Data Security and Privacy for Outsourced Data in the Cloud

Cetin Sahin and Amr El Abbadi
University of California, Santa Barbara
{cetin, amr}@cs.ucsb.edu

Abstract—**Although outsourcing data to cloud storage has become popular, the increasing concerns about data security and privacy in the cloud limits broader cloud adoption. Ensuring data security and privacy, therefore, is crucial for better and broader adoption of the cloud. This tutorial provides a comprehensive analysis of the state-of-the-art in the context of data security and privacy for outsourced data. We aim to cover common security and privacy threats for outsourced data, and relevant novel schemes and techniques with their design choices regarding security, privacy, functionality, and performance. Our explicit focus is on recent schemes from both the database and the cryptography and security communities that enable query processing over encrypted data and access oblivious cloud storage systems.**

I. Introduction

Recent advances in cloud technologies have made outsourcing personal and corporate data to cloud storage servers increasingly popular and attractive. However, this increase in utility comes with a risk of exposing data to a number of security threats. For example, a curious administrator might snoop on private data or an adversary might gain unauthorized access to sensitive information. Therefore, potential customers remain skeptical about joining the cloud due to existing confidentiality and privacy concerns [17]. For broader adoption of cloud services, concerns about data security and privacy must be addressed without sacrificing the ability to execute queries efficiently.

Providing secure and privacy-preserving data services over outsourced data is challenging. Both the database and the cryptography communities have shown great interest in providing privacy-preserving and secure data services, but there is no one scheme that solves all the security and privacy problems. Different schemes have different security and privacy guarantees, and these protection guarantees come at a cost: decrease in performance and functionality. There is an obvious trade-off between security/privacy and functionality/performance. Sacrificing functionality and performance completely for the sake of security and privacy makes outsourcing services impractical. Therefore, any data related service needs to seek a proper balance in the space of security, privacy, functionality and performance. In this tutorial, we aim to cover common security and privacy threats for outsourced data, and relevant state-of–the-art solutions from the database and the cryptography literature. We also discuss their limitations, open problems and further research directions for secure and private cloud storage systems. This tutorial explicitly focuses on the ability to query data in a cloud storage, while maintaining data confidentiality and access privacy.

II. Tutorial Outline

This tutorial presents recent schemes from both the database and the cryptography and security communities in the context of outsourced data in the cloud. In particular, we focus on two aspects of outsourced data in the cloud: query processing over encrypted data and access oblivious cloud storage systems. The tutorial consists of three main sections: 1) security and privacy threats for outsourced data, 2) query processing over encrypted data, and 3) access privacy for oblivious storage. The tutorial is intended to last 3 hours. The initial section highlights security and privacy concerns for outsourced data services. The next sections provide a broad survey of research in the area concerning security/privacy models, proposed techniques/schemes, and associated problems and challenges.

An earlier version of this tutorial was presented at EDBT 2017 [44].

A. Security and Privacy Threats in the Cloud

The cloud is a popular and tempting attack target. It hosts many businesses at different scales using a shared infrastructure. When an attacker attacks the cloud, it has access to consolidated data, which can have great financial value. To develop secure and privacy-preserving systems, the system designers must first develop a clear understanding of the possible threats. Therefore, the tutorial starts with a general overview of possible security and privacy threats in the context of storage services. The cloud service is assumed to be untrusted. Any unauthorized access or the cloud provider will be considered as an honest-but-curious *adversary*, where the adversary runs the protocol correctly, but may try to learn as much as possible about data. After highlighting possible security and privacy threats, to draw attention to the significance of the concerns, we will cover a few recent data breaches in terms of their vulnerabilities and consequences [1], [2]. Security and privacy are required, but performance and functionality are also essential for cloud storage systems and these conflict with security and privacy requirements. The question that concludes the section is "What is the proper balance between privacy, security, functionality, and performance?".

B. Query Processing over Encrypted Data

Storing encrypted data in a hostile environment provides strong data confidentiality. However, the ability to perform practical query processing on encrypted data remains a major challenge. Both the database and the cryptography research communities have shown great interest in querying encrypted data including keyword search [14], [48], equality queries [54], range queries [29], [31], and order preserving encryption [5], [38]. These methods sacrifice some degree of data confidentiality for more effective querying on encrypted data and provide different levels of security guarantees. Other proposals sacrifice query efficiency for stronger data confidentiality. Examples include homomorphic encryption and predicate encryption, which enable numerical computations on encrypted data without the need for decryption [22], [23], [35]. These have been shown to be quite expensive, and thus not practical [46].

Recent tutorials that appear in VLDB, ICDE and SIGMOD [3], [4], [7], [42] present detailed surveys of systems that perform query processing on encrypted data. In this tutorial, our approach is slightly different from these earlier works. We cover concepts that have seen significant interest recently in the security and the cryptography communities such as *Symmetric Searchable Encryption* (SSE). We revisit some important privacy and security concepts and cover important papers from the main security venues like S&P and CCS while still presenting recent results in the database community.

Initially, various primitive encryption schemes are introduced since they form the building blocks for other system developments. The functionality and security guarantees of non-deterministic and deterministic encryption scheme are presented using *Advanced Encryption Standard* (AES) [39]. Homomorphic encryption provides a desirable and interesting feature which allows computations directly over encrypted data. However, to date, only specific functionality, e.g. aggregation, can be performed efficiently. The need for different encryption schemes for specific tasks has resulted in various proposals such as order preserving encryption [5] and encrypted keyword search [48]. Both the database and the cryptography communities still show great interest in developing more efficient schemes for specific tasks.

Keyword search over encrypted data has received considerable attention in the cryptography and the security communities as well as the database community. Song et al. [48] propose a foundational technique for keyword search, also known as the first SSE scheme. This work has been followed upon by various competing new security definitions and constructions in the context of SSE [12], [16], [19], [24], [33], [40]. In this part of the tutorial, we start with [19] which provides security definitions for SSE for both adaptive and non-adaptive adversarial settings and proposes constructions for both adversarial settings. In recent work, Cash et al. [12] introduce a dynamic SSE solution which supports the modification of data. It supports storing large data and has optimal and parallelizable search complexity. Another dynamic SSE solution is proposed by Naveed et al. [40] and is based on a notion of *Blind Storage*. In an interesting study, Cash et al. also show that it is possible to extend the SSE approach to handle boolean queries in [13]. We discuss how such an extension might be a guide for further developments in different contexts.

Range queries are widely used as fundamental database operations to retrieve records between an upper and a lower boundary (e.g., retrieving students who have grades between A and B). A canonical SQL query for such a query is "select * from students where grade $\leq$ B and grade $\geq$ A". In spite of its wide utilization, performing range queries in a privacy-preserving manner is still challenging. Agrawal et al. introduce *order preserving encryption* (OPE) [5] to support range queries efficiently. Unfortunately, OPE is vulnerable to statistical attacks and is limited in terms of further modifications. Since it was first proposed, there have been a large number of proposals that aim to provide more secure solutions while still being efficient [10], [20], [30], [34], [36], [38]. Modular order preserving encryption (MOPE) [10] adds a secret offset to the data before encryption to shift the ciphertext (in a ring), and to hide the real location of the encrypted data in their distribution. In [38], an improved version of MOPE has been proposed. It uses fake queries over the gap between the maximum and minimum values to improve the security of MOPE against attacks that analyze the query patterns to detect the max/min values among the encrypted data. Improvements in SSE have also benefited the database community. Similar to [13], which handles boolean queries by extending SSE, Demertzis et al. [20] propose a range query solution that uses SSE. To take advantage of SSE, Demertzis et al. propose three types of indexing approaches with different space requirements in terms of domain size: quadratic, linear and logarithmic. We follow a different approach and propose PINED-RQ [43], a differentially private range query execution framework that constructs a differentially private index over an outsourced database. In a recent work, Fuhry et al. [21] propose a hardware-based approach for searching over encrypted data using Intel's SGX. We again discuss the proposed schemes in terms of their computational and space overheads, supported functionality, and security guarantees.

We finish this section of the tutorial by discussing full-fledged secure systems [6], [8], [41], [52]. CryptDB [41] is a secure system that processes different types of database queries using layers of different encryption mechanisms and removes layers of encryption to an appropriate layer for solving a specific query. MONOMI [52] follows CryptDB's approach of using different encryption schemes for specific queries. On the other hand, it is designed for executing analytical queries. Cipherbase [6] and TrustedDB [8] are full-fledged database system proposals that benefit from secure hardware. We discuss the advantages and disadvantages along with the security guarantees of these systems.

C. *Oblivious Storage*

Although it is necessary, encryption alone is not sufficient to solve all privacy challenges posed by the outsourcing of private data. Indeed, if *access patterns* are *not* hidden from the cloud provider, the provider could detect, for example, whether and when the same data item is repeatedly accessed, even if it does not learn the actual content of the item. This is a real threat to the privacy of outsourced data, as data access patterns can leak sensitive information using prior knowledge. For example, Islam et al. [32] showed a concrete inference attack against an encrypted e-mail repository exploiting access patterns alone. *Oblivious RAM* (ORAM) – a cryptographic primitive originally proposed by Goldreich and Ostrovsky [25], [26] as a solution for software protection – is the standard approach to make access patterns *oblivious*. ORAM shuffles and re-encrypts data in each data access, making access patterns from any two equally long sequences of read/write operations completely indistinguishable. Hiding access patterns was initially considered in the context of memory access [26]. While classical ORAM schemes with small client memory apply directly to the memory access setting, in cloud applications a client has more storage space and is capable of storing more data locally and more importantly can outsource the storage of a large dataset to the cloud. The novel features and fast adoption of the cloud gave impetus to the research community to develop new secure data services in the past several years and many ORAM schemes have been constructed for secure cloud storage systems [9], [11], [37], [45], [49], [50], [53]. Recent works from both the database and cryptography literature present a comprehensive analysis of ORAM schemes as oblivious cloud storage [9], [15], [45].

This section of the tutorial starts with the definition of access patterns. We explicitly define the notion of securing an access pattern. This is followed by a famous attack by Islam et al. [32] that shows how the leakage of access patterns can be harmful to sensitive data. Why should we care about access patterns? Why do we need to achieve *oblivious access*? After the motivation, we move to the details of ORAM constructions, which ensure oblivious accesses. To date, two main types of ORAM constructions exist: *hierarchical* and *tree-based*. The first hierarchical ORAM to be discuss is GO-ORAM [26]. Follow-up hierarchical ORAM constructions improve different aspects of GO-ORAM such as reduced overhead and faster shuffling [27], [28]. Next, we cover the tree-based ORAM constructions which have been proposed relatively recently and extended in a large number of works [18], [47], [51]. Tree-based constructions organize the memory as a tree. The current state-of-the-art construction, Path ORAM [51], will be covered as a prototype of tree-based ORAMs. Both GO-ORAM and Path ORAM were designed for a single client and such systems do not fit the requirements of cloud deployments, since accesses to the storage are performed *sequentially*. Therefore, after explaining the building blocks of single client hierarchical and tree-based ORAMs, we will discuss how to construct ORAMs in such a way that they simulate real-world storage scenarios by inheriting features like multi-client concurrent access, asynchronicity, and, of course, security.

PrivateFS by Williams et al. [53] increases the throughput of storage by enabling parallel accesses to the storage. We present the PrivateFS framework and then focus on how it allows multiple clients to obliviously access data in parallel along with its limitations. Follow-up improvements for more practical oblivious storage schemes [9], [45], [49] will be considered in the context of system design, performance, correctness and security. Stefanov and Shi propose *ObliviStore* [49] which provides a definition for asynchronous ORAM and introduces a proxy based approach where the proxy mediates the communication between clients and the server. In a recent study, Bindschaedler et al. [9] present a subtle security issue in ObliviStore and propose a modular oblivious storage system, called *CURIOUS*. In our recent work [45], we show that the security definition used by both ObliviStore and CURIOUS does not capture asynchrony when multiple clients access storage concurrently in a realistic deployment scenario. We, therefore, propose TaoStore, a new *tree-based* ORAM scheme that processes client requests concurrently and asynchronously in a non-blocking fashion.

At the end of this section, we provide a detailed analysis of the current state of secure cloud storage, the open problems and challenges, and further research directions towards providing more practical oblivious cloud storage systems.

III. Intended Audience

This tutorial aims to provide a broad survey on data security and privacy, and is intended to be beneficial for anyone interested in data security and privacy. We intend to introduce to the database community state-of-the-art results from the security literature that are particularly relevant for databases. The tutorial is self-contained and does not require any prior knowledge about data security and privacy.

IV. Biography

Amr El Abbadi is a Professor of Computer Science at the University of California, Santa Barbara. He received his B. Eng. from Alexandria University, Egypt, and his Ph.D. from Cornell University. Prof. El Abbadi is an ACM Fellow, AAAS Fellow, and IEEE Fellow. He was Chair of the Computer Science Department at UCSB from 2007 to 2011. He has served as a journal editor for several database journals and as program chair for multiple database and distributed systems conferences. He currently serves on the executive committee of the IEEE Technical Committee on Data Engineering (TCDE) and was a board member of the VLDB Endowment from 2002 to 2008. In 2007, Prof. El Abbadi received the UCSB Senate Outstanding Mentorship Award for his excellence in mentoring graduate students. Most recently Prof. El Abbadi was the co-recipient of the Test of Time Award at EDBT/ICDT 2015.

Cetin Sahin is a Senior Developer at SAP Big Data Services. Cetin earned his B.Sc. in Computer Science from Bilkent University, Turkey in 2011, and masters and Ph.D. degrees in Computer Science from University of California, Santa Barbara in 2016 and 2017, respectively. His Ph.D. dissertation focuses on data security and privacy in the cloud. He was a summer research assistant at NEC Laboratories in 2013 and 2014.

Acknowledgment

This work is partly funded by NSF grants CNS-1528178 and CNS-1649469.

References

[1] Dropbox breach from 2012 comes back to haunt users. https://www.identityforce.com/blog/2016-data-breaches.

[2] Yahoo data breach: Almost 500 million affected. https://www.identityforce.com/blog/yahoo-data-breach-almost-500-million-affected.

[3] D. Agrawal, A. E. Abbadi, and S. Wang. Secure and privacy-preserving database services in the cloud. In *ICDE '13*, pages 1268–1271, 2013.

[4] D. Agrawal, A. El Abbadi, and S. Wang. Secure and privacy-preserving data services in the cloud: A data centric view. *VLDB Endow.*, 5(12):2028–2029, 2012.

[5] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. SIGMOD '04, pages 563–574. ACM, 2004.

[6] A. Arasu, S. Blanas, K. Eguro, R. Kaushik, D. Kossmann, R. Ramamurthy, and R. Venkatesan. Orthogonal security with cipherbase. In *CIDR '13*, 2013.

[7] A. Arasu, K. Eguro, R. Kaushik, and R. Ramamurthy. Querying encrypted data. SIGMOD '14, pages 1259–1261. ACM, 2014.

[8] S. Bajaj and R. Sion. Trusteddb: A trusted hardware based database with privacy and data confidentiality. SIGMOD '11, pages 205–216. ACM, 2011.

[9] V. Bindschaedler, M. Naveed, X. Pan, X. Wang, and Y. Huang. Practicing oblivious access on cloud storage: The gap, the fallacy, and the new way forward. CCS '15, pages 837–849. ACM, 2015.

[10] A. Boldyreva, N. Chenette, and A. O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. CRYPTO '11, pages 578–595. Springer-Verlag, 2011.

[11] D. Boneh, D. Mazieres, and R. A. Popa. Remote oblivious storage: Making oblivious ram practical. Technical report, MIT, 2011. MIT Tech-report: MIT-CSAIL-TR-2011-018.

[12] D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *NDSS '14*, 2014.

[13] D. Cash, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In *CRYPTO '13, Proceedings, Part I*, pages 353–373, 2013.

[14] Y.-C. Chang and M. Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *ACNS '05, Proceedings*, pages 442–455, 2005.

[15] Z. Chang, D. Xie, and F. Li. Oblivious RAM: A dissection and experimental evaluation. *PVLDB*, 9(12):1113–1124, 2016.

[16] M. Chase and S. Kamara. Structured encryption and controlled disclosure. In *ASIACRYPT '10*, pages 577–594. Springer Berlin Heidelberg, 2010.

[17] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling data in the cloud: Outsourcing computation without outsourcing control. CCSW '09, pages 85–90. ACM, 2009.

[18] K.-M. Chung, Z. Liu, and R. Pass. Statistically-secure ORAM with $\tilde{O}(\log^2(n))$ overhead. In *ASIACRYPT '14, Proceedings, Part II*, pages 62–81. Springer Berlin Heidelberg, 2014.

[19] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. CCS '06, pages 79–88. ACM, 2006.

[20] I. Demertzis, S. Papadopoulos, O. Papapetrou, A. Deligiannakis, and M. Garofalakis. Practical private range search revisited. SIGMOD '16, pages 185–198. ACM, 2016.

[21] B. Fuhry, R. Bahmani, F. Brasser, F. Hahn, F. Kerschbaum, and A. Sadeghi. Hardidx: Practical and secure index with SGX. In *Data and Applications Security and Privacy XXXI - 31st Annual IFIP WG 11.3 Conference, DBSec 2017, Philadelphia, PA, USA, July 19-21, 2017, Proceedings*, pages 386–408, 2017.

[22] T. Ge and S. Zdonik. Answering aggregation queries in a secure system model. VLDB '07, pages 519–530. VLDB Endowment, 2007.

[23] C. Gentry. Fully homomorphic encryption using ideal lattices. STOC '09, pages 169–178. ACM, 2009.

[24] E.-J. Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. http://eprint.iacr.org/2003/216/.

[25] O. Goldreich. Towards a theory of software protection. In *CRYPTO '86*, pages 426–439. Springer-Verlag, 1987.

[26] O. Goldreich and R. Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, 1996.

[27] M. T. Goodrich, M. Mitzenmacher, O. Ohrimenko, and R. Tamassia. Oblivious storage with low I/O overhead. *CoRR*, abs/1110.1851, 2011.

[28] M. T. Goodrich, M. Mitzenmacher, O. Ohrimenko, and R. Tamassia. Practical oblivious storage. CODASPY '12, pages 13–24. ACM, 2012.

[29] H. Hacigümüş, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database-service-provider model. SIGMOD '02, pages 216–227. ACM, 2002.

[30] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu. Secure multidimensional range queries over outsourced data. *The VLDB Journal*, 21(3):333–358, 2012.

[31] B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. VLDB '04, pages 720–731. VLDB Endowment, 2004.

[32] M. S. Islam, M. Kuzu, and M. Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *NDSS '12*, 2012.

[33] S. Kamara, C. Papamanthou, and T. Roeder. Dynamic searchable symmetric encryption. CCS '12, pages 965–976. ACM, 2012.

[34] P. Karras, A. Nikitin, M. Saad, R. Bhatt, D. Antyukhov, and S. Idreos. Adaptive indexing over encrypted numeric data. SIGMOD '16, pages 171–183. ACM, 2016.

[35] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. EUROCRYPT '08, pages 146–162. Springer-Verlag, 2008.

[36] R. Li, A. X. Liu, A. L. Wang, and B. Bruhadeshwar. Fast range query processing with strong privacy protection for cloud computing. *VLDB Endow.*, 7(14):1953–1964, 2014.

[37] J. R. Lorch, B. Parno, J. Mickens, M. Raykova, and J. Schiffman. Shroud: Ensuring private access to large-scale data in the data center. In *USENIX FAST '13*, pages 199–213. USENIX, 2013.

[38] C. Mavroforakis, N. Chenette, A. O'Neill, G. Kollios, and R. Canetti. Modular order-preserving encryption, revisited. SIGMOD '15, pages 763–777, 2015.

[39] National Institute of Standards and Technology. Advanced encryption standard (AES). Federal Information Processing Standards Publications - 197, November 2001.

[40] M. Naveed, M. Prabhakaran, and C. A. Gunter. Dynamic searchable encryption via blind storage. IEEE SP '14, pages 639–654, 2014.

[41] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. SOSP '11, pages 85–100. ACM, 2011.

[42] R. Ramamurthy, R. Kaushik, A. Arasu, and K. Eguro. Querying encrypted data. In *ICDE '13*, pages 1262–1263, 2013.

[43] C. Sahin, T. Allard, R. Akbarinia, A. El Abbadi, and E. Pacitti. A differentially private index for range query processing in clouds. In *ICDE '18*, 2018.

[44] C. Sahin and A. El Abbadi. Data security and privacy for outsourced data in the cloud. In *EDBT'17*, pages 606–609, 2017.

[45] C. Sahin, V. Zakhary, A. El Abbadi, H. Lin, and S. Tessaro. Taostore: Overcoming asynchronicity in oblivious data storage. In *IEEE SP '16*, pages 198–217, 2016.

[46] B. Schneier. Homomorphic encryption breakthrough, 2009. http://www.schneier.com/blog/archives/2009/07/homomorphic_enc.html.

[47] E. Shi, T. H. H. Chan, E. Stefanov, and M. Li. Oblivious ram with o((logn)3) worst-case cost. In *ASIACRYPT '11, Proceedings*, pages 197–214. Springer Berlin Heidelberg, 2011.

[48] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *IEEE SP '00*, pages 44–55, 2000.

[49] E. Stefanov and E. Shi. Oblivistore: High performance oblivious cloud storage. In *IEEE SP '13*, pages 253–267, 2013.

[50] E. Stefanov, E. Shi, and D. X. Song. Towards practical oblivious RAM. In *NDSS '12*, 2012.

[51] E. Stefanov, M. van Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu, and S. Devadas. Path oram: An extremely simple oblivious ram protocol. CCS '13, pages 299–310. ACM, 2013.

[52] S. Tu, M. F. Kaashoek, S. Madden, and N. Zeldovich. Processing analytical queries over encrypted data. *Proc. VLDB Endow.*, 6(5):289–300, 2013.

[53] P. Williams, R. Sion, and A. Tomescu. Privatefs: A parallel oblivious file system. CCS '12, pages 977–988. ACM, 2012.

[54] Z. Yang, S. Zhong, and R. N. Wright. Privacy-preserving queries on encrypted data. In *ESORICS*, pages 479–495, 2006.