1. Introduction
It is becoming easy to compose the dispersed storage system with more than two storage servers or corresponding Cloud servers. We have been developing the data recovery technology, named HS-DRT, which stands for High Security-Distribution and Rake Technology. It is for saving important data by making use of scrambling and encrypting mechanism and in addition, distributing to the appropriate network-resources. HS-DRT achieves the high security by the secured distribution and the effective spatial scrambling of the data. It also can attain a high dependability by the appropriately designed number of plural distribution and the appropriate redundancy[1-3].

The essential configuration of HS-DRT is shown in Figure 1. The total data for backing up are compressed at first. Then it is encrypted and spatially scrambled, and divided into 64 data blocks. After this procedure, each segment data is scrambled again individually. Next to that, in order to attain the redundancy, a parity data are generated and added. After these procedures, it will be delivered and is stored into several other Cloud storages.

The informations to process the data, named “meta-key”, is also processed by the similar way and is stored several other Cloud storages.

2. Performance Evaluation of the HS-DRT
To study the implementation of the spatial scrambling functions in the HS-DRT core engine, we evaluated a simple non-optimized method, and two cases which were optimized for use with multi-core processors; for each method we used three sizes of data, and the whole simulation was written in the C and C++ language. In the first case, the original data is divided into 64-kbyte data blocks and shuffled using the Fisher-Yates algorithm three times successively. In the other case, each 64-kbyte data block is further divided into 64 data blocks and the resulting 1024-byte data blocks are shuffled using the Fisher-Yates algorithm. Since in the second case, the scrambling effect is limited within the range of 1024 bytes, 64 numbers of 1024 bytes data block are further shuffled three times using the Fisher-Yates algorithm in order to achieve the appropriate degree of randomness.

The above mentioned data processing has been implemented using a single thread as a basic application program. To optimize multi-core processing, the Fisher-Yates shuffling is applied to each thread in the environment of the ubuntu12.04LTS OS.

We adopted three types of CPU, which are 1-core (AMD Athlon 1640B), 2-core (Intel Celeron G530), and 4-core (Intel Corei7 960), from the viewpoint of economy. We evaluated the processing time by using the data sizes of 64 Kbytes, 640 Kbytes and 6.4 Mbytes. The measured results are shown in Figure 2.

It was confirmed that the processing efficiency of the 1-kbyte data blocks is worse than that of 64-kbyte data blocks. The reason is considered to be that when 1-kbyte data blocks are processed by each thread, then the corresponding overhead for producing 64 threads in a multi-thread environment OS are comparatively large. In addition, the processing time increases with an increase in the data size. This means that each thread of the OS has been sufficiently optimized.

In case of a 1-core processor, the efficiency of the single thread processing for each 64-kbyte data block (referred to as the 64k-single method) is approximately the same as the case of multi-thread processing for each 64-kbyte block (referred to as 64k-multi method). It shows that there is no overhead for multi-processing in the case of 1-core processing.
In contrast, in the case of the 2-core and 4-core processing, when handling 6.4 Mbytes of data, the efficiency of the 64k-multi method is five times faster than the 64k-single method for 2-core processing, and nine times faster than the 64k-single method for 4-core processing.

![Fig. 2 Evaluation of HS-DRT](image)

Consequently, we derived the following two characteristics of the HS-DRT engine.

1. The 64k-multi method can be used effectively with multi-core processing in handling the spatial, scrambling and shuffling procedure under the Linux OS.
2. Since an increase in scrambled data size does not result in much increase in the time required by the Fisher-Yates shuffling, it is recommended to increase the size of the data block for spatial scrambling by utilizing the 64k-multi method.

### 3. Commercialized Equipment with HS-DRT

In this section, we describe an equipment we have newly developed, which is effective not only for cloud systems, but also for on-premise systems. The equipment, named “DRTbox”, is a small box with an ARM-CPU and Linux based OS. The DRTbox is equipped with one Ethernet port and one serial port. It works under temperatures of 0 to 55 degrees centigrade and the power consumption is only 5.0W. One of the typical configurations of the system is illustrated in Figure 3, which shows the case of a configuration in which a user site is backed up by 2 other sites. Data1 is the original data which is stored by a user. Data1a is stored in Site A, and Data1b is stored in Site B. Data1a and Data1b include HS-DRT-processed 32 blocks comprised of 64 blocks of whole data.

![Fig. 3 Implemented System](image)

This means that even if one cloud out of three clouds does not work, the corresponding data will not be lost. Unlike conventional cloud systems, each site has its own file server for users. In this case, in accordance with the specific user requirements, the system is on-premises, and each site is basically controlled by a single company or an organization. Each site can be linked via a secure network.

### 4. Conclusion

We have presented a commercialized system and performance results for a system using HS-DRT in cloud computing environments. Further studies should address the optimum network utilization technology. We are planning to verify the essential characteristics necessary to fully utilize network resources, in order to commercialize an ideal disaster recovery system.

### References