G. Ateniese, VABKS: verifiable attribute-based keyword search over outsourced encrypted data, in: Proceedings of the 2014 IEEE Conference on Computer Communications (INFOCOM 2014), 2014, pp. 522–530.

[76] S. Kamara, C. Papamanthou, T. Roeder, Dynamic searchable symmetric encryption, in: Proceedings of the ACM Conf. on Computer and Communications Security, 2012, pp. 965–976.

[77] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: Advances in Cryptology - Proceedings of EUROCRYPT 2004, Vol. 3027 of Lecture Notes in Computer Science, Springer, 2004, pp. 506–522.

[78] P. Xu, X. Gao, W. Wang, W. Susilo, Q. Wu, H. Jin, Dynamic searchable public-key ciphertexts with fast performance and practical security, IACR Cryptology ePrint Archive (2017) 741.

[79] X. Boyen, B. Waters, Anonymous hierarchical identity-based encryption (without random oracles), in: Annual International Cryptology Conference, Springer, 2006, pp. 290–307.

[80] D. J. Park, K. Kim, P. J. Lee, Public key encryption with conjunctive field keyword search, in: International Workshop on Information Security Applications, Springer, 2004, pp. 73–86.

[81] H. S. Rhee, J. H. Park, W. Susilo, D. H. Lee, Trapdoor security in a searchable public-key encryption scheme with a designated tester, J. Syst. Softw. 83 (5) (2010) 763–771.

[82] D. Boneh, B. Waters, Conjunctive, subset, and range queries on encrypted data, in: Proceedings of the 4th Theory of Cryptography Conference (TCC 2007), 2007, pp. 535–554.

[83] S. Faber, S. Jarecki, H. Krawczyk, Q. Nguyen, M. Rosu, M. Steiner, Rich queries on encrypted data: Beyond exact matches, in: Proceedings of ESORICS 2015 Part II - 20th European Symposium on Research in Computer Security, 2015, pp. 123–145.

[84] G. Kellaris, G. Kollios, K. Nissim, A. O'Neill, Generic attacks on secure outsourced databases, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, ACM, 2016, pp. 1329–1340.

[85] P. Grubbs, M.-S. Lacharite, B. Minaud, K. G. Paterson, Pump up the volume: Practical database reconstruction from volume leakage on range queries,

in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, 2018, pp. 315–331.

[86] M.-S. Lacharité, B. Minaud, K. G. Paterson, Improved reconstruction attacks on encrypted data using range query leakage, in: 2018 IEEE Symposium on Security and Privacy (SP), IEEE, 2018, pp. 297–314.

[87] Y. Zhang, J. Katz, C. Papamanthou, All your queries are belong to us: The power of file-injection attacks on searchable encryption, in: 25th USENIX Security Symposium (USENIX Security 16), USENIX Association, 2016, pp. 707–720.

[88] K. S. Kim, M. Kim, D. Lee, J. H. Park, W.-H. Kim, Forward secure dynamic searchable symmetric encryption with efficient updates, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17, ACM, 2017, pp. 1449–1463.

[89] S. Lai, S. Patranabis, A. Sakzad, J. K. Liu, D. Mukhopadhyay, R. Steinfeld, S.-F. Sun, D. Liu, C. Zuo, Result pattern hiding searchable encryption for conjunctive queries, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, 2018, pp. 745–762.

[90] S.-F. Sun, X. Yuan, J. K. Liu, R. Steinfeld, A. Sakzad, V. Vo, S. Nepal, Practical backward-secure searchable encryption from symmetric puncturable encryption, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, 2018, pp. 763–780.

[91] R. L. Rivest, A. Shamir, L. M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 21 (2) (1978) 120–126.

[92] T. El Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in: Advances in Cryptology, Proceedings of CRYPTO '84, 1984, pp. 10–18.

[93] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: Advances in Cryptology - Proceedings of EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, 1999, pp. 223–238.

[94] D. Boneh, E. Goh, K. Nissim, Evaluating 2-DNF formulas on ciphertexts, in: Proceedings of the Second Theory of Cryptography Conference (TCC 2005), 2005, pp. 325–341.

[95] D. Catalano, D. Fiore, Using linearly-homomorphic encryption to evaluate degree-2 functions on encrypted data, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 1518–1529.

[96] M. Barbosa, D. Catalano, D. Fiore, Labeled homomorphic encryption: Scalable and privacy-preserving processing of outsourced data, IACR Cryptology ePrint Archive (2017) 326.

[97] C. Gentry, Fully homomorphic encryption using ideal lattices, in: Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009), 2009, pp. 169–178.

[98] J. H. Cheon, J. Coron, J. Kim, M. S. Lee, T. Lepoint, M. Tibouchi, A. Yun, Batch fully homomorphic encryption over the integers, in: Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, 2013, pp. 315–335.

[99] Z. Brakerski, V. Vaikuntanathan, Efficient fully homomorphic encryption from (standard) LWE, SIAM Journal on Computing 43 (2) (2014) 831–871.

[100] Z. Brakerski, C. Gentry, V. Vaikuntanathan, (leveled) fully homomorphic encryption without bootstrapping, ACM Trans. Comput. Theory 6 (3) (2014) 13:1–13:36.

[101] J. Fan, F. Vercauteren, Somewhat practical fully homomorphic encryption, IACR Cryptology ePrint Archive (2012) 144.

[102] J. W. Bos, K. E. Lauter, J. Loftus, M. Naehrig, Improved security for a ring-based fully homomorphic encryption scheme, in: Proceedings of the 14th IMA International Conference in Cryptography and Coding(IMACC 2013), 2013, pp. 45–64.

[103] C. A. Melchor, S. Fau, C. Fontaine, G. Gogniat, R. Sirdey, Recent advances in homomorphic encryption: A possible future for signal processing in the encrypted domain, IEEE Signal Processing Magazine 30 (2) (2013) 108–117.

[104] I. Chillotti, N. Gama, M. Georgieva, M. Izabachène, Faster packed homomorphic operations and efficient circuit bootstrapping for tfhe, in: Advances in Cryptology – ASIACRYPT 2017, Springer International Publishing, 2017, pp. 377–408.

[105] F. Bourse, M. Minelli, M. Minihold, P. Paillier, Fast homomorphic evaluation of deep discretized neural networks, in: Advances in Cryptology – CRYPTO 2018, Springer International Publishing, Cham, 2018, pp. 483–512.

[106] X. Jiang, M. Kim, K. Lauter, Y. Song, Secure outsourced matrix computation and application to neural networks, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2018, pp. 1209–1222.

[107] A. Akavia, D. Feldman, H. Shaul, Secure search on encrypted data via multi-ring sketch, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, ACM, 2018, pp. 985–1001.

[108] K. Nissim, E. Weinreb, Communication efficient secure linear algebra, in: Proceedings of the Third Theory of Cryptography Conference (TCC 2006), 2006, pp. 522–541.

[109] R. Hall, S. E. Fienberg, Y. Nardi, Secure multiple linear regression based on homomorphic encryption, J. Off. Stat. 27 (4) (2011) 669–691.

[110] J. Alderman, B. R. Curtis, O. Farràs, K. M. Martin, J. Ribes-González, Private outsourced kriging interpolation, in: Proceedings of the FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Revised Selected Papers, 2017, pp. 75–90.

[111] W. Du, M. J. Atallah, Privacy-preserving cooperative statistical analysis, in: Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC 2001), 2001, pp. 102–110.

[112] M. Hirt, K. Sako, Efficient receipt-free voting based on homomorphic encryption, in: Advances in Cryptology - Proceedings of EUROCRYPT 2000, 2000, pp. 539–556.

[113] M. Kantarcioglu, W. Jiang, Y. Liu, B. Malin, A cryptographic approach to securely share and query genomic sequences, IEEE Trans. Inf. Technol. Biomed. 12 (5) (2008) 606–617.

[114] H. Lipmaa, First CPIR protocol with data-dependent computation, in: Information, Security and Cryptology - Proceedings of ICISC 2009, 12th International Conference, 2009, pp. 193–210.

[115] Y. Zhang, M. Blanton, Efficient secure and verifiable outsourcing of matrix multiplications, in: Information Security - 17th International Conference, ISC, 2014, pp. 158–178.

[116] J. Duan, J. Zhou, Y. Li, Secure and verifiable outsourcing of nonnegative matrix factorization (NMF), in: Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec, 2016, pp. 63–68.

[117] X. Lei, X. Liao, T. Huang, H. Li, C. Hu, Outsourcing large matrix inversion computation to A public cloud, IEEE Trans. Cloud Computing 1 (1).

[118] J. Wang, K. Ren, C. Wang, Q. Wang, Harnessing the cloud for securely outsourcing large-scale systems of linear equations, IEEE Transactions on Parallel & Distributed Systems 24 (2013) 1172–1181.

[119] H. Nie, X. Chen, J. Li, J. Liu, W. Lou, Efficient and verifiable algorithm for secure outsourcing of large-scale linear programming, in: 2014 IEEE 28th International Conference on Advanced Information Networking and Applications (AINA), Vol. 00, 2014, pp. 591–596.

[120] S. Hohenberger, A. Lysyanskaya, How to securely outsource cryptographic computations, in: Theory of Cryptography, Second Theory of Cryptography Conference, TCC, 2005, pp. 264–282.

[121] R. Gennaro, C. Gentry, B. Parno, Non-interactive verifiable computing: Outsourcing computation to untrusted workers, in: Proceedings of the 30th Annual Conference on Advances in Cryptology, CRYPTO'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 465–482.

[122] A. Kosba, C. Papamanthou, E. Shi, xjsnark: A framework for efficient verifiable computation, in: 2018 IEEE Symposium on Security and Privacy (SP), 2018, pp. 944–961.

[123] R. S. Wahby, Y. Ji, A. J. Blumberg, A. Shelat, J. Thaler, M. Walfish, T. Wies, Full accounting for verifiable outsourcing, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17, ACM, 2017, pp. 2071–2086.

[124] D. Demirel, L. Schabhser, J. Buchmann, Privately and Publicly Verifiable Computing Techniques: A Survey, 1st Edition, Springer Publishing Company, Incorporated, 2017.

[125] A. C. Yao, How to generate and exchange secrets (extended abstract), in: Proceedings of the 27th IEEE Symp. on Foundations of Comp. Science, IEEE Comp. Soc., 1986, pp. 162–167.

[126] O. Goldreich, S. Micali, A. Wigderson, Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems, J. ACM 38 (3) (1991) 691–729.

[127] B. Kreuter, A. Shelat, C. Shen, Billion-gate secure computation with malicious adversaries, in: Proceedings of the 21st USENIX Security Symposium, 2012, pp. 285–300.

[128] R. Cramer, R. Gennaro, B. Schoenmakers, A secure and optimally efficient multi-authority election scheme, European Transactions on Emerging Telecommunications Technologies 8 (5) (1997) 481–490.

[129] M. Hirt, Multi party computation: Efficient protocols, general adversaries, and voting, Ph.D. thesis, ETH Zurich, Zürich, Switzerland (Sep. 2001).

[130] P. Bogetoft, I. Damgård, T. P. Jakobsen, K. Nielsen, J. Pagter, T. Toft, A practical implementation of secure auctions based on multiparty integer computation, in: Proceedings of the 10th Intl. Financial Cryptography and Data Security Conf. (FC 2006), 2006, pp. 142–147.

[131] Y. Lindell, B. Pinkas, Privacy preserving data mining, Journal of Cryptology 15 (3) (2002) 177–206.

[132] M. J. Freedman, K. Nissim, B. Pinkas, Efficient private matching and set intersection, in: Advances in Cryptology - Proceedings of EUROCRYPT 2004, Vol. 3027 of Lecture Notes in Computer Science, Springer, 2004, pp. 1–19.

[133] L. Kissner, D. X. Song, Privacy-preserving set operations, in: Proceedings of CRYPTO 2005: 25th Annual International Cryptology Conference, 2005, pp. 241–257.

[134] A. López-Alt, E. Tromer, V. Vaikuntanathan, On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption, in: Proceedings of the 44th Annual ACM Symposium on Theory of Computing Conference (STOC 2012), 2012, pp. 1219–1234.

[135] R. Cramer, I. Damgård, J. B. Nielsen, Secure Multiparty Computation and Secret Sharing, Cambridge University Press, 2015.

[136] S. N. Premnath, Z. J. Haas, A practical, secure, and verifiable cloud computing for mobile systems, in: Proceedings of 11th International Conference on Mobile Systems and Pervasive Computing, Vol. 34 of Procedia Computer Science, Elsevier, 2014, pp. 474–483.

[137] M. Jensen, J. Schwenk, J. Bohli, N. Gruschka, L. L. Iacono, Security prospects through cloud computing by adopting multiple clouds, in: Proceedings of the IEEE Fourth International Conference in Cloud Computing, IEEE Computer Society, 2011, pp. 565–572.

[138] F. Wang, J. Mickens, N. Zeldovich, V. Vaikuntanathan, Sieve: Cryptographically enforced access control for user data in untrusted clouds, in: 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16), USENIX Association, 2016, pp. 611–626.

[139] T. P. Jakobsen, Practical aspects of secure multiparty computation, Ph.D. thesis, Aarhus University, Aarhus, Denmark (Oct. 2015).

[140] S. Kamara, P. Mohassel, M. Raykova, Outsourcing multi-party computation, IACR Cryptology ePrint Archive (2011) 272.

[141] B. Pinkas, T. Schneider, N. P. Smart, S. C. Williams, Secure two-party computation is practical, in: Advances in Cryptology - Proceedings of ASIACRYPT 2009, Vol. 5912 of Lecture Notes in Computer Science, Springer, 2009, pp. 250–267.

[142] R. Canetti, Y. Ishai, R. Kumar, M. K. Reiter, R. Rubinfeld, R. N. Wright, Selective private function evaluation with applications to private statistics, in: Proceedings of the 20th Annual ACM Symposium on Principles of Distributed Computing, ACM, 2001, pp. 293–304.

[143] D. Boneh, M. K. Franklin, Identity-based encryption from the weil pairing, SIAM Journal on Computing 32 (3) (2003) 586–615.

[144] B. Waters, Efficient identity-based encryption without random oracles, in: Proceedings of EUROCRYPT 2005, Vol. 3494, Springer, 2005, pp. 114–127.

[145] D. Boneh, X. Boyen, Efficient selective-id secure identity-based encryption without random oracles, in: Advances in Cryptology - Proceedings of EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, 2004, pp. 223–238.

[146] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: Advances in Cryptology - EUROCRYPT 2005, Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2005, pp. 457–473.

[147] B. Waters, Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions, in: Advances in Cryptology - Proceedings of CRYPTO 2009, Vol. 5677 of Lecture Notes in Computer Science, Springer, 2009, pp. 619–636.

[148] A. Boldyreva, V. Goyal, V. Kumar, Identity-based encryption with efficient revocation, in: Proceedings of the 15th ACM Conference on Computer and Communications Security, ACM, 2008, pp. 417–426.

[149] J. Baek, J. Newmarch, R. Safavi-Naini, W. Susilo, A survey of identity-based cryptography, in: Proceedings of Australian Unix Users Group Annual Conference, 2004, pp. 95–102.

[150] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: Proceedings of the 2007 IEEE Symposium on Security and Privacy, 2007, pp. 321–334.

[151] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS 2006), ACM, 2006, pp. 89–98.

[152] N. Attrapadung, H. Imai, Attribute-based encryption supporting direct / indirect revocation modes, in: Proceedings of the 12th IMA International Conference on Cryptography and Coding, Vol. 5921 of Lecture Notes in Computer Science, Springer, 2009, pp. 278–300.

[153] J. Katz, A. Sahai, B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, in: Proceedings of EUROCRYPT 2008, 27th Annual International Conf. on the Theory and Applications of Cryptographic Techniques, 2008, pp. 146–162.

[154] M. Chase, Multi-authority attribute based encryption, in: Proceedings of the Theory of Cryptography Conference (TCC 2007), Vol. 4392, Springer, 2007, pp. 515–534.

[155] C. Lee, P. Chung, M. Hwang, A survey on attribute-based encryption schemes of access control in cloud environments, Int. J. Netw. Secur. 15 (4) (2013) 231–240.

[156] D. Boneh, A. Sahai, B. Waters, Functional encryption: Definitions and challenges, in: Proceedings of the Theory of Cryptography Conference (TCC 2011), Vol. 6597 of Lecture Notes in Computer Science, Springer, 2011, pp. 253–273.

[157] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, B. Waters, Candidate indistinguishability obfuscation and functional encryption for all circuits, SIAM Journal on Computing 45 (3) (2016) 882–929.

[158] G. Ateniese, K. Fu, M. Green, S. Hohenberger, Improved proxy re-encryption schemes with applications to secure distributed storage, ACM Transactions on Information and System Security 9 (1) (2006) 1–30.

[159] M. Blaze, G. Bleumer, M. Strauss, Divertible protocols and atomic proxy cryptography, in: Advances in Cryptology - Proceedings of EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, 1998, pp. 127–144.

[160] R. Canetti, S. Hohenberger, Chosen-ciphertext secure proxy re-encryption, in: Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007, pp. 185–194.

[161] M. Green, G. Ateniese, Identity-based proxy re-encryption, in: Applied Cryptography and Network Security - Proceedings of ACNS 2007, Vol. 4521, Springer, 2007, pp. 288–306.

[162] K. Liang, J. K. Liu, D. S. Wong, W. Susilo, An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing, in: Proceedings of ESORICS 2014, Vol. 8712, Springer, 2014, pp. 257–272.

[163] Y. Zhou, H. Deng, Q. Wu, B. Qin, J. Liu, Y. Ding, Identity based proxy re-encryption version 2: Making mobile access easy in cloud, Fut. Gen. Comp. Syst. 62 (2016) 128–139.

[164] N. P. Smart, Study on cryptographic protocols, European Union Agency for Network and Information Security (ENISA), 2014.

[165] S. Goldwasser, S. Micali, C. Rackoff, The knowledge complexity of interactive proof systems, SIAM Journal on Computing 18 (1) (1989) 186–208.

[166] F. Sebé, J. Domingo-Ferrer, A. Martínez-Ballesté, Y. Deswarte, J. Quisquater, Efficient remote data possession checking in critical information infrastructures, IEEE Trans. on Knowledge and Data Engineering 20 (8) (2008) 1034–1038.

[167] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, D. X. Song, Provable data possession at untrusted stores, in: Proceedings of the 14th ACM Conference on Computer and Communications Security, ACM, 2007, pp. 598–609.

[168] C. C. Erway, A. Küpçü, C. Papamanthou, R. Tamassia, Dynamic provable data possession, ACM Trans. Inf. Syst. Secur. 17 (4) (2015) 15:1–15:29.

[169] A. Juels, B. S. K. Jr., PORs: proofs of retrievability for large files, in: Proceedings of the 14th ACM Conference on Computer and Communications Security, ACM, 2007, pp. 584–597.

[170] D. Cash, A. Küpçü, D. Wichs, Dynamic proofs of retrievability via oblivious ram, Journal of Cryptology 30 (1) (2017) 22–57.

[171] G. J. Watson, R. Safavi-Naini, M. Alimomeni, M. E. Locasto, S. Narayan, LoSt: location based storage, in: Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW 2012), 2012, pp. 59–70.

[172] L. Hippelainen, I. Oliver, S. Lal, Towards dependably detecting geolocation of cloud servers, in: Network and System Security, Springer International Publishing, 2017, pp. 643–656.

[173] E. F. Brickell, J. Camenisch, L. Chen, Direct anonymous attestation, in: Proceedings of the 11th ACM Conference on Computer and Communications Security, ACM, 2004, pp. 132–145.

[174] J. Camenisch, M. Drijvers, A. Lehmann, Anonymous attestation with subverted tpms, in: Advances in Cryptology – CRYPTO 2017, Springer International Publishing, 2017, pp. 427–461.

[175] F. Prasser, F. Kohlmayer, K. Babioch, I. Vujosevic, R. Bild, ARX - data anonymization tool (2018).
URL http://arx.deidentifier.org

[176] M. Kantarcioglu, A. Inan, M. Kuzu, UTD anonymization toolbox (2012).
URL http://cs.utdallas.edu/dspl/cgi-bin/toolbox

[177] A. Hundepool, R. Ramaswamy, P.-P. de Wolf, L. Franconi, R. Brand, J. Domingo-Ferrer, μ-ARGUS (2018).
URL http://research.cbs.nl/casc/mu.htm

[178] A. Hundepool, A. Van de Wetering, R. Ramaswamy, P.-P. de Wolf, S. Giessing, M. Fischetti, J. J. Salazar, J. Castro, P. Lowthian, τ-ARGUS (2018).
URL http://research.cbs.nl/casc/tau.htm

[179] M. Templ, B. Meindl, A. Kowarik, sdcmicro: Statistical disclosure control methods for anonymization of microdata (2018).
URL https://cran.r-project.org/web/packages/sdcMicro/index.html

[180] B. Meindl, sdctable: Methods for statistical disclosure control in tabular data (2017).
URL https://cran.r-project.org/web/packages/sdcTable/index.html

[181] J. Soria-Comas, J. Domingo-Ferrer, Big data privacy: Challenges to privacy principles and models, Data Science and Engineering 1 (1) (2016) 21–28.

[182] X. Liu, R. Lu, J. Ma, L. Chen, H. Bao, Efficient and privacy-preserving skyline computation framework across domains, Fut. Gen. Comp. Syst. 62 (2016) 161–174.

[183] K. R. Choo, J. Domingo-Ferrer, L. Zhang, Cloud cryptography: Theory, practice and future research directions, Fut. Gen. Comp. Syst. 62 (2016) 51–53.