Chapter 1
Introduction

The amount of data stored, processed, and exchanged by private companies and public organizations is rapidly increasing. As a consequence, users are today, with increasing frequency, resorting to service providers for disseminating and sharing resources they want to make available to others. The protection against privacy violations is becoming therefore one of the most important issues that must be addressed in such an open and collaborative context. In this book, we define a comprehensive approach for protecting sensitive information when it is stored on systems that are not under the data owner's direct control. In the remainder of this chapter, we give the motivation and the outline of this book.

1.1 Motivation

The rapid evolution of storage, processing, and communication technologies is changing the traditional information system architecture adopted by both private companies and public organizations. This change is necessary for mainly two reasons. First, the amount of information held by organizations is increasing very quickly thanks to the growing storage capacity and computational power of modern devices. Second, the data collected by organizations contain sensitive information (e.g., identifying information, financial data, health diagnosis) whose confidentiality must be preserved.

Systems storing and managing these data collections should be *secure* both from external users breaking the system and from malicious insiders. However, the design, realization, and management of a secure system able to grant the confidentiality of sensitive data might be very expensive. Due to the growing costs of in-house storage and management of large collections of sensitive data, since it demands for both storage capacity and skilled administrative personnel, *data outsourcing and dissemination* services have recently seen considerable growth and promise to become a common component of the future Web, as testifies by the growing success of Web companies offering storage and distribution services (e.g., MySpace, Flickr,

S. Foresti, *Preserving Privacy in Data Outsourcing*, Advances in Information Security 51,
DOI 10.1007/978-1-4419-7659-8_1, © Springer Science+Business Media, LLC 2011

and YouTube). The main consequence of this trend is that companies often store their data on external *honest-but-curious* servers, which are relied upon for ensuring availability of data and for enforcing the basic security control on the data they store. While trustworthy with respect to their services in making published information available, these external systems are however trusted neither to access the content nor to fully enforce access control policy and privacy protection requirements.

It is then clear that users as well as the companies would find an interesting opportunity in the use of a dissemination service offering strong guarantees about the protection of user privacy against both malicious users breaking into the system and the service provider itself. Indeed, besides well-known risks of confidentiality and privacy breaks, threats to outsourced data include improper use of information: the service provider could use substantial parts of a collection of data gathered and organized by the data owner, potentially harming the data owner's market for any product or service that incorporates that collection of information.

There are mainly three security aspects that need to be considered when designing a system for ensuring confidentiality of data stored and managed by a honest-but-curious server, as briefly outlined in the following.

- *Access control enforcement.* Traditional architectures assign a crucial role to the *reference monitor* [7] for access control enforcement. The reference monitor is the system component responsible of the validation of access requests. The scenario considered in this book however challenges one of the basic tenets of traditional architectures, where a trusted server is in charge of defining and enforcing access control policies. This assumption no longer holds here, because the server does not even have to know the access defined (and possibly modified) by the data owner. We therefore need to rethink the notion of access control in open environments, where honest-but-curious servers are in charge of managing the data collection and are not trusted with respect to the data confidentiality.
- *Privacy protection.* The vast amounts of data collected and maintained by organizations often include sensitive personally identifiable information. This trend has raised the attention of both individuals and legislators, which are forcing organizations to provide privacy guarantees over sensitive information when storing, processing or sharing it with others. Indeed, recent regulations [22, 78] explicitly require specific categories of sensitive information to be either *encrypted* or *kept separate* from other personally identifiable information to grant confidentiality. Since encryption makes access to stored data inefficient, because it is not possible to directly evaluate queries on encrypted data, it is necessary to define new solutions that grant data confidentiality and efficient query evaluation.
- *Safe data integration.* More and more emerging scenarios require different parties, each withholding large amounts of independently managed information, to cooperate for sharing their information. Since the data collection detained by each subject contains sensitive information, classical distributed query evaluation mechanisms cannot be adopted [23, 64]. We therefore need an approach for regulating data flows among parties and for redefining query evaluation mechanisms to the aim of fulfilling access control restrictions imposed by each party. Indeed, data flows among the cooperating parties may be prohibited by privacy

constraints, thus making the design of query execution depending on both efficiency principles and privacy constraints.

There are many real-life examples of applications need a mechanism to exchange and disclose data in a selective and secure way. We outline here three possible scenarios.

Multimedia sharing systems. The amount of multimedia data people collect every day is quickly increasing. As a consequence, systems offering storage and distribution services for photographs and videos are becoming more and more popular. However, these data may be sensitive (e.g., photographs retracting people) and their wide diffusion on the Internet should be prevented if not explicitly authorized by the data owner. Since the distribution service may not be trusted with respect to data confidentiality, it cannot enforce the access control policy defined by the data owner. Therefore, it is necessary to think to an alternative solution to prevent sensitive data publication.

Healthcare system. More and more healthcare systems collect sensitive information about historical and present hospitalizations, diagnosis, and more in general health conditions of patients. Since these data, associated with the identity of patients, are sensitive, their storage, management, and distribution is subject to both state-level and international regulations. As a consequence, any healthcare system should adopt an adequate privacy protection system, which guarantees, for example, that sensitive information is never stored together with patients' identity.

Recently, the functionalities of healthcare systems have been extended, thanks also to the evolution and wide diffusion of network communication technologies, to allow data exchange among cooperating parties, such as medical personnel, pharmacies, insurance companies, and the patients themselves. Even if this solution improves the quality of the service offered to patients, it however needs to be carefully designed to avoid non authorized data disclosure. It is therefore necessary to define a data integration protocol that guarantees data confidentiality.

Financial system. Financial systems store sensitive information that needs to be adequately protected. As an example, the data collected by companies for credit card payments are sensitive and need protection both when stored and managed (e.g., credit card numbers and the corresponding security codes cannot be stored together), as demanded by law. Furthermore, thanks also to the wide diffusion of online transactions, the amount of financial data that systems need to manage and protect is increasing very quickly. Financial systems, as well as healthcare systems, need also to cooperate with other parties, managing independent data collections, such as governmental offices, credit card companies, and clients.

From the above description, it is straightforward to see that the security problems envisioned for healthcare systems apply also to the financial scenario, which demands for the same solutions and technologies for guaranteeing data confidentiality in data storage and exchange.

1.2 Contribution of the Book

The book provides an analysis of the main problems arising when the data owner does not directly control her data, since they are manager and/or stored by a honest-but-curious server. The contributions of this book focus on the three security aspects above-mentioned, that is, access control enforcement, privacy protection, and safe data integration. In the remainder of this section, we present the contributions in more details.

1.2.1 Access Control Enforcement

The first important contribution of this book is the proposal of a model for access control enforcement on encrypted, possibly outsourced, data [15, 41, 44]. The original contribution of our work can be summarized as follows.

Selective encryption. An access control system protecting data stored by a honest-but-curious system cannot rely on a trusted component (i.e., the reference monitor) that evaluates clients' requests. Since the data owner cannot act as an intermediary for data accesses, the access control policy should be embedded in the stored data themselves. Preliminary solutions try to overcome this issue proposing a novel access control model and architecture that eliminates the need for a reference monitor and relies on cryptography to ensure confidentiality of the data stored on the server. New solutions instead propose to combine authorization policy and encryption, thus allowing access control enforcement to be delegated together with the data. The great advantage is that the data owner, while specifying the policy, does not need to be involved in its enforcement. The access control system illustrated in this book exploits this same idea: different portions of the data are encrypted using different encryption keys, which are then distributed to users according to their access privileges. The model proposed in this book differs from previous ones since it exploits key derivation methods [8, 31] to limit the number of secret keys that users and the data owner herself need to securely manage. Key derivation methods allow the derivation of a secret key from another key by exploiting a piece of publicly available information. This solution allows us to reduce the amount of sensitive information that users and owners have to protect against third parties.

Efficient access to data. Since key derivation requires a search process in the catalog of publicly available information and the evaluation of a function, the key derivation process may become expensive from the client's point of view. In fact, the public catalog is stored at the provider's site and therefore any search operation implies a communication between the client and the server. To limit the burden due to the key derivation process, in this book we propose a solution that tries to minimize the size of the public catalog. Since such a minimization problem is *NP-hard*, we present a heuristic solution that experimentally obtains good results.

Policy updates. Since access control enforcement bases on selective encryption, any time the policy changes, it is necessary for the data owner to re-encrypt the data to reflect the new policy. However, the re-encryption process is expensive from the data owner's point of view, since it requires interaction with the remote server. To reduce the burden due to this data exchange process, we propose a two-layer encryption model where a inner layer is imposed by the owner for providing initial protection and an outer layer is imposed by the server to reflect policy modifications. The combination of the two layers provides an efficient and robust solution, which avoids data re-encryption while correctly managing policy updates.

Collusion model. An important aspect that should always be taken into account when designing a security system is its protection degree. To this purpose, we analyzed the security of the two-layer model with respect to the risk of collusion among the parties interacting in the considered scenario. In particular, we consider the case when the server, knowing the encryption keys adopted at the outer layer, and a user, knowing a subset of the keys adopted at the inner layer, collude to gain information that none of them is authorized to access. From this analysis, it is clear that the proposed model introduces a low collusion risk, which can be further reduced at the cost of a less efficient query evaluation process.

1.2.2 Privacy Protection

The second contribution we present in this book is a system that nicely combines fragmentation and encryption for privacy purposes [29, 27, 28]. The original contribution of our work can be summarized as follows.

Confidentiality constraints. The release, storage, and management of data is nowadays subject to a number of rules, imposed by either legislators or data owners, aimed at preserving the privacy of sensitive information. Since not all the data in a collection are sensitive per se, but their association with other information may need to be protected, solutions encrypting the whole data may be an overdo. Therefore, recently solutions combining fragmentation and encryption have been proposed [2]. In this book, we propose a simple while expressive model for representing privacy requirements, called *confidentiality constraints* that exploits fragmentation and encryption for enforcing such constraints. A confidentiality constraint is a set of attributes whose joint visibility should be prevented; a singleton constraint indicates that the values of the single attribute need to be kept private. This model, while simple, nicely captures different privacy requirements that need to be enforced on a data collection (e.g., sensitive data and sensitive associations).

Minimality. The main goal of the approach proposed in this book is to minimize the use of encryption for privacy protection. A trivial solution for solving confidentiality constraints consists in creating a fragment for each attribute that does not appear in a singleton constraint. Obviously such a solution is not desiderated, unless demanded by constraints, since it makes query evaluation inefficient. Indeed, since fragments

cannot be joined by non authorized users, the client posing the query would be in charge of combining the data extracted from the different fragments. To avoid such a situation, we propose three different models for designing a fragmentation that, while granting privacy protection, maximizes query evaluation efficiency. The three solutions differ in the efficiency measure proposed (i.e., number of fragments, affinity among attributes, query workload).

Query evaluation. Data fragmentation is usually transparent to the final user, meaning that queries are formulated on the original schema and then they are reformulated to operate on fragments. Since, as already noted, encryption and fragmentation reduce the efficiency in data retrieval, we propose to add indexes to fragments. Indexes are defined on attributes that do not appear in clear form in the fragment. Also, since indexes may open the door to inference and linking attacks, we carefully analyze the exposure risk due to different indexing methods, considering the external knowledge of a possible malicious user.

1.2.3 Safe Data Integration

The third and last contribution we present in this book is a solution for the integration of data from different data sources, which must be subject to confidentiality constraints [42, 43]. The original contribution coming from our work can be summarized as follows.

Access control model. We present a simple, yet powerful, approach for the specification and enforcement of permissions regulating data release among data holders collaborating in a distributed computation, to ensure that query processing discloses only data whose release has been explicitly authorized. The model is based on the concept of *profile*, which nicely models both the information carried by the result of a query, and the information whose release is authorized by permissions. To easily evaluate when a data release is allowed by the permissions of the requesting subject, we propose a graph based model. Profiles are then represented by adequately coloring the graph. The process of controlling if a query must be denied or allowed is then based on the comparison of the colors of vertices and edges in the graphs representing the query and the permissions in the system.

Permission composition. The amount of data that need to be integrated is potentially large and therefore it is not possible to check queries against single permissions, since the number of permissions to be explicitly defined would increase quickly. We then introduce the principle that a query must be allowed if the information release it (directly or indirectly) entails is allowed by the permissions. In other words, if the subject formulating the query is able to compute its result by combining information she is allowed to access, then the query should be allowed. To enforce this basic principle, we propose a permission composition method, which is based on reachability properties on the graphs representing the profiles of the permissions. The composition method proposed has the great advantage of working in

polynomial time, even if the number of possible composed permissions is exponential in the number of base permissions. This is due to a nice dominance property, which we prove in this book, between composed permissions and their components.

Safe query planning. Besides defining and composing permissions, it is necessary to evaluate if a query operating in the distributed scenario can be executed (i.e., the query is safe) or if the query must be denied. To this purpose, we characterize the flows of information among the interacting subjects for the evaluation of the given query, considering also different methods for executing join operations between distinct data sources. A query is therefore safe if all the data flows it requires for its evaluation are allowed by the set of (composed) permissions characterizing the system. We present an algorithm that given a query checks if the query can be evaluated without violating the set of permissions regulating the distributed system. If the query can be safely executed, the algorithm we propose also determines which server is in charge for executing which operation.

1.3 Organization of the Book

In this chapter, we discussed the motivation and the main objectives of our work and described the major contributions of this book. The remaining chapters are structured as follows.

Chapter 2 discusses the state of the art of the security aspects related to the objectives of the book. It presents the main results obtained in the data outsourcing scenario, focusing on mechanisms for query evaluation, inference exposure measurement, and data integrity. Also, it introduces preliminary works on access control enforcement, privacy protection, and data integration in the considered scenario.

Chapter 3 illustrates our access control system for securing data stored at a honest-but-curious server and proposes an efficient mechanism for managing access control policy updates. The risk of collusion among parties is also analyzed to prove the security of the presented solution.

Chapter 4 addresses the problem of modeling and enforcing privacy requirements to protect sensitive data and/or their associations. It also presents three cost models for computing an optimal fragmentation, that is, a fragmentation that allows efficient query evaluation.

Chapter 5 focuses on the problem of integrating data made available from different parties and that must satisfy security constraints. It proposes a model for expressing restrictions on data flows among parties and a mechanism for querying distributed data collections under these constraints.

Chapter 6 summarizes the contributions of this book and outlines future work.