Outsourced Bits

A research blog on cloud computing, cryptography, security, privacy, …

How to Search on Encrypted Data: Introduction (Part 1)

This entry was posted on October 6, 2013, in Crypto Design, Encrypted Search, Privacy and tagged cloud storage, encrypted search, privacy, search on encrypted data, searchable encryption. Bookmark the permalink. 9 Comments

(https://cloudcrypto.files.wordpress.com/2013/10/sse.jpg)

This is the first part of a series on searching on encrypted data. See parts 2 (http://outsourcedbits.org/2013/10/14/how-to-search-on-encrypted-data-part-2/), 3 (http://outsourcedbits.org/2013/10/30/how-to-search-on-encrypted-data-part-3/) and 4 (http://outsourcedbits.org/2013/12/20/how-to-search-on-encrypted-data-part-4-oblivious-rams/).

I recently finished giving a series of talks on one of my favorite topics: *searching on encrypted data*.

My slides are available here (http://research.microsoft.com/en-us/um/people/senyk/slides/encryptedsearch-full.pdf), but given the current interest in this topic I thought it might be useful to turn the talk into a series of posts.

Over the years, the problem of encrypted search has become an important problem in security and cryptography. I think this is due to a combination of three things: (1) search is now the primary way we access our data; (2) we are outsourcing more and more of our data to third parties; and (3) we trust these third parties less and less (for obvious reasons!).

Because of this, the problem of encrypted search is now of interest to many sub-fields in computer science (e.g., databases, security, cryptography, privacy) but also to industry and to governments.

While a-priori searching on encrypted data may sound impossible and even contradictory, we know of many ways to do it. Some methods are more practical than others, some are more secure than others and some are more functional/flexible.

Some History

The problem of searching on encrypted data was first considered explicitly by Song, Wagner and Perrig in this paper (http://www.cs.berkeley.edu/~dawnsong/papers/se.pdf) from 2001. When I first read this paper as a graduate student, I thought this was amazing! I had never imagined one could do anything like this with encryption and, amazingly, the construction even seemed practical. Combined with the obvious sense that these "searchable encryption" schemes could have a huge impact, I got really excited about this problem.

I later understood that perhaps I shouldn't have been so surprised (but no less impressed!) about the *possibility* of searching on encrypted data since prior work on oblivious RAMs (http://www.cs.ucla.edu/~rafail/PUBLIC/09JACM.pdf) by Goldreich and Ostrovsky and on secure two-party computation (http://en.wikipedia.org/wiki/Secure_two-party_computation) by Yao also provided solutions to this problem—albeit a lot less efficiently.

It's been about 10 years since the Song, Wagner and Perrig paper, 20 years since the Goldreich and Ostrovsky paper and 30 years since Yao's paper. So where do we stand with respect to encrypted search?

The short answer is that we've made a ton of progress and we' re at the point where encrypted search is reasonably well understood [1], efficient and could be deployed. Of course, like any cryptographic technology there are tradeoffs one has to consider and one of the goals of this series will be to make these tradeoffs clear.

The Setup

To structure our discussion and to allow for comparisons of the different solutions, we'll assume the following setting. We have two parties: a client and a server; that interact in two phases: a setup phase and a search phase.

During the setup phase, the client will take n documents $(D_1, \ldots, D_n)$ and generate:

1. an encrypted database (EDB) [2],
2. a set of encrypted documents $(c_1, \ldots, c_n)$.

We'll assume the documents are encrypted using a standard encryption scheme (e.g., AES using your favorite mode of operation) so the interesting part will be how exactly the EDB is constructed.

So as a concrete example if the documents are an email collection, the client will generate an EDB and then encrypt each email separately using, say, AES. After generating the EDB and encrypted documents, it will send everything to the server, concluding the setup phase.

During the search phase, the client wants the server to send back all the encrypted documents associated with some keyword w . To do this, the client will send a *token* that encapsulates w without revealing information about it [3]. The server will then use the token with the EDB to somehow figure out which encrypted documents it should send back.

Conclusions

We know of six different ways to search on encrypted data, each based on one of the following cryptographic primitives:

- property-preserving encryption (http://outsourcedbits.org/2013/10/14/how-to-search-on-encrypted-data-part-2/)
- functional encryption (http://outsourcedbits.org/2013/10/30/how-to-search-on-encrypted-data-part-3/)
- fully-homomorphic encryption
- searchable symmetric encryption
- oblivious RAMs
- secure two-party computation

In the rest of this series we' ll go through some of these approaches and see the various tradeoffs they provide between efficiency, security and functionality.

Notes

[1] There are some aspects of this problem that we don't fully understand yet; in particular the tradeoffs between efficiency and security. But I'll come back to this in later posts.

[2] Though I'll use the term database here, this should not necessarily be interpreted as a SQL database. In the cryptography literature the term database is often used to refer to some data structure that allows for some form of search (even very basic search).

[3] As we'll see in future posts, depending on the approach we take, the tokens can sometime reveal some information about the keyword.

9 thoughts on "How to Search on Encrypted Data: Introduction (Part 1)"

1. Pingback: How to Search on Encrypted Data (Part 2) | Outsourced Bits

2. Pingback: How to Search on Encrypted Data: Functional Encryption (Part 3) | Outsourced Bits

3. Pingback: How to Search on Encrypted Data (Part 4): Oblivious RAMs | Outsourced Bits

4. Pingback: How to Search on Encrypted Data | Outsourced Bits | Backfill for 'Note to Self'

5. Pingback: How to Search on Encrypted Data: Searchable Symmetric Encryption (Part 5) | Outsourced Bits

6. Pingback: High level design for secure web application | DL-UAT

7. Pingback: Bitcoin Faucet Rotator Blog How to Search on Encrypted Data, in Practice

8. Pingback: Applied Crypto Highlights: Searchable Encryption with Ranked Results | Outsourced Bits

9. **Jay Patil says:**
   January 26, 2016 at 9:05 pm
   can you provide the all codes for "Secure searching using modern cryptographic tools" which includes of DES,Inverted Index,Hashing(SHA1),key Generation,Searching all this algorithm and code?

   Reply