Advances in Cryptography and Secure Hardware for Data Outsourcing

Shantanu Sharma, Anton Burtsev, and Sharad Mehrotra
University of California, Irvine, USA.

***Abstract*—Despite extensive research, secure outsourcing remains an open challenge. This tutorial focuses on recent advances in secure cloud-based data outsourcing based on cryptographic (encryption, secret-sharing, and multi-party computation (MPC)) and hardware-based approaches. We highlight the strengths and weaknesses of state-of-the-art techniques, and conclude that, while no single approach is likely to emerge as a silver bullet. Thus, the key is to merge different hardware and software techniques to work in conjunction using partitioned computing wherein a computation is split across different cryptographic techniques carefully, so as not to compromise security. We highlight some recent work in that direction.**

I. Introduction

With cloud emerging as the dominant computing platform, secure data outsourcing, originally described in [1], has remained a significant data management challenge. Initial work on secure computing focused extensively on encryption techniques to allow operations to execute on the cloud. Order-preserving encryption (OPE) [2], deterministic encryption (DET) [3], non-deterministic encryption (NDET) [4], homomorphic encryption [5], bucketization [1], searchable encryption [6], and searchable symmetric encryption [7] are example of encryption techniques. These techniques have resulted in several secure data processing systems. CryptDB [8], Monomi [9], CorrectDB [10], ZeroDB [11], and MrCrypt [12] are encryption-based systems. Microsoft Always Encrypted, Oracle 12c, Amazon Aurora [13], and MariaDB [14] are industrial secure encrypted databases. Some prior work explored specialized hardware, *e.g.*, FPGA, for implementing a secure execution environment with the objective that the secure hardware acts as the data owner's agent in an untrusted cloud (and can hence decrypt and compute on plaintext data). Such systems include TrustedDB [15] and Cipherbase [16]. Several tutorials have considered encryption-based techniques and presented in VLDB, SIGMOD, and ICDE. In VLDB 2007 [17], the tutorial focused on result verification and data confidentiality at the cloud. In ICDE 2013 [18] and SIGMOD 2014 [19], the tutorials focused on encryption techniques and an overview of Intel Software Guard Extensions (SGX) [20]. In ICDE 2013 [21], the tutorial focused on encrypted data processing and access-pattern-hiding techniques. In ICDE 2018 [22], the tutorial focused on ORAM-based data processing.

Over the decade, significant new directions of research in secure data outsourcing have emerged as a consequence of the changing software and hardware landscape. Academia and industries have started focusing on information-theoretically secure techniques that are unconditionally secure and independent of adversary's computational capabilities, and are, thus, quantum-safe. Encryption-based techniques, which form the bulk of the work on secure data outsourcing, are, in contrast, not secure against computationally powerful adversaries, as well as, not quantum-safe.[1]

Shamir's secret-sharing (SSS) [25] is a well-known information-theoretically secure protocol. Secret-sharing mechanisms also have applications in other areas such as Byzantine agreement, secure multiparty computations (MPC), and threshold cryptography, as discussed in [26]. Recently, several new types of secret-sharing techniques, prototypes, and systems have been evolved; for example, distributed point function [27], function secret-sharing [28], accumulating-automata [29], [30], Obscure [31], SMCQL [32], Conclave [33], Splinter [34], Pulsar [35], Jana [36], and others [37]–[39]. However, such solutions either do not support general SQL queries, overburden the database (DB) owner (by fully participating in a query execution), or reveal the identity of the qualified tuples (*i.e.*, access-patterns).

Another major mainstream direction of work is exploiting secure hardware for building secure databases. This is a result of the emergence of secure enclaves as exemplified by Intel SGX [20], which is now commodity hardware. SGX allows us to create a small trusted execution environment that is isolated and protected from the rest of the system. Systems such as Opaque [40], EnclaveDB [41], StealthDB [42], M2R [43], and VC3 [44] are build using SGX. However, these secure hardware-based systems still suffer from several side-channel attacks, such as cache-line [45]–[47], branch shadow [48], and others [49]–[51]. Work is ongoing in developing secure hardware that do not suffer from such attacks [52].

Despite extensive research on cryptographic techniques and secure hardware, today, there is no single secure system that can provide a completely trusted environment at the cloud while supporting most types of SQL queries. In fact, it is unlikely that such a fully secure approach will emerge anytime soon that fully addresses a general solution to secure data outsourcing. We maintain that practically secure solutions will need to interoperate across different systems/cryptographic solutions storing data in systems offering security properties. For instance, if data can be classified as sensitive and non-sensitive, non-sensitive data could be stored in clear text while sensitive data is stored encrypted. However, recent work has identified several challenges in supporting such partitioned

This material is based on research sponsored by DARPA under agreement number FA8750-16-2-0021. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. This work is partially supported by NSF grants 1527536 and 1545071. A. Burtsev work is partially supported by NSF grant 1817143.

[1] Google, with sufficient computational capabilities, broke SHA-1 [23]. Also, Google showed a quantum processor can solve a computation in few minutes, while the same computation should take more than 10,000 years on the current supercomputer [24].

2375-026X/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDE48307.2020.00173

computing when different cryptographic techniques are used in conjunction [53], [54]. Several recent techniques have explored such challenges that arise in partitioned computing, such as HybrEx [55], Sedic [56], Prometheus [57], Tagged-MapReduce [58], SEMROD [59], and [54], [60].

Tutorial outline. To set the background, the tutorial starts by describing existing encryption-based approaches and highlighting their limitations in terms of security and performance. We discuss, briefly, new searchable encryption techniques, such as PB-Tree [61], IB-Tree [62], and Seabed [63]. We, then, focus on secret-sharing, MPC, and secure hardware (Intel SGX)-based data processing techniques and systems built using those techniques. We observe that while mechanisms based on secret-sharing [25] are potentially more scalable, splitting data amongst multiple non-colluding clouds (an assumption that is not valid in a general setting) incurs significant communication overheads and can only support a limited set of selection and aggregation queries efficiently. Then, we show that the current SGX architectures suffer from side-channel attacks, due to which they are not secure as they are assumed to be. Also, we show that the current cryptographic- or SGX-based solutions cannot handle large-sized datasets, and hence, we discuss data-partitioning-based approaches that work on systems supporting different cryptographic techniques to securely inter-operate between them.

II. Cryptographic Secure Data Processing

Adversarial and security models for the Cloud. Cryptographic techniques are designed to deal with different types of adversarial public clouds, namely honest-but-curious (HBC) [64] and malicious clouds. An HBC adversarial cloud – which is considered widely in many cryptographic algorithms [1], [64]–[66] – stores an outsourced dataset without tampering, correctly computes assigned tasks, and returns answers; however, it may exploit side knowledge (*e.g.*, query execution, background knowledge, and the output size) to gain as much information as possible about the sensitive data. A malicious adversary may deviate from the algorithm and may execute a task that he/she wishes (*e.g.*, delete tuples from the relation). To ensure security in HBC and malicious adversarial models, IND-CKA and IND-CKA2 [7] are well-known security properties, respectively. Further, to deal with securely updating and inserting data, forward and backward security properties were introduced. In this tutorial, we discuss different adversarial models and security properties.

A. Secret-Sharing-based Data Outsourcing

Secret-Sharing was invented independently by Adi Shamir [25] and George Blakley [67] in 1979. In using secret-sharing, the database (DB) owner divides a secret value into c different fragments, called *shares*, and sends each share to a set of c non-communicating participants/servers. These servers cannot know the secret value until they collect $c' < c$ shares. Note that $c' \leq c$, where c is often taken to be larger than c' to tolerate malicious adversaries that may modify the value of their shares. We discuss additional assumptions about the adversarial model when using secret-sharing, *e.g.*, the adversary cannot collude with all (or possibly the majority of) the servers. Also, the adversary cannot eavesdrop on a majority of communication channels between the DB owner and the servers. Note that if the adversary could either collude with or successfully eavesdrop on the communication channels between the majority of servers and the DB owner, the secret-sharing technique will not apply. Order-preserving secret-sharing (OP-SS), introduced in [37], maintains the order of the values in secret-shares too. Due to maintaining the ordering of values, finding records with maximum/minimum values using OP-SS is trivial, while revealing ordering information to the adversary.

In 2006, Emekçi et al. [37] introduced the first work for data outsourcing using SSS and OP-SS for executing sum, maximum, and minimum queries. Another paper by Emekçi et al. [38] using OP-SS for aggregation queries requires the DB owner to retain each polynomial, which was used to create database shares. Like [38], [39] proposed a similar approach. [68] proposed SSS-based sum and average queries; however, they require the DB owner to retain tuple-ids of qualifying tuples. [29] introduced a novel approach for searching over the secret-shared data without taking help from the DB owner. [30] used the string-matching operation over the shares at the server by applying a MapReduce job. In short, these solutions offer a limited form of selection or aggregation queries, but overburden the DB owner (by storing enough data related to polynomials and fully participating in a query execution), are insecure due to OP-SS, or reveal access-patterns. Recent work, Obscure [31] eliminates all such limitations and provides a fully secure and efficient solution for implementing aggregation queries with selections. Obscure exploits OP-SS – while OP-SS, in itself, is not secure (it is prone to background knowledge attacks, for instance). The way Obscure cleverly uses OP-SS, it prevents such attacks by appropriately partitioning data, while still being able to exploit OP-SS for efficiency. SMCQL [32] and Conclave [33] are two systems that allow executing SQL queries among different DB owners, while pushing most parts of the computation in cleartext.

B. Secure Processor-based Data Outsourcing

Intel SGX. Recent versions of Intel CPUs introduced SGX, a collection of microarchitectural mechanisms aimed to protect third-party cloud applications from the software stack of an untrusted system. SGX allows us to create a small trusted execution environment that is isolated and protected from the rest of the system. For example, in the cloud, SGX protects the computation from the operating system controlled by the tenant, but vulnerable to numerous applications, system-level attacks, and the hypervisor controlled by the cloud. In addition to protecting against software attacks, SGX provides encryption of enclave's memory having code and data and integrity is protected by the CPU, when the data leaves the last level of the caching hierarchy. This protects SGX applications from hardware attacks like memory snooping. While running on the servers controlled by the cloud, from a trust point of view, SGX enclaves are effectively controlled by the client.

Shortcoming of the existing SGX architecture. Unfortunately, existing implementations of SGX are prone to a range of side-channel attacks that exploit one of the microarchitectural components of the CPU, *e.g.*, branch target buffers [48], [51], pattern-history table [69], caches [45]–[47], [49], [50], DRAM row buffer [70], page-tables [70], [71], page-fault exception handlers [72], [73], and speculative execution capabilities [74] to exfiltrate sensitive data and sometimes the entire memory of the enclave [74]. Page-fault attacks rely on the operating system that is under control of the attacker to exfiltrate sensitive data from the enclave by triggering page-faults and tracking enclave's memory accesses at the granularity of memory pages [72]. Branch shadowing attacks allow reconstructing control flow inside the enclave, and hence, secret data by monitoring all taken branches through a side-channel in the branch prediction unit [48]. Cache-based side-channel attacks allow attackers to trace all memory accesses within the enclave [45], [47]. Foreshadow [74] and Meltdown-type attacks on SGX allow complete access to enclave's memory and registers. Surprisingly, the side-channel attacks allow an attacker to reconstruct a significant fraction of the sensitive dataset, often gaining complete access to it in cleartext. To illustrate the power of side-channel attacks, we discuss several recent case studies that extract cryptographic keys, graphical images, and sensitive genomic data from the enclave in a practical and realistic scenario [45], [47].

Data Processing using SGX. Several recent systems emerged to provide secure data processing in the SGX environment. We provide a historical perspective on the problem starting with TrustedDB [15] and Cipherbase [16], the systems that were first to use secure hardware and initiated the field of secure hardware-based data processing. We then discuss recent systems, such as Opaque [40], EnclaveDB [41], StealthDB [42], M2R [43], and VC3 [44], that are built to leverage SGX. We provide an overview of types of queries these systems support, and discuss their advantages and limitations, and whether they provide practical security in the face of powerful side-channel attacks. We then discuss an ongoing micro-architectural work [52], [75] aimed to address side-channel attacks in the next generation of chips, and possible limitations of hardware approaches. Finally, we provide an overview of possible system-level defenses and algorithmic approaches, providing a secure environment with SGX.

C. Data Partitioning-based Outsourcing

Existing secure techniques do not scale to large-sized datasets. For example, on TPC-H LineItem table, executing a simple selection query took 1051 seconds on 1M rows using secret-sharing-based Jana [36] and 89 seconds using SGX-based Opaque [40] on 6M rows [54]. Partitioned computing deals with data partitioning into sensitive and non-sensitive data and, potentially, provides significant benefits by (*i*) avoiding (expensive) cryptographic operations on non-sensitive data, and, (*ii*) allowing query processing on non-sensitive data to exploit indices. Such indices (that cannot be easily supported alongside encryption-based mechanisms in a non-interactive setting) are a key mechanism for efficient query processing in traditional database systems. Partitioned computing over sensitive/non-sensitive data has been considered, especially, in hybrid clouds (*e.g.*, HybrEx [55], Sedic [56], Prometheus [57], Tagged-MapReduce [58], SEMROD [59], and [60]), where sensitive data stays at a private cloud and only cleartext non-sensitive data stays at a public cloud. However, [55]–[60] do not generalize to partitioned computing in the public cloud setting (where sensitive data is stored cryptographically secure and non-sensitive data resides in cleartext). Recent work [54] generalizes the partitioned computing approach at the cloud. We discuss a new security challenge due to simultaneous execution of queries on the encrypted (sensitive) dataset and on the plaintext (non-sensitive) datasets, and then, show the need to new security definition, called *partitioned data security*. We highlight the importance of partitioned computation at the cloud by illustrating selection query execution.

III. Goal of the Tutorial

Outcome, intended audience, and duration. This tutorial provides a survey on data security and privacy by introducing the state-of-the-art results from the security literature (especially, secret-sharing-, secure hardware-, and data partitioning-based techniques) that are particularly relevant for databases. Researchers, students, developers, practitioners interested in data security and privacy should be benefited. We cover content for different audiences, as: 10% beginner, 40% intermediate, 50% advanced. The duration will be 3 hours.

Open problems. We show that the existing data outsourcing techniques and systems are not enough, if we wish systems that simultaneously (*i*) are efficient and general enough (*i.e.*, support significant parts of SQL) to be practical, and (*ii*) offer provable security from the user's perspective. Below, we provide open questions in different directions:

Secret-Sharing Context. We may think about how SSS-based solutions can be developed for large-sized datasets, how they can support complex SQL queries, and how one can minimize the number of clouds while doing all operations at the cloud.

Secure Hardware Context. In the face of numerous side-channel attacks, naïve adoption of SGX does not provide any practical protection. Some of the challenges will be solved by future hardware [52], [75], but some will require algorithmic and compiler-level solutions to support efficient memory and branch-oblivious computations.

Partition Computation Context. To make partition computing more practical, we need to find ways to execute complex queries (*e.g.*, join and nested queries) using partitioned computing. Another open problem is in finding how one can classify the data into sensitive and non-sensitive data. Another open problem could be in finding which types of cryptographic techniques can be supported by partition computing.

IV. Biographies

Shantanu Sharma received his Ph.D. degree in Computer Science in 2016 from Ben-Gurion University, Israel. He obtained his Master of Technology (M.Tech.) degree in Computer Science from National Institute of Technology, Kurukshetra,

India, in 2011. He was awarded a gold medal for the first position in his M.Tech. degree. Currently, he is pursuing his Post Doc at the University of California, Irvine, USA, assisted by Prof. Sharad Mehrotra. His research interests include data security and privacy, building secure and privacy-preserving systems on sensor data for smart buildings, designing models for MapReduce computations, and distributed algorithms.

Anton Burtsev received the PhD degree in computer science from the University of Utah, in 2013. He is currently an Assistant Adjunct Professor in the Department of Computer Science, University of California, Irvine. Previously, he was a Research Assistant Professor at the University of Utah.

Sharad Mehrotra received the PhD degree in computer science from the University of Texas, Austin, in 1993. He is currently a professor in the Department of Computer Science, University of California, Irvine. Previously, he was a professor with the University of Illinois at Urbana Champaign. He has received numerous awards and honors, including the 2011 SIGMOD Best Paper Award, 2007 DASFAA Best Paper Award, SIGMOD test of time award, 2012, DASFAA ten year best paper awards for 2013 and 2014, 1998 CAREER Award from the US National Science Foundation (NSF), and ACM ICMR best paper award for 2013.

REFERENCES

[1] H. Hacigümüs *et al.*, "Executing SQL over encrypted data in the database-service-provider model," in *SIGMOD*, pp. 216–227, 2002.
[2] R. Agrawal *et al.*, "Order-preserving encryption for numeric data," in *SIGMOD*, pp. 563–574, 2004.
[3] M. Bellare *et al.*, "Deterministic and efficiently searchable encryption," in *CRYPTO*, pp. 535–552, 2007.
[4] S. Goldwasser *et al.*, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.
[5] C. Gentry, *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.
[6] D. X. Song *et al.*, "Practical techniques for searches on encrypted data," in *IEEE SP*, pp. 44–55, 2000.
[7] R. Curtmola *et al.*, "Searchable symmetric encryption: Improved definitions and efficient constructions," *JCS*, vol. 19, pp. 895–934, 2011.
[8] R. A. Popa *et al.*, "CryptDB: processing queries on an encrypted database," *Commun. ACM*, vol. 55, no. 9, pp. 103–111, 2012.
[9] S. Tu *et al.*, "Processing analytical queries over encrypted data," *PVLDB*, vol. 6, no. 5, pp. 289–300, 2013.
[10] S. Bajaj *et al.*, "Correctdb: SQL engine with practical query authentication," *PVLDB*, vol. 6, no. 7, pp. 529–540, 2013.
[11] M. Egorov and M. Wilkison, "ZeroDB white paper," *CoRR*, vol. abs/1602.07168, 2016.
[12] S. D. Tetali *et al.*, "MrCrypt: static analysis for secure cloud computations," in *OOPSLA*, pp. 271–286, 2013.
[13] Amazon Aurora, available at:https://aws.amazon.com/rds/aurora/.
[14] MariaDB, available at:https://mariadb.com/.
[15] S. Bajaj *et al.*, "TrustedDB: A trusted hardware-based database with privacy and data confidentiality," *IEEE TKDE*, vol. 26, no. 3.
[16] A. Arasu *et al.*, "Orthogonal security with cipherbase," in *CIDR*, 2013.
[17] R. Sion, "Secure data outsourcing," in *VLDB*, pp. 1431–1432, 2007.
[18] A. Arasu *et al.*, "Querying encrypted data," in *ICDE*, 2013.
[19] A. Arasu *et al.*, "Querying encrypted data," in *SIGMOD*, 2014.
[20] V. Costan *et al.*, "Intel SGX explained," *IACR ePrint Archive*, 2016.
[21] D. Agrawal *et al.*, "Secure and privacy-preserving database services in the cloud," in *ICDE*, pp. 1268–1271, 2013.
[22] C. Sahin *et al.*, "Data security and privacy for outsourced data in the cloud," in *ICDE*, pp. 1731–1734, 2018.
[23] https://shattered.io/.
[24] https://tinyurl.com/wxuaaaz.
[25] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, 1979.
[26] A. Beimel, "Secret-sharing schemes: A survey," in *IWCC*, 2011.
[27] N. Gilboa and Y. Ishai, "Distributed point functions and their applications," in *EUROCRYPT*, pp. 640–658, 2014.
[28] E. Boyle, N. Gilboa, and Y. Ishai, "Function secret sharing," in *EUROCRYPT*, pp. 337–367, 2015.
[29] S. Dolev *et al.*, "Accumulating automata and cascaded equations automata for communicationless information theoretically secure multi-party computation," *TCS*, vol. 795, pp. 81 – 99, 2019.
[30] S. Dolev *et al.*, "Privacy-preserving secret shared computations using MapReduce," *IEEE TDSC*, 2019.
[31] P. Gupta *et al.*, "Obscure: Information-theoretic oblivious and verifiable aggregation queries," *PVLDB*, vol. 12, no. 9, pp. 1030–1043, 2019.
[32] J. Bater *et al.*, "SMCQL: secure query processing for private data networks," *PVLDB*, vol. 10, no. 6, pp. 673–684, 2017.
[33] N. Volgushev *et al.*, "Conclave: secure multi-party computation on big data," in *EuroSys*, pp. 3:1–3:18, 2019.
[34] F. Wang *et al.*, "Splinter: Practical private queries on public data," in *NSDI*, pp. 299–313, 2017.
[35] Stealth Pulsar, available at:http://www.stealthsoftwareinc.com/.
[36] D. W. Archer *et al.*, "From keys to databases - real-world applications of secure multi-party computation," *IACR Cryptology ePrint*, 2018.
[37] F. Emekçi *et al.*, "Privacy preserving query processing using third parties," in *ICDE*, p. 27, 2006.
[38] F. Emekçi *et al.*, "Dividing secrets to secure data outsourcing," *Inf. Sci.*, vol. 263, pp. 198–210, 2014.
[39] T. Xiang *et al.*, "Processing secure, verifiable and efficient SQL over outsourced database," *Inf. Sci.*, vol. 348, pp. 163–178, 2016.
[40] W. Zheng *et al.*, "Opaque: An oblivious and encrypted distributed analytics platform," in *NSDI*, pp. 283–298, 2017.
[41] C. Priebe *et al.*, "Enclavedb: A secure database using SGX," in *IEEE SP*, pp. 264–278, 2018.
[42] D. Vinayagamurthy *et al.*, "StealthDB: a scalable encrypted database with full SQL query support," *PoPETs*, pp. 370–388, 2019.
[43] T. T. A. Dinh *et al.*, "M2R: enabling stronger privacy in mapreduce computation," in *USENIX*, pp. 447–462, 2015.
[44] F. Schuster *et al.*, "VC3: trustworthy data analytics in the cloud using SGX," in *IEEE SP*, pp. 38–54, 2015.
[45] F. Brasser *et al.*, "Software grand exposure: SGX cache attacks are practical," in *WOOT*, pp. 11–11, 2017.
[46] J. Götzfried *et al.*, "Cache attacks on Intel SGX," in *EuroSec*, p. 2, 2017.
[47] A. Moghimi *et al.*, "Cachezoom: How SGX amplifies the power of cache attacks," in *CHES*, pp. 69–90, 2017.
[48] S. Lee *et al.*, "Inferring fine-grained control flow inside SGX enclaves with branch shadowing," in *USENIX Security*, pp. 557–574, 2017.
[49] M. Hähnel *et al.*, "High-resolution side channels for untrusted operating systems," in *USENIX ATC*, 2017.
[50] M. Schwarz *et al.*, "Malware guard extension: Using sgx to conceal cache attacks," in *DIMVA*, pp. 3–24, 2017.
[51] G. Chen *et al.*, "SgxPectre attacks: Stealing intel secrets from SGX enclaves via speculative execution," *arXiv preprint*, 2018.
[52] V. Costan *et al.*, "Sanctum: Minimal hardware extensions for strong software isolation," in *USENIX*, pp. 857–874, 2016.
[53] M. Naveed *et al.*, "Inference attacks on property-preserving encrypted databases," in *CCS*, pp. 644–655, 2015.
[54] S. Mehrotra *et al.*, "Partitioned data security on outsourced sensitive and non-sensitive data," in *ICDE*, pp. 650–661, 2019.
[55] S. Y. Ko *et al.*, "The hybrex model for confidentiality and privacy in cloud computing," in *HotCloud*, 2011.
[56] K. Zhang *et al.*, "Sedic: privacy-aware data intensive computing on hybrid clouds," in *CCS*, pp. 515–526, 2011.
[57] Z. Zhou *et al.*, "Prometheus: Privacy-aware data retrieval on hybrid cloud," in *INFOCOM*, pp. 2643–2651, 2013.
[58] C. Zhang *et al.*, "Tagged-MapReduce: A general framework for secure computing with mixed-sensitivity data on hybrid clouds," 2014.
[59] K. Y. Oktay *et al.*, "SEMROD: secure and efficient MapReduce over hybrid clouds," in *SIGMOD*, pp. 153–166, 2015.
[60] K. Y. Oktay *et al.*, "Secure and efficient query processing over hybrid clouds," in *ICDE*, pp. 733–744, 2017.
[61] R. Li *et al.*, "Fast range query processing with strong privacy protection for cloud computing," *PVLDB*, vol. 7, no. 14, pp. 1953–1964, 2014.
[62] R. Li *et al.*, "Adaptively secure conjunctive query processing over encrypted data for cloud computing," in *ICDE*, pp. 697–708, 2017.
[63] A. Papadimitriou *et al.*, "Big data analytics over encrypted datasets with seabed," in *OSDI*, pp. 587–602, 2016.
[64] R. Canetti *et al.*, "Adaptively secure multi-party computation," in *STOC*, pp. 639–648, 1996.
[65] C. Wang *et al.*, "Secure ranked keyword search over encrypted cloud data," in *ICDCS*, pp. 253–262, 2010.
[66] S. Yu *et al.*, "Attribute based data sharing with attribute revocation," in *ASIACCS*, pp. 261–270, 2010.
[67] G. R. Blakley *et al.*, "Safeguarding cryptographic keys," in *Proceedings of the national computer conference*, vol. 48, 1979.
[68] B. Thompson *et al.*, "Privacy-preserving computation and verification of aggregate queries on outsourced databases," in *PETS*, 2009.
[69] D. O'Keeffe *et al.*, "Spectre attack against SGX enclave," 2018.
[70] W. Wang *et al.*, "Leaky cauldron on the dark land: Understanding memory side-channel hazards in sgx," in *CCS*, pp. 2421–2434, 2017.
[71] J. Van Bulck *et al.*, "Telling your secrets without page faults: stealthy page table-based attacks on enclaved execution," in *USENIX*, 2017.
[72] Y. Xu *et al.*, "Controlled-channel attacks: Deterministic side channels for untrusted operating systems," 2015.
[73] S. Shinde and other, "Preventing page faults from telling your secrets," in *CCS*, pp. 317–328, 2016.
[74] J. V. Bulck *et al.*, "Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution," in *USENIX*, 2018.
[75] J. Yu *et al.*, "Data oblivious ISA extensions for side channel-resistant and high performance computing," in *NDSS*, 2019.