Future Generation Computer Systems 62 (2016) 104–118

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Entangled cloud storage

Giuseppe Ateniese[a], Özgür Dagdelen[b], Ivan Damgård[c], Daniele Venturi[d,*]

[a] Department of Computer Science, Stevens Institute of Technology, 1 Castle Point on Hudson, Hoboken, NJ 07030-5991, USA
[b] BridgingIT, N7 5, 68161 Mannheim, Germany
[c] Department of Computer Science, Aarhus University, IT-Parken, Åbogade 34, DK-8200 Aarhus N, Denmark
[d] Department of Computer Science, Sapienza University of Rome, Via Salaria 113, 00198 Rome, Italy

HIGHLIGHTS

- We model simulation-based security for data entanglement in the cloud.
- We introduce stronger security notions for entangled encoding schemes.
- We give a protocol for entangled storage satisfying our simulation-based definition.

ARTICLE INFO

Article history:
Received 29 April 2015
Received in revised form
12 November 2015
Accepted 13 January 2016
Available online 28 January 2016

Keywords:
Cloud storage
Data entanglement
Simulation-based security
Universal composability

ABSTRACT

Entangled cloud storage (Aspnes et al., ESORICS 2004) enables a set of clients to "entangle" their files into a single *clew* to be stored by a (potentially malicious) cloud provider. The entanglement makes it impossible to modify or delete significant part of the clew without affecting *all* files encoded in the clew. A clew keeps the files in it private but still lets each client recover his own data by interacting with the cloud provider; no cooperation from other clients is needed. At the same time, the cloud provider is discouraged from altering or overwriting any significant part of the clew as this will imply that none of the clients can recover their files.

We put forward the first simulation-based security definition for entangled cloud storage, in the framework of *universal composability* (Canetti, 2001). We then construct a protocol satisfying our security definition, relying on an *entangled encoding scheme* based on privacy-preserving polynomial interpolation; entangled encodings were originally proposed by Aspnes et al. as useful tools for the purpose of data entanglement. As a contribution of independent interest we revisit the security notions for entangled encodings, putting forward stronger definitions than previous work (that for instance did not consider collusion between clients and the cloud provider).

Protocols for entangled cloud storage find application in the cloud setting, where clients store their files on a remote server and need to be ensured that the cloud provider will not modify or delete their data illegitimately. Current solutions, e.g., based on Provable Data Possession and Proof of Retrievability, require the server to be challenged regularly to provide evidence that the clients' files are stored *at a given time*. Entangled cloud storage provides an alternative approach where any single client operates implicitly on behalf of all others, i.e., as long as one client's files are intact, the entire remote database continues to be safe and unblemished.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

Background. Due to the constantly increasing need of computing resources, and to the advances in networking technologies, modern IT organizations nowadays are prompted to outsource their storage and computing needs. This paradigm shift – often known as "cloud computing" – allows for applications from a server to be executed and managed through a client's web browser, with no installed client version of an application required. Cloud computing includes different types of services, the most prominent known under the name of *Infrastructure as a Service* (IaaS), *Platform as a service* (PaaS), and *Software as a Service* (SaaS). In rough terms, a solution at the SaaS level allows a customer (e.g., the end-user) to make use of a service provider's computing, storage or

* Corresponding author.
E-mail address: venturi@di.uniroma1.it (D. Venturi).

http://dx.doi.org/10.1016/j.future.2016.01.008
0167-739X/© 2016 Elsevier B.V. All rights reserved.

networking infrastructure. A solution at the PaaS level, instead, allows a customer (e.g., a programmer or a software developer) to exploit pre-configured software environments and tools. Finally, a solution at the IaaS level allows a customer (e.g., a service provider) to acquire physical resources such as storage units, network devices and virtual machines.[1]

Cloud infrastructures can belong to one of two categories: *public* and *private* clouds. In a private cloud, the infrastructure is managed and owned by the customer and located in the customer's region of control. In a public cloud, on the contrary, the infrastructure is owned and managed by a cloud service provider and is located in the cloud service provider's region of control. The latter scenario poses serious security issues, due to the fact that a malicious cloud provider could misbehave putting the confidentiality of a customer data at edge.

Cloud storage. Cloud computing has generated new intriguing challenges for cryptographers. In this paper, we deal with the problem of *cloud storage*, where clients store their files on remote servers based on public clouds (e.g., via Microsoft's Azure or Amazon's S3). Outsourcing data storage provides customers with several benefits. In particular, by moving their data to the cloud, customers can avoid the costs of building and maintaining a private storage infrastructure; this results in improved availability (as data is accessible from anywhere) and reliability (as, e.g., customers do not need to take care of backups) at lower costs.

While the benefits of using a public cloud infrastructure are clear, companies and organizations (especially enterprises and government organizations) are still reluctant to outsource their storage needs. Files may contain sensitive information and cloud providers can misbehave. While encryption can help in this case, it is utterly powerless to prevent data corruption, whether intentional or caused by a malfunction. Indeed, it is reasonable to pose the following questions: How can we be certain the cloud provider is storing the entire file intact? What if rarely-accessed files are altered? What if the storage service provider experiences Byzantine failures and tries to hide data errors from the clients? Can we detect these changes and catch a misbehaving provider?

PDP/POR. It turns out that the questions above have been studied extensively in the last few years. Proof-of-storage schemes allow clients to verify that their remote files are still pristine even though they do not possess any local copy of these files. Two basic approaches have emerged: Provable Data Possession (PDP), introduced by Ateniese et al. [1], and Proof of Retrievability (POR), independently introduced by Juels and Kaliski [2] (building on a prior work by Naor and Rothblum [3]). They were later extended in several ways in [4–11]. In a PDP scheme, file blocks are signed by the clients via authentication tags. During an audit, the remote server is challenged and proves possession of randomly picked file blocks by returning a short proof of possession. The key point is that the response from the server is essentially constant, thanks to the homomorphic property of authentication tags that makes them *compressible* to fit into a short string. Any data alteration or deletion will be detected with high probability. In POR, in addition, error correction codes are included along with remote file blocks. Now, the server provides a proof that the entire file could potentially be recovered in case of hitches.

Data entanglement. The main shortcoming of proof-of-storage schemes is that a successful run of an audit provides evidence about the integrity of a remote file *only at a given time*. As a consequence, all users must challenge the storage server regularly to make sure their files are still intact.

An alternative approach has been proposed by Aspnes et al. [12], under the name of *data entanglement*.[2] The main idea is to make altering or deleting files extremely inconvenient for the cloud provider. To achieve this feature, the authors of [12] considered a setting where many clients encode all their files into a single digital *clew*[3] c, that can be used as a representation of all files and be stored on remote and untrusted servers. The goal is to ensure that any significant change to c is likely to disrupt the content of all files.

Unfortunately, the original model of [12] suffers from an important shortcoming: The entanglement is created by a trusted authority, and files can only be retrieved through the trusted authority. Although the assumption of a trusted party significantly simplifies the task of designing (and analyzing) protocols for data entanglement, it also makes such protocols not suitable for cloud computing.

1.1. Our contributions

The main contribution of this paper is to overcome the above limitation. In particular, we propose the first simulation-based definition of security for data entanglement as well as protocols satisfying our definition without the need for a trusted party. More in detail, our results and techniques are outlined below.

Entangled encodings. *Entangled encoding schemes* were introduced by [12] as useful tools for the purpose of data entanglement. As a first contribution, we revisit the notion of entangled encodings putting forward stronger definitions w.r.t. previous work (see below for a comparison). In our language, an entangled encoding consists of an algorithm Encode that takes as input n strings $f_1, \ldots, f_n$ (together with a certain amount of randomness $r_1, \ldots, r_n$), and outputs a single codeword c which "entangles" all the input strings. The encoding is efficiently decodable, i.e., there exists an efficient algorithm Decode that takes as input (c, r_i, i) and outputs the file f_i together with a verification value ξ. Since only r_i is required to retrieve f_i (we do not need $r_j, j \neq i$), we refer to this as "local decodability". The verification value is a fixed function of the encoded string and the randomness.

In addition, the encoding satisfies two main security properties. First off, it is *private* in the sense that even if an adversary already knows a subset of the input strings and randomness used to encode them, the resulting encoding reveals no additional information about any of the other input strings other than what can be derived from the knowledge of this subset. Second, it is *all-or-nothing* in the sense that whenever an adversary has "large" uncertainty about c (i.e., a number of bits linear in the security parameter), he cannot design a function that will answer any decoding query correctly. See Section 3 for a precise definition.

We remark that our definitions are stronger than the one considered in [12]. First, [12] did not considered privacy as an explicit property of entangled encodings. Second, and more importantly, our definition of all-or-nothing integrity is more general in that for instance it allows the adversary to known a subset of the input strings and randomness; in the cloud storage setting this will allow to model arbitrary collusion between clients and a malicious cloud provider.

We additionally provide a concrete instantiation of an entangled encoding scheme based on polynomials over a finite field

[1] While the brief description above tries to make a clear distinction between the IaaS, PaaS, and SaaS layers, such a distinction is not always easy to draw in practice.

[2] "Entanglement" usually refers to a physical interaction between two particles at the quantum level: Even if separated, the particles are in a quantum superposition until a measurement is made, in which case both particles assume definitive and correlated states. Analogously, two entangled files are somehow linked together: A file that is intact implies the other must also be intact. Any single change to one file, destroys the other.

[3] The terminology "clew" typically refers to a ball of yarn or string.

Download English Version:

https://daneshyari.com/en/article/425543

Download Persian Version:

https://daneshyari.com/article/425543

Daneshyari.com