Cloud Storage System which Prohibits Information Leakage on Both Client and Server

Kuniyasu Suzaki*, Toshiki Yagi*, Kazukuni Kobara*, Nobuko Inoue‡, Tomoyuki Kawade‡, Koichiro Shoji‡
{k.suzaki , yagi-toshiki , k-kobara}@aist.go.jp, {ninoue , kawade , shoji}@sciencepark.co.jp
* National Institute of Advanced Industrial Science and Technology(AIST), ‡SciencePark Corporation

1. Introduction

Most information leakages on cloud storage occur on both edge machines, i.e. servers and clients, rather than network. For example, an administrator of a server may peek through user's information and a user may leak information from an application with mis-configuration on user's client machine.

We propose a Virtual Jail Storage System (VJSS) to solve this problem. Our system encrypts a file with All-Or-Nothing Transform (AONT) and cut off a small part of the encrypted file. The small part is stored in a client as a split tally and the other part is up-loaded on some cloud storage systems after encoding of Reed-Solomon error correction code. The original file is reconstructed with the split tally in the VJSS on a client. Furthermore the file is managed by access control on the storage system. Files in the VJSS can be opened by a suitable application but cannot be copied, printed, and screen-captured&pasted. This system is implemented on Windows 7.

2. Prohibition of Information Leakage on Server

While Cloud Storages are becoming popular, information leakage from them is concerned about. For example, even if a file is encrypted by a user with a password, the file may be decrypted by a brute-force attack. In addition, Dropbox had an incident that it accepted any password to log in to customers' accounts in 2011.

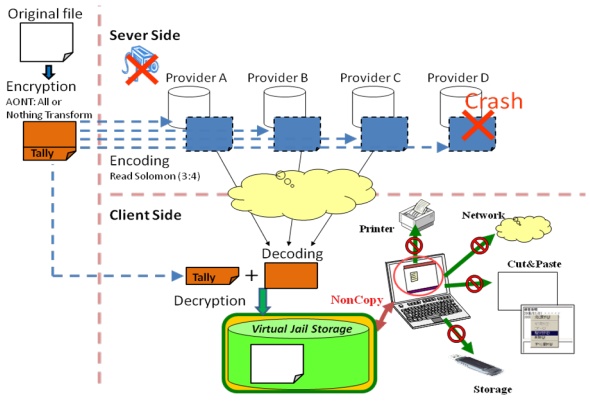

Figure 1. Overview of Virtual Jail Storage System (VJSS).

Our Virtual Jail Storage System (VJSS) uses not only encryption but also a split tally. The small split tally of encrypted file is stored only in a client and has enough entropy. Therefore brute force attacks cannot be applied on servers. The split tally is needed to reconstruct the whole file. Figure 1 shows the whole image of our VJSS. A file is uploaded through three stages; encryption, cutting off a small split tally, and encoding for error correction. As encryption, we employ All-Or-Nothing Transform (AONT), since AONT endorses that the original file is reconstructed only if the whole parts of the encrypt file are gathered. Our VJSS cuts off a small part of the encrypted file as a small split tally and keeps it on a client so that VJSS on it can reconstruct the original file. The remaining data where the split tally is removed are divided according to the error correcting code and then uploaded to some cloud storage systems. The redundancy in the distributed data enables the recovery even if a server stops its service. We know most cloud servers offer high reliability, but there are still some incidents to fail services. We think we should avoid the risk to depend on only one service. Furthermore, the redundancy also make possible to balance the loading data on a client. For the error correcting code, we employ Reed-Solomon since it is widely used and has open-source libraries.

The technique which combines AONT and Reed-Solomon is resemble to AONT-RS[2], but our proposed technique adds split tally to prevent information leakage when all server data are gathered. Furthermore, our method includes simple deduplication to reduce network traffic. The processes for uploading and reconstructing a file on VJSS are shown in figure 3 and 4. A file is divided into small pieces for deduplication. If some small pieces have same contents, they are shared by deduplication. The deduplication is also applied on existing pieces of other files and reduces the total network traffic. Even if the deduplication is simple, it can find many same pieces when the VJSS is used as a file backup system.

3. Prohibition of Information Leakage on Client

Information leakage on a client machine can be caused by inadvertent copying a file to USB memory or printing papers. If a user has evil intent, he/she can move the contents by screen-capture&paste. PDF file can add options to prevent printing and screen-capturing&pasting, but normal file cannot be set such a security function.

Our VJSS offers virtual jail storage to distribute a file from cloud storage services. A file reconstructed in the VJSS is guarded by an access control called "NonCopy" which prevents Information leakage on the client. As shown in figure 1, NonCopy prevents copying the file, printing the file, and screen-capturing&pasting of the file.

The NonCopy is a set of hooking for APIs of Windows kernel, functions of DLL, and event handlers. I/O APIs for a VJSS are hooked and copying a file is prevented. Function table for printing DLL is hooked (e.g., StartDoc function), and printing is prohibited. Event messages for keyboard and mouse are hooked by SetWindowsHookEx and screen-capturing&pasting is prevented.

4. Current Implementation

Current VJSS is based on Loopback Content Addressable Storage (LBCAS) [3,4], and adds security functions. The VJSS is implemented on Windows 7. It uses "Dokan[1]" for user mode file system, BerkleyDB for managing data, and some libraries for AONT and Reed-Solomon.

The VJSS is created as a client application and does not require special function on servers. The servers only have to work as data storage (i.e., file server or database). We plan to use Amazon S3, Dropbox, and other popular cloud storage services.

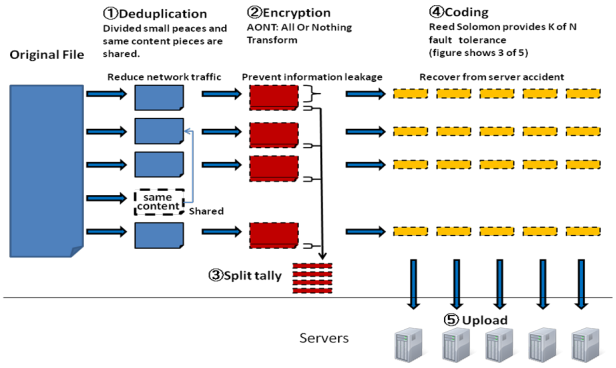

Figure 2. Uploading a file on servers for VJSS.

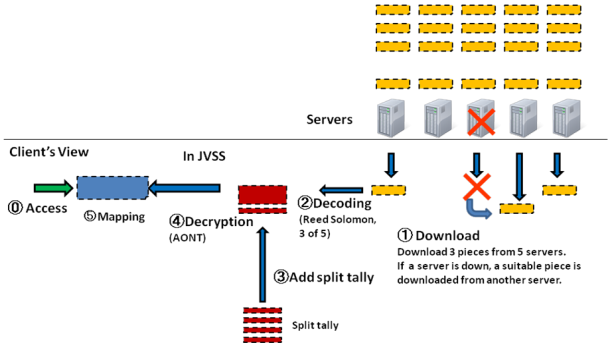

Figure 3. Reconstructing a file on VJSS.

Reference

[1] Dokan: http://dokan-dev.net/en/

[2] J. Resch, and J, Plank, AONT-RS: Blending Security and Performance in Dispersed Storage Systems, USENIX FAST'11.

[3] K. Suzaki, T. Yagi, K. Iijima, and N.A. Quynh, OS Circular: Internet Client for Reference, USENIX LISA'07.

[4] K. Suzaki, K. Iijima, T. Yagi, and C. Artho, Analyze Disk Access Pattern of File Systems for Content Addressable Storage, Ottawa Linux Symposium'11.