Lecture 1: Overview

Contents

1 Motivation

As we produce and consume increasing amounts of data, we are witnessing several conflicting trends. On one hand, these datasets are becoming more intrusive and privacy-sensitive and, on the other, they are becoming harder to secure.

Data breaches. This is well-illustrated by the constant occurrence of data breaches. In 2016, the Democratic National Committee (DNC) was hacked and in July Wikileaks published $19,252$ emails and $8,034$ attachments from the DNC. This led to the resignation of DNC chair Debbie Wasserman-Schultz [18]. Recently, 191M voter records were leaked online including voter names, addresses and party affiliations [14]. In 2015, 104K tax records were stolen from the IRS [13]. In 2014, the office of personnel management (OPM) was hacked and the records of government employees including the personal files of employees seeking top secret clearance were stolen [17]. The records of the latter included information about their past activities, their families and friends. In 2015, Ashley Madison—a website that facilitates extra-marital affairs—lost the personal records of each of 37M users. The breach has resulted in numerous divorces, political scandals and even suicides. These are just a few examples of recent high-profile data breaches but there are many more including from banks, tech companies, retail companies, government agencies from all around the world.

The Snowden disclosures. In 2013, Edward Snowden leaked classified information from the National Security Agency (NSA) to journalists Glenn Greenwald and Laura Poitras. These disclosures revealed the existence of a large number of NSA surveillance programs ; often in cooperation with other agencies from Australia, Canada, New Zealand and the United Kingdom. The programs were varied and included bulk, international and, most surprising, domestic surveillance.

End-to-end encryption and zero-knowledge systems. The prevalence of data breaches and the existence of a global and pervasive surveillance infrastructure points to the fact that the data we produce—and is produced about us—is not properly secured. While systems *sometimes* encrypt data in transit and at rest, there are still many stages in our data's lifetime where it remains unencrypted. Webmail, for example, can be encrypted from your browser to your webmail provider and then encrypted while at rest in your provider's backend storage. But it remains in plaintext on your provider's servers when your mailbox is accessed. Clearly, the way we currently integrate cryptography into systems is not good enough.

An alternative way of deploying cryptography is sometimes referred to as *end-to-end* encryption. In this approach, the data is encrypted *by the user* before it even leaves its device. The encryption keys are managed by the user so its data can never appear in plaintext anywhere unless it has explicitly been decrypted by its owner. Systems and services based on end-to-end encryption are sometimes called *zero-knowledge* systems [1] because they can operate with little to no information about the data they consume. These systems can provide much stronger privacy and security guarantees than the current generation of systems. The tradeoff, however, is that users are burdened with key management and all the problems this entails.

The main challenge in building zero-knowledge systems is that end-to-end encryption breaks many of the applications and services we rely on, including cloud computing, analytics, spam filtering, databases and of course search. A zero-knowledge webmail service would be unusable if one did not have the ability to search over one's encrypted emails. An application backed by an end-to-end encrypted database could not function.

Encrypted search. The area of *computing on encrypted data* aims to address some of the challenges posed by end-to-end encryption by producing new cryptosystems that support various forms of operations on encrypted data. A particularly important problem in this area is *encrypted search*; that is, the problem of searching on encrypted data. Search is one the most basic computational operations and, since the 90's, is arguably the most important functionality in information technology. Clearly, Web, enterprise and desktop search are core operations in both consumer and enterprise settings. Databases are integral to almost any system and, today, search functionality is embedded in almost every application.

Because of this, encrypted search could impact a wide variety of information systems. It would be an immediately applicable and, in fact *necessary*, component of most zero-knowledge services. For example, it could be used to enhance end-to-end encrypted cloud storage (e.g., Dropbox, OneDrive, iCloud), email (e.g., GMail, Outlook.com) and chat services (e.g., Signal, Slack, Skype, iMessenger) with private search capabilities. It could also be used in non-end-to-end settings, e.g., to add search to the encrypted back-end systems of cloud providers. It could also be used to support queries over databases that remain encrypted even in memory. All these applications would have a positive impact on the privacy and security of consumers and enterprises.

This course. In this course we will study the problem of encrypted search and the various solutions that have been proposed. We will learn how to formalize the security of these solutions and study state-of-the-art constructions. We will study their limitations and learn

[1] The term here is borrowed from the notion of zero-knowledge proofs (ZKPs) which are proofs that reveal at most the validity of a statement. Keep in mind that, unlike ZKPs, zero-knowledge *systems* are not formally defined and the usage of the term zero-knowledge here is more for marketing purposes.

how to attack various solutions. We will study how encrypted search solutions can and should be integrated into larger systems and see examples from startup and established companies alike. We will also cover open problems and new research directions in the area.

2 Overview of the Solution Space

Encrypted search blends techniques and ideas from cryptography, data structures, algorithms, information retrieval and databases. Currently, the state-of-the-art constructions achieve different tradeoffs between security, efficiency and query expressiveness. This is mostly due to the underlying primitives they use which we summarize below.

Fully-homomorphic encryption (FHE). An FHE scheme [15, 9] is an encryption scheme that supports arbitrary computation. That is, given a set of encryptions $(\mathrm{ct}_1, \ldots, \mathrm{ct}_n)$ of messages $(m_1, \ldots, m_n)$, one can generate a ciphertext ct of $f(m_1, \ldots, m_n)$, for any efficiently computable function f. FHE provides a natural solution to any encrypted search problem: (1) encrypt the data; (2) to query the data, encrypt the query q and homomoprhically evaluate the arithmetic circuit that evaluates q on the data. This results in an encrypted answer to the query.

Oblivious RAM (ORAM). An ORAM [11] takes as input an array and pre-processes it in such a way that it can support read and write operations *obliviously*; that is, without disclosing which addresses are accessed. ORAM also provides a natural solution to any encrypted search problem: (1) store the data in an array and pre-process it with an ORAM; (2) to search, simulate a search algorithm and replace every read and write operation with an oblivious read and write operation. Note that this approach is a naive way of using ORAM for search; we will see better alternatives.

Property-preserving encryption (PPE). A PPE scheme [2, 1, 3, 5] is an encryption scheme that reveals a property of the plaintext. For example, an equality-preserving encryption scheme—usually referred to as a deterministic encryption (DtE) scheme—reveals the equality between two plaintexts [2]. Similarly, order-preserving encryption (OPE) [1, 3] and order-revealing encryption schemes (ORE) [5] reveal the order of two plaintexts.[2] Given PPE schemes, one can solve any encrypted search problem that relies on equality testing and comparisons. For example, a relational EDB can be constructed by encrypting the cells of each table with an appropriate PPE scheme (DtE for attributes that will be equality tested and OPE/ORE for attributes that will be compared). To query the EDB on a SQL query q, the query is broken down into a sequence of equality and comparison operations which

[2]The difference between OPE and ORE schemes are that with the latter, comparisons of two ciphertexts can be done with any arbitrary computation whereas the former require that they be done using standard numerical comparison.

are then be evaluated directly on the encrypted tables. A similar approach can be used for encrypted non-relational databases.

Functional encryption (FE). A FE scheme [6] encrypts a message m in such a way that, given a token for some function f and the ciphertext, one can compute $f(m)$ without learning anything beyond the result. A special case of FE is *identity-based* encryption (IBE) [4] in which messages have the form $\langle \mathsf{id}, m\rangle$, where id is an arbitrary bit string, and the functions have the form

$$f_{\mathsf{id}'}\left(\langle \mathsf{id}, m\rangle\right) = \begin{cases} m & \text{if } \mathsf{id} = \mathsf{id}'; \\ \perp & \text{if } \mathsf{id} \neq \mathsf{id}'. \end{cases} .$$

General-purpose FE [12]—which can handle any efficiently-computable function—can be used to solve any encrypted search problem: (1) encrypt the data; (2) to query the data on q, generate a token for the function f which evaluates the query on the data. This results in an plaintext answer to the query (unlike the FHE-based solution discussed above). IBE can be used to construct an encrypted search engine as follows. First, encrypt each document D in the document collection with a standard encryption scheme. Then associate each encrypted document with a set of IBE encryptions of the form $\langle w, \mathsf{true}\rangle$ for every word w in the document. To search for a keyword w', generate a token for the function $f_{w'}$ and try to decrypt each IBE ciphertext of every encrypted document. Return the encrypted documents for which at least one IBE ciphertext decrypted to true.

Structured encryption (StE). A StE scheme [7] encrypts a data structure in such a way that, given a token for a query to the structure, one can evaluate the query and learn at most some well-defined leakage of the structure and/or query. A special case of StE is *searchable symmetric encryption* (SSE) [16, 10, 8] which encrypts search structures like inverted indexes or search trees. SSE provides a natural solution to designing encrypted search engines: (1) generate a search structure for the data and encrypt it; (2) to search for a keyword w, generate a token for w and query the encrypted structure. Other forms of StE like graph encryption schemes [7] provide natural solutions to designing encrypted graph databases.

Secure multi-party computation (MPC). An MPC protocol allows n mutually distrustful parties to execute a computation on the union of their inputs without disclosing any information about their inputs to each other (beyond what can be inferred from the result). Like FHE, MPC is a general-purpose primitive in the sense that it can be used to securely evaluate any efficiently computable function. MPC provides a natural solution to any encrypted search solution: (1) encrypt the data using a standard (symmetric) encryption scheme; (2) to query the data, securely evaluate a function that takes as input the encrypted data, the key and the query, decrypts the data, searches for the query and outputs the result.

Private information retrieval (PIR). A PIR protocol allows a client to read an element from an array without revealing which location was read. PIR can be used to design encrypted search solutions as follows: (1) store the data in an array; (2) encrypt each item with a symmetric-key encryption scheme; (3) to search, simulate a search algorithm and replace every read operation with a PIR query. Similarly to how we used ORAM above, this approach is a naive way of using PIR for search.

Tradeoffs. Roughly speaking, when maximizing security and query expressiveness at the expense of efficiency, the best solutions are based on ORAM and FHE. When maximizing efficiency and query expressiveness at the expense of security, the best solutions are based on PPE. When maximizing security and efficiency at the expense of query expressiveness, the best solutions are based on STE.

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In *ACM SIGMOD International Conference on Management of Data*, pages 563–574, 2004.

[2] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, Heidelberg, August 2007.

[3] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 224–241. Springer, Heidelberg, April 2009.

[4] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.

[5] Dan Boneh, Kevin Lewi, Mariana Raykova, Amit Sahai, Mark Zhandry, and Joe Zimmerman. Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 563–594. Springer, Heidelberg, April 2015.

[6] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011.

[7] Melissa Chase and Seny Kamara. Structured encryption and controlled disclosure. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 577–594. Springer, Heidelberg, December 2010.

[8] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. In *ACM Conference on Computer and Communications Security (CCS '06)*, pages 79–88. ACM, 2006.

[9] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[10] E-J. Goh. Secure indexes. Technical Report 2003/216, IACR ePrint Cryptography Archive, 2003. See `http://eprint.iacr.org/2003/216`.

[11] O. Goldreich and R. Ostrovsky. Software protection and simulation on oblivious RAMs. *Journal of the ACM*, 43(3):431–473, 1996.

[12] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 536–553. Springer, Heidelberg, August 2013.

[13] CNN Money. Criminals use irs website to steal data on 104,000 people. See `http://money.cnn.com/2015/05/26/pf/taxes/irs-website-data-hack/index.html`.

[14] Reuters. Database of 191 million u.s. voters exposed on internet: researcher. See `http://uk.reuters.com/article/us-usa-voters-breach-idUKKBN0UB1E020151229`.

[15] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–180, 1978.

[16] D. Song, D. Wagner, and A. Perrig. Practical techniques for searching on encrypted data. In *IEEE Symposium on Research in Security and Privacy*, pages 44–55. IEEE Computer Society, 2000.

[17] The state of security. The opm breach: Timeline of a hack. See `http://www.tripwire.com/state-of-security/security-data-protection/cyber-security/the-opm-breach-timeline-of-a-hack/`.

[18] Wikipedia. 2016 democratic national committee email leak. See `https://en.wikipedia.org/wiki/2016_Democratic_National_Committee_email_leak`.