Showcase of a Fragment-based Distributed Cloud Storage System

Tuan T. Tran†, Phani C. Polina‡, Xiaolong Tang†, Zhen Jia┼, Yi Yang=, Anup Kumar‡, and Bin Xie†
† InfoBeyond Technology LLC, Louisville, KY USA
‡Department of Computer Engineering and Computer Science, University of Louisville, Louisville, KY USA
┼ Department of Electrical Engineering, University of Cincinnati, Cincinnati, OH USA
= Department of Geography, SUNY Buffalo, NY USA
Email: {tuan, x.long, bin.xie}@infobeyonds.com; {p0poli01, ak}@louisville.edu; jiazhen6@gmail.com; yyang33@buffalo.edu

***Abstract*—We propose to demonstrate a prototype of fragment-based distributed cloud storage system. The prototype is implemented by using efficient encoding/decoding, multiple-layer encryption and spatial data distribution for data efficiency and security. We will demonstrate that the proposed prototype offers significant improvement of data protection, compared with the file-based storage system, on both data reliability and security. For example, we will show that how fragment-based user data is processed and distributed over the cloud, or from security aspect, we will show how the system copes with the cybersecurity attacks during which some of the storage nodes are compromised (e.g., all stored data is lost and storage nodes are inaccessible.) The demonstration is performed via a web-based interface on several mobile devices which remotely connect to our prototype via the Internet.**

***Index Terms*—Secure cloud storage, distributed storage, cybersecurity attack, fragment-based storage.**

I. Introduction

Today, most of the storage systems implement a file-based storage architecture in which each data file is stored as a whole into the system. Such an architecture offers a simple way for operating and managing the user data. Data is well organized and easily manipulated. For example, a data user can easily copy or delete a data file from the system. However, the architecture shows several issues on data security, e.g., stored data may be visible to all clients accessed to the system. More importantly, the conventional architecture is vulnerable to several cybersecurity attacks such as eavesdropping, interception, spyware, etc. As an example, file-based data transmission over the network can easily be eavesdropped as the information is transmitted continuously in a stream from the sender to the receiver. Although the data is encrypted and may be routed via multiple paths, obtaining only a portion of the transmitted data is enough to reveal some sensitive information. Severely, network attackers can compromise machines or routers and retrieve all stored or relayed information for extracting sensitive information. Also, as data is stored into a single machine, it is subject to the single-point of failure, i.e., data will be lost when the storage device is damaged or lost.

Different from the file-based storage architecture, we propose a fragment-based distributed cloud storage (FDCS) system. The proposed system protects user data not only from the cybersecurity attackers but also from the service providers by a combination of multi-layer data encryptions and spatial data distribution. Particularly, in the first layer, the whole data file is encrypted using a strong encryption key at the user side. Next, the encrypted file is sent to the system and it is then divided into several fragments and each of them is encrypted by a unique encryption key. Finally, the multi-layer encrypted fragments are secretly distributed into different storage nodes. Using this approach, the proposed system offers several superior capabilities on data protection:

- **Data reliability**: A data file can be recovered by accessing to any subset (specified by the coding scheme which is described in Section II) of the data fragments. Additionally, damage of any subset (specified by the coding scheme) of storage nodes won't affect to the data accessibility.
- **Data security**: The service provider knows nothing about the data stored in the cloud. On the other hand, it is very difficult to the attackers to reveal any useful information on the data content due to:
  - ▷ *Multi-layer encryption*: Data is protected by two-layer encryptions (file level and fragment level).
  - ▷ *Uncorrelated data distribution*: Data is never continuously stored or transmitted as any meaningful data streams.
  - ▷ *Secure Data Sharing*: Data sharing is securely performed as it is never transmitted from the sender to the receivers. Instead, the receivers directly connect to the cloud to retrieve the information.
  - ▷ *Spatially distributed data*: Data is secretly distributed among multiple storage nodes in the cloud where each of them stores a small number of encrypted fragments.
  - ▷ *Spatially distributed keys*: Decryption keys are divided and secretly distributed among the storage nodes.

However, there are many technical challenges we need to address before a prototype can be implemented. Some of them are:

- **Efficient encoding/decoding**: The user data can be very large, how to efficiently to encode and decode the information is a challenge. For example, what is the optimal code ratio minimizing encoding/decoding delay while satisfying a specified reliability threshold?
- **Optimal fragment size**: In the proposed approach, a data file will be divided into several fragments, what is the optimal fragment size for efficient encoding/decoding, encryption/decryption, and storage space usage while achieving high level of cyber security?
- **Optimal data distribution**: Further, data will be distributed over several storage nodes, what is the optimal data distribution? How many fragments should be stored on a storage node?
- **Optimal data regeneration**: Also, each storage node may fail at any time, how to detect and regenerate the data stored on it is a technical challenge we need to address.

The preliminary design and performance evaluation of the proposed system have been published in [1]–[3]. Significantly

system performance gains in data encoding/decoding, data reliability, system resiliency, system security, and end-to-end delays have been reported. In this demonstration, we show our prototype of the fragment-based distributed cloud storage system implemented on a cloud of 10 storage servers at InfoBeyond Technology LLC. The demonstration will be performed over the Internet via a web-based interface. Details of the system implementation and demonstration are described in Sections II and III, respectively.

II. The Proposed FDCS

In this section, we will describe in details how the prototype is implemented. The prototype architecture is shown in Fig. 1.

A. Registration Server

This server is implemented to perform user registration and system authentication. In the current prototype, we implemented authentication by using an ID and password combination. When authenticated, a user will have an *authentication certificate* indicating what kind of services can be accessed.

B. Data Gateway Server

Data gateway server is implemented to process the user data before it is stored into the system. It receives the encrypted data from the user and performs encoding and encryption (i.e., fragment encryption). In our prototype, we implemented the erasure code [4] for data encoding. Particularly, our code is specified by a tuple of three parameters (n, l, k), where n, l, and k, respectively, denote the number of storage nodes, data pieces, and the minimum number of coded fragments required for recovering a data piece. For example $(4, 2, 2)$ coding scheme implies that a data file is divided into $l = 2$ data pieces; each data piece is then equally divided into $k = 2$ fragments $c_{i,j}$, and these fragments are then combined together to generate $n = 4$ coded fragments for each data piece using the following formula:

$$c = \sum_{i \in [1,l]; j \in [1,k]} \alpha_{i,j} c_{i,j}, \tag{1}$$

where $\alpha_{i,j}$ is the coding coefficients randomly selected from a finite field such as GF_q (q is the field size). The coded fragments are then distributed into the storage nodes. Using our proposed (n, l, k) coding scheme, the data owner can still recover the original data for upto $n - k$ storage nodes failures.

Next, the encoded fragments will be encrypted by different sub-keys by using the Shamirs secret sharing algorithm [5]. The sub-keys and encrypted fragments are then distributed over the cloud storage nodes. The location of storage servers is found via an optimization process for system efficiency and security. In the prototype, the system computes the storage availability and link capacity to each storage server to decide where the coded fragments will be stored to reduce the effect of load unbalanced issues and end-to-end delay.

C. Monitor Server

To monitor the state of stored data, we implemented a monitor server which periodically checks the state of storage servers and data on them. When a server failure or data loss is detected, the monitor server will identify the lost data and locate a new storage server to replace the failed one. In our prototype, the new storage server is selected from the remaining server of the 10 storage servers. It then triggers a data regeneration function on the new storage server which actually connects to multiple storage servers having data of the lost file to download coded fragments, and regenerates new data to replace the lost fragments. In our implementation, the new generated fragments may not be the same as the lost data, but it satisfies the data recovery condition, i.e., any subset of k storage nodes is sufficient to recover the original data.

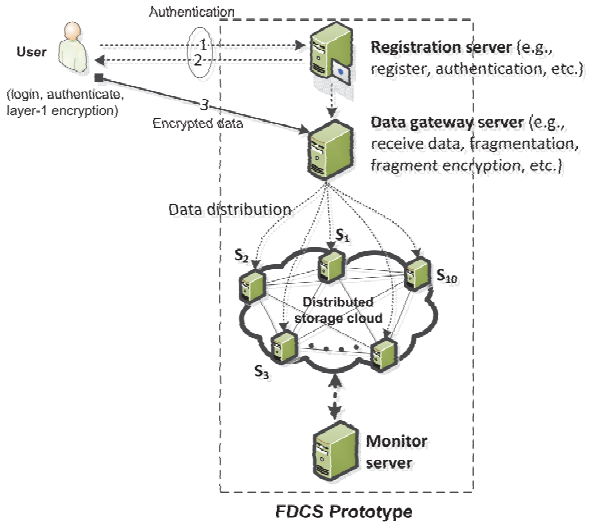

Fig. 1. The prototype architecture of FDCS. The solid and dash lines represent the control signals and data flow, respectively.

III. Objectives, Setup and Expected Results

A. Demo Objectives

The main objectives of the proposed demonstration are to show:

- How is data processed (i.e., data encryption/decrytion, encoding/decoding)? What is the optimal coding ratio for a given network state?
- How is data distributed over the storage servers (i.e., data distribution)?
- What is the end-to-end delay? How the system performances change with the data size?
- How are the encryption/decryption keys processed and distributed over the cloud?
- How does the system detect and handle storage server failures (i.e., data regeneration)? How and how much will data be regenerated? Where is the new data stored?

B. Demonstration Setup

We use the setup configuration illustrated in Fig. 2 to demonstrate our prototype FDCS. In particular, we will use several mobile devices (laptops, smartphones, and tablets) to connect to FDCS via web browsers to perform data storing and downloading. In this configuration, the information is transmitted via a *local access point* (AP) and routed through the Internet to the FDCS storage servers which are located at InfoBeyond Technology LLC (Louisville, Kentucky, USA.) The gateway server will act as the system coordinator to receive the information, process and distribute it to the storage servers. In our demonstration, we will test the prototype with heterogonous storage servers (i.e., different storage capacity and transmission speed). In addition,

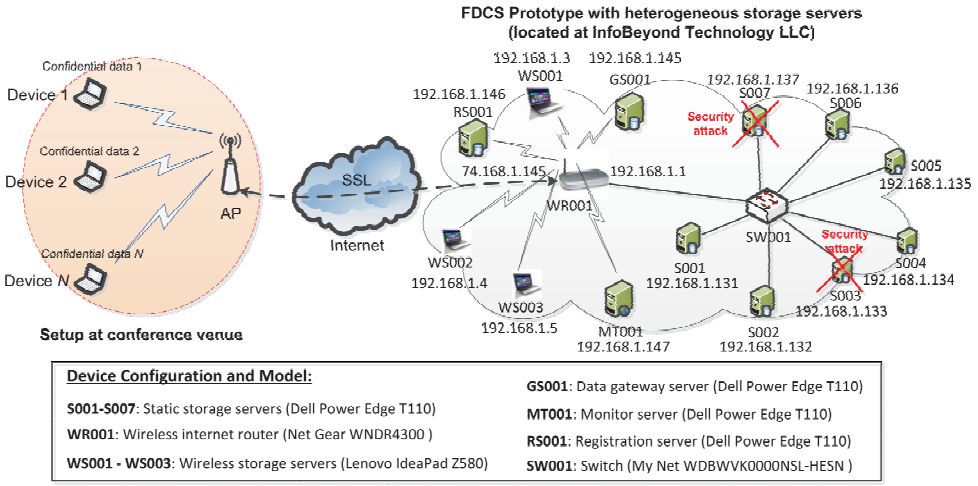

Fig. 2. Configuration setup of the proposed demonstration

we will test the prototype security capability by assuming some storage servers are compromised (i.e., unaccessible) and show how the system cope with such situations.

C. Required Equipment

To perform the demonstration, we will provide the following devices:

- Two laptops, one tablet, and one smartphone
- Two AC power cords

In addition to the above equipment, the following facilities need to be provided at the conference venue: Internet access (preferably wireless), AC power source, one 60cm $\times$ 120cm table. The setup time is about 30 minutes.

D. Expected Results

The expected results of this demonstration is to show the feasibility, efficiency, and security of FDCS prototype. We use several devices to remotely connect to the FDCS prototype for data storing and downloading. Different types of data files will be stored into and read from the system during the demonstration. The system scalability and efficiency will be tested by changing the file size, which can go upto hundred of Mega bytes, depending on the transmission speed of the Internet and local network. We are expected that the data storing and downloading will be fast with small *end-to-end* delay. We are also expected that the failures of data severs within some limit (depending on the selected code ratio) won't affect to the data reliability. Although the data maintenance process is transparent to the users, we will show how data is regenerated when a storage server fails. Some of the screen shots of the demonstration are illustrated in Fig. 3. For example, Fig. 3(a) shows a screen shot of the window where the user can perform data manipulation (storing, reading, copy, etc.) to and from the prototype. In addition, Figs. 3(b) and (c), respectively, show how a data file is processed (i.e., encoded/decoded encrypted/decrypted) and where the locations of the stored fragments are.

IV. Conclusion

We proposed to demonstrate prototype of a fragment-based distributed cloud storage system. The demonstration is performed via a web-based interface from multiple mobile devices. Several system capabilities on data protection will illustrated during the demonstration.

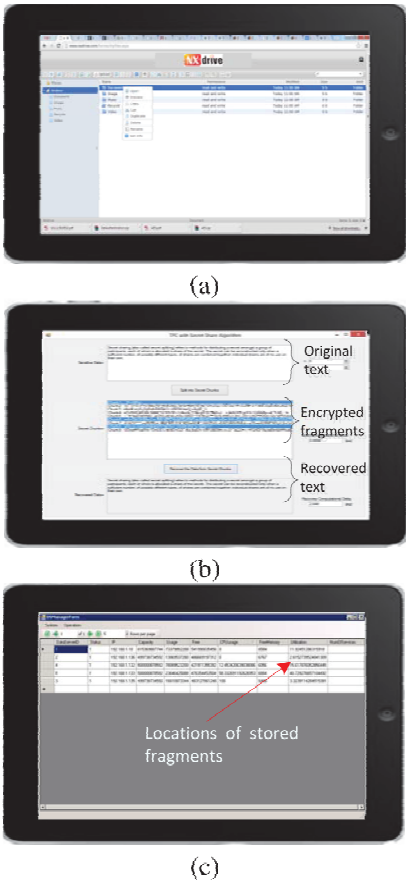

Fig. 3. Some screen shots of the proposed demonstration: (a) data manipulation window; (b) data encoding/decodingn & encryption/decryption; and (c) data *distribution*.

References

[1] P. Chakravarthey, T. Tran, B. Xie, and A. Kumar, "SOS: Social network-based distributed data storage," in *to appear in the 38th Annual IEEE Conference on Local Computer Networks (LCN)*, 2013.
[2] B. Xie and T. Tran, "Social cloud data storage system," in *Technical report 13 – 05, InfoBeyond Technology LLC*, 2013.
[3] B. He, T. Tran, and B. Xie, "Authentication and identity management for secure cloud business and services," in *Accepted for Book chapter in Cloud Computing, IGI Global*, 2013.
[4] Y. W. A. Dimakis, K. Ramchandran and C. Suh, "A survey on network codes for distributed storage," *Proceedings of the IEEE*, pp. 476–489, 2011.
[5] A. Shamir, "How to share a secret," *Magazine Communications of ACM*, vol. 22, no. 11, 1979.