Commercialized Practical Network Service Applications from the Integration of Network Distribution and High-Speed Cipher Technologies in Cloud Environments

Kazuo Ichihara, Naoko Nojima
Net&Logic Inc.,
Meguro-ward,
Tokyo, Japan
e-mail: ichihara@nogic.net, nojima@nogic.net

Yoichiro Ueno, Shuichi Suzuki, Noriharu Miyaho
Department of Information Environment,
Tokyo Denki University, Inzai-shi,
Chiba, Japan
e-mail: ueno416@mail.dendai.ac.jp,
suzuky@mail.dendai.ac.jp, miyaho@mail.dendai.ac.jp

***Abstract*— This paper presents the evaluation results of the commercialization of High Security Disaster Recovery Technology (HS-DRT) that uses network distribution and high-speed strong cipher technologies to realize efficient and secure network services. We have commercialized a disaster recovery system and evaluated the performance of the distributed engines using the hash functions, versatile spatial scrambling functions, etc., in cloud computing environments. The average processing time has been estimated in terms of the method of implementation of the engine. As for practical network applications, an automatic back-up system using an FTP server has been introduced. We have developed on-premise systems which achieve high security through the use of HS-DRT. Finally, we also propose future technologies for preventing an insider attack.**

Keywords-disaster recovery; backup; distributed processing; cloud; strong cipher.

I. INTRODUCTION

Innovative network technology, which can guarantee, as far as possible, the security of users' or institutes' massive files of important data from any risks such as an unexpected natural disaster, a cyber attack, etc., are becoming increasingly indispensable day by day. As a means of satisfying this need, cloud computing technology is expected to provide an effective and economical backup system by making use of a very large number of data stores and processing resources which are not fully utilized. It is expected that this file data backup mechanism will be utilized by government and municipal offices, hospitals, insurance companies, etc., to guard against the occurrence of unexpected disasters such as earthquakes, large fires and storms and Tsunamis. To achieve secure back up, there is an indispensable need for prompt restoration, which may make versatile use of cellular phones, smart phones, digital signage equipment and PCs, in addition to cloud resources dispersed in multiple geographical locations. In addition to these factors, many companies and individuals involved in industry and commerce are interested in making use of public or private cloud computing facilities, provided by carriers or computer vendors as a means of achieving security and low maintenance and operation costs. In this paper we present the results of an evaluation of an innovative file backup network service, which makes use of an effective ultra-widely distributed data transfer mechanism and a high-speed strong cipher technology to realize efficient, safe data backup at an affordable maintenance and operation cost [1-6].

When a block cipher is used, the required processor and memory costs increase in an exponential manner with increasing data volume. However, with a stream cipher, the input data is simply operated on bit-by-bit, using a simple arithmetic operation, and high-speed processing becomes feasible. This is the fundamental difference between the two cipher technologies. It is possible to combine the use of technologies, specifically, the spatial scrambling of all data files, the random fragmentation of the data files, and the corresponding encryption and replication of each file fragment using a stream cipher. Figure 1 shows the concept of the proposed network service compared with a conventional back up system using the leased lines.

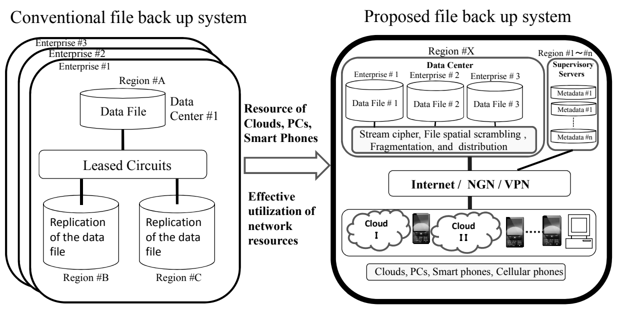

Conventional file back up system
Proposed file back up system
Enterprise #3
Enterprise #2
Enterprise #1
Region #A
Data File
Data Center #1
Leased Circuits
Replication of the data file
Replication of the data file
Region #B
Region #C
Resource of Clouds, PCs, Smart Phones
Effective utilization of network resources
Region #X
Data Center
Enterprise # 1
Enterprise # 2
Enterprise # 3
Data File # 1
Data File # 2
Data File # 3
Stream cipher, File spatial scrambling , Fragmentation, and distribution
Region #1～#n
Supervisory Servers
Metadata #1
Metadata #1
Metadata #n
Internet / NGN / VPN
Cloud I
Cloud II
Clouds, PCs, Smart phones, Cellular phones

Figure 1. Concept of the proposed network service

In the data center, it is appropriate to introduce a secret sharing scheme for sending "encryption metadata" to the supervisory servers deployed in the several different locations for deciphering the original file data. To enhance security it is better to send the metadata using a Virtual Private Network(VPN). This mechanism make it quite difficult to find out any series in the "encryption metadata" itself. From a disaster recovery point of view, a secret sharing scheme with some appropriate "thresholds" should be introduced in the proposed system. If the system uses a (3,5)-threshold scheme, the system uses five supervisory servers, and can tolerate the simultaneous failure of two servers. On the other hand, from a cyber terrorism point of view, if the system uses a (3, 5)-threshold scheme, a cracker has to intrude at least three “encryption metadata” servers and one alive/valid information server at the same time [1].

We have developed a high security disaster recovery technology (HS-DRT) for realizing file backup network services. To realize the proposed DRT mechanism, the following three principal network components are required: (1) a secure data center, (2) several secure supervisory servers, and (3) a number of client nodes such as smart phones, cellular phones, digital signage equipment or cloud storage which are available for use. The corresponding history data, including the encryption key code sequences, which we call "encryption metadata", are used to recover the original data. This mechanism is equivalent to realizing a strong cipher code, comparable to any conventionally developed cipher code, by appropriately assigning affordable network resources.

To realize a safe and highly secure system, it is necessary to assure the users that their important file data cannot be stolen from the service provider. When the important file data is composed of a number of fragments distributed to the several providers' clouds, we need to ensure that even a single fragment cannot be deciphered by any of the providers. From this point of view, we considered the special case of preventing an insider attack. The proposed technology can also increase both the cipher code strength and the operation of decryption and reassembly of original file data.

In this paper, we briefly describe related work in Section II, and then, the basic configuration of the HS-DRT engine in Section III, and a performance evaluation of the proposed network services using the HS-DRT engine is presented in Section IV. Practical commercialized systems are discussed in detail in Section V. In Section VI, we describe the technique behind the proposed method of preventing insider attacks. Finally, we describe the conclusions and the future issues in Section VII.

II. RELATED WORK

In the field of Disaster Recovery Systems, there have been many research publications and many commercial products. Most disaster recovery systems include data replication functions using stand-by servers in remote locations. In contrast, the proposed disaster recovery system using HS-DRT uses a secure distributed data backup scheme. By making use of the HS-DRT mechanism to achieve a reliable backup scheme, we have been able to provide a system product at a reasonable price to both individuals and companies. For example, we can provide only the backup application by using HS-DRT with multiple cloud services as backup storage in accordance with appropriate customer contracts. So, it is rather difficult to compare our proposed system quantitatively with an ordinary disaster recovery system from the viewpoint of the price.

In the field of secure data backup systems, other related studies have included the concept of a distributed file backup system [7][8]. However, in these studies, neither a precise performance evaluation nor a practical network service system is clearly described.

In the field of intrusion tolerance, a file server should introduce such functions as encryption, fragmentation, replication, and scattering [9]. The core technologies of HS-DRT resemble those of a persistent file server, except for the spatial scrambling and random dispatching technology. With these two technologies, deciphering by a third party, by comparing and combining the encrypted fragments, becomes almost impossible. In addition, HS-DRT is applicable to other fields, such as secure video streaming, etc.

In the field of Distributed Anonymous Storage Services, the replication and scattering techniques were introduced in the Eternity Service [15]. However, the main objectives of these services are longevity and anonymity.

In contrast, these objectives (longevity and anonymity) are not taken into account as primary requirements in the HS-DRT. HS-DRT is effectively utilized for the purpose of fast, safe and secure file back up until the next backup event. Only the authorized users are able to initiate the process of recovery of their file data contents.

III. BASIC CONFIGURATION OF THE HS-DRT ENGINE

The HS-DRT file backup mechanism has three principal components, as shown in Figure 2. The main functions of the proposed network components, which are Data Center, Supervisory Server and various client nodes, can be specified as follows. The client nodes (at the bottom of Figure 2 are PCs, Smart Phones, Network Attached Storage (NAS) devices, Digital Signage and Storage Services in the Cloud. They are connected to a Supervisory Server in addition to the Data Center via a secure network.

The Supervisory Server (on the right in Figure 2) acquires the history data, which includes the encryption key code sequence (metadata) from the Data Center (on the left in Figure 2) via a network. The basic procedure in the proposed network system is as follows.

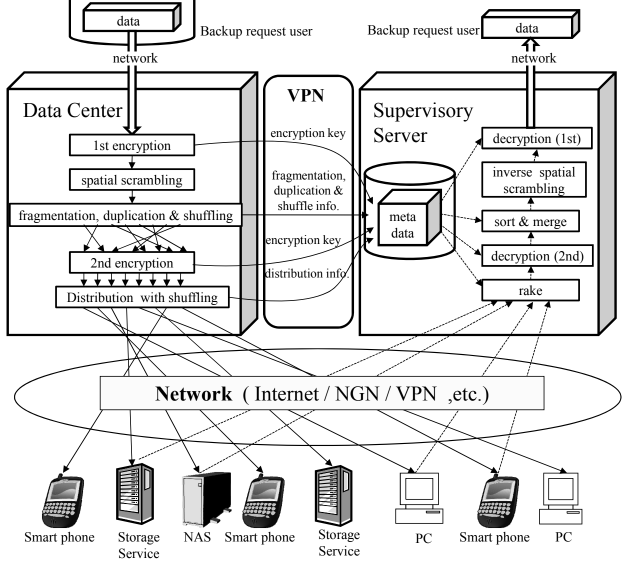


Figure 2. Principle of HS-DRT file backup mechanism

A. Backup sequence

When the Data Center receives the data to be backed up, it encrypts it, scrambles it, and divides it into fragments, and thereafter replicates the data to the extent necessary to satisfy the required recovery rate according to the pre-determined service level. The Data Center encrypts the fragments again in the second stage and distributes them to the client nodes in a random order. At the same time, the Data Center sends the metadata to be used for deciphering the series of fragments to the supervisory servers. The metadata comprises encryption keys (for both the first and second stages), and several items of information related to fragmentation, replication, and distribution.

B. Recovery sequence

When a disaster occurs, the Supervisory Server initiates the recovery sequence. The Supervisory Server collects the encrypted fragments from the various appropriate clients in a manner similar to a rake reception procedure. When the Supervisory Server has collected a sufficient number of encrypted fragments, these are decrypted, merged, and descrambled in the reverse order of that performed during the second stage of encryption, and the decryption is then complete. Through these processes, the Supervisory Server can recover the original data that has been backed-up.

Let us consider the probability of successful recovery, which can be estimated from the following equation. Here, P is the failure rate of each client node, n is the degree of duplication of each fragment, and m is the number of fragments [1].

Probability of recovery $= (1-P^n)^m \cong 1-mP^n$

Probability of recovery failure $\cong mP^n$

For example, when each file fragment's failure rate P is assumed to be 0.2, and the original file is divided into 30 fragments, and 30 replications are made of each fragment, the probability of recovery failure becomes less than 10^{-19}. The above case applies to the use of smartphones, cellular phones, or PCs.

The failure rate of such devices can be estimated to be 0.2 by considering their connectivity and reliability, erring on the safe side. Here, the size of users' important data is classified into three types, called Type1, Type2, and Type3. The data size of Type1 is at most around several hundreds of megabytes in size. The data for Type2 is at most around several tens of gigabytes, while that for Type3 is up to several terabytes. Considering a smart phone's memory capacity to be 32 Gbytes, and the number of terminals to be several tens of millions, these are used for Type1 and Type2 data, with the assurance that less than 1% of the vacant memory resource in the terminals offered would be used to support the backup service. This percentage can usually be measured by the user's self-check monitoring and the monitored result can be transmitted to the remote supervisory center. The value of 1% is an example of the conditions to be temporarily assigned to encourage people to participate. Several cloud storage resources can be effectively utilized for Type3 data. When cloud storage is used, then P can be less than 10^{-11} and a reliability much higher than that available commercially can be easily obtained.

The security level of the HS-DRT does not only depend on the cryptographic technology but also on the method of specifying the three combined factors, that is, spatial scrambling, fragmentation/replication, and the shuffling algorithm. Because of these three factors, nobody is able to decrypt the data without collecting all relevant fragments and sorting the fragments into the correct order. Even if some fragments are intercepted, nobody is able to decrypt parts of the original data from such fragments.

The spatial scrambling procedure can be realized by executing a simple algorithm using a C-style description [1]. This computation process should be repeated several times. To de-scramble, it is only necessary to perform the same operations in the reverse order. Introducing the above mentioned spatial scrambling technology makes it almost impossible for a third party to decipher the data by comparing and combining the encrypted fragments, since the spatial scrambling scatters the relevant fragments widely and uniformly amongst the storage devices.

One of the innovative ideas of HS-DRT is that the combination of fragmentation and distribution can be achieved in an appropriately shuffled order. Even if a cracker captured all the raw packets passing between the data center and the client nodes, it would be extremely difficult to assemble all the packets in the correct order, because it would be necessary to try about N! (N: number of fragments) possibilities, where N is sufficiently large. In fact since the bit patterns of any two encrypted fragments are completely different from each other owing to the different encryption keys, it is impossible to associate one encrypted fragment with another. Crackers would require innumerable attempts to decipher the data. In addition, HS-DRT mainly uses a shuffling method that uses pseudo-random number generators for the distribution to the client nodes. When we distribute the fragments of the encrypted data to widely dispersed client nodes, we can send them in a shuffled order, since we predetermine the destination client nodes from the shuffled table in advance. When we use a shuffle table which makes use of the "Fisher-Yates shuffle" algorithm with 3 rounds, leading to a uniform distribution, the table itself is hard for a third party to guess [10].

Practical systems to realize a hybrid HS-DRT engine can be realized effectively by making use of a cloud computing system at the same time. The system essentially consists of the following four parts: thin clients, a web applications server (Web-Apps Server), an HS-DRT engine, and Storage Clouds. Thin Clients are terminals which can use web applications in a SaaS (Software as a Service) environment. Thin Clients can make use of the application services which are provided by the web applications server. The HS-DRT engine is considered to be a component of the hybrid cloud computing system, which can also strengthen the cloud computers' security level at the same time. Usually, users can also make use of one of multiple HS-DRT engines available in the cloud environments through a contract with the providers. The data center and the Supervisory Server can be integrated in the HS-DRT engine. The HS-DRT engine can effectively utilize the storage clouds which have a function related to a web application and execute the encryption, spatial scrambling, and fragmentation of the corresponding data files. It is very important to note that the processing efficiency of the HS-DRT engine can easily be improved by increasing the amount of the web cache memory.

We need to consider the scalability of the HS-DRT engine, since it may become a bottleneck in a very large system, owing to the number of clients and the amount of storage. In such cases, the HS-DRT engine may use a key-value database. As the HS-DRT engine can easily work with other HS-DRT engines, the system can be extended.

IV. Performance Evaluation of the HS-DRT

To study the implementation of the spatial scrambling functions in the HS-DRT engine, we evaluated a simple non-optimized method, and an optimized method for use with multi-core processors; for each method we used four sizes of data, and the whole simulation was written in the C language. The original data is divided into 64-kbyte data blocks and each 64-kbyte data block is further divided into 64 data blocks. The resulting 1024-byte data blocks are shuffled using the Fisher-Yates algorithm. Since the scrambling effect is limited within the range of 1024 bytes, the 64 1024-byte data blocks 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 for 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 were 1-core (AMD Athlon 1640B), 2-core (Intel Celeron G530), and 4-core (Intel Corei7), from the viewpoint of economy. We evaluated the processing time by using data sizes of 64kbytes, 640kbytes, 6.4Mbytes and 64MBytes. The measured results are shown in Figure 3. In the graphs, the processing time is shown in terms of the equivalent bit-rate. In case of the 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 the 64k-multi method). It shows that there is no benefit in 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 speed 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. 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.

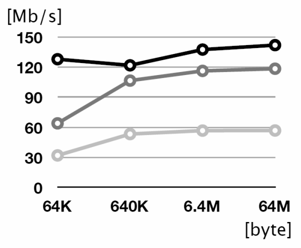
2) Since an increase in scrambled data size does not result in much increase in the time required by the Fisher-Yates shuffling, it is recommended to increase the size of the data block for spatial scrambling by utilizing the 64k-multi method.

64K-single method [msec]

	64K	640K	6.4M	64M
1Core	16	96	902	9004
2Core	8	48	440	4320
4Core	4	42	372	3604

64K-multi method [msec]

	64K	640K	6.4M	64M
1Core	16	94	924	9012
2Core	1	10	88	853
4Core	1	10	42	412



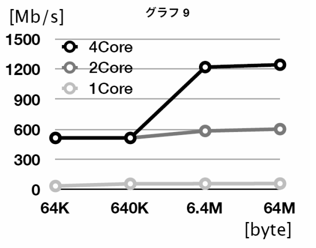


Figure 3. Evaluation of HS-DRT

V. Practical Commercialized System

We have considered the following business model.

1) The proposed system can share the network resources such as clouds for different users with different security and availability requirements. This can lead to an effective cost reduction compared with conventional systems using leased lines and data centers.

2) The proposed system can utilize the enormous number of PCs, smart phones, and digital signage systems as network resources to provide the backup services in accordance with individual contracts with the owner users. This also can lead to an effective cost reduction compared with the conventional network system provider resources.

3) By effectively making use of the above mentioned network resources, it is possible to provide backup services for only a little bit more than the cloud usage prices.

4) For users who cooperate in offering available unused network resources (memory), we can offer them a communication tariff reduction or alternatively free backup services.

A. Simple FTP server automatic backup system without using Clouds

In 2010, we commercialized the personal HS-DRT backup system that is called “@Cloud-DRT backup”. In this system, the customer only installs client software on their PC, and drags-and-drops the required files to the specific folder. This process is suited to personal use, but it is not fit for a large scale file service, such as an FTP server.

We have therefore made an autonomous backup system for FTP servers by adapting the way “@Cloud-DRT backup” is used. The components of our autonomous backup system are shown in Figure 4. We added a backup server and only installed the watch program on the target ftp server, without modification of the ftp daemon. When a user uploads a file to the FTP server, the watch program needs to look for the

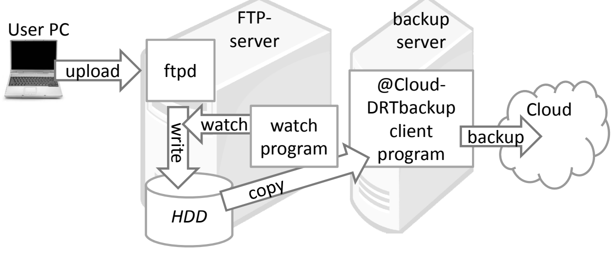


Figure 4. FTP server automatic backup system

added or updated file. This process consumes significant CPU resources, so we used the log file of the ftp daemon. The log file is updated after every transfer, and it contains all the information about added or updated or deleted files. When the watch program detects a modification of the log file, the updated files are copied from the FTP server to the backup server. In the backup server, the client program of "@Cloud-DRT backup" receives and backs up the updated files to the cloud.

In our prototype backup-system, the speed of performance of the file backup is only 1.4Mbps. The reason for this rather low performance is due to the use of the client program of "@Cloud-DRT backup" without modification. In this prototype, we introduced the personal use client program on the backup server. It will be necessary to build a new client program for an enterprise use as the next step.

B. *Practical disaster recovery system demonstrating the advantage of mutually independent geographical sites*

The essential configuration which has been developed for the data backup system which makes use of cloud environments is shown in Figure 5. The Meta-keys Processor uses several cloud environments after user authentication, and the multiple encrypted/divided/replicated data blocks are deployed in the different cloud environments, as judged appropriate, from the available network and computer resources.

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 [13] 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

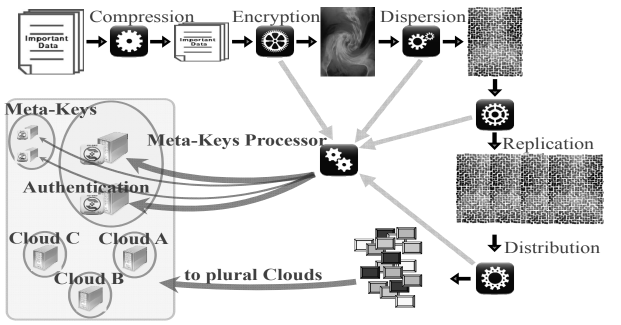


Figure 5. Data backup configuration using several clouds

5.0W. One of the typical configurations of the system is illustrated in Figure 6, which shows the case of a configuration in which a user site is backed up by two 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.

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. Let us consider the case where the data are processed by HS-DRT, and the size of Data1 is 100M bytes. Data1a and Data1b are each 50Mbytes in size. In this case, each site is basically controlled by a single company or an organization. Each site can be linked via a secure network. We used the technology to store the metadata keys for the other sites using a simple AES encryption. This makes the system operation simple, since the method to back up the metadata keys is complex and it is generally difficult to transport them securely.

If the user can use one more site (Site C), Data1 is divided into 64 blocks, each of 100/64 Mbytes, and processed to give 96 blocks. The additional 32 blocks are the parity data of the other 64 blocks. After that, the 96 blocks of data is divided into 3 groups, which are stored in Site A, Site B and Site C, respectively. This means that even if two sites out of four sites do not work, the corresponding data will not be lost. This configuration of the system is more robust than the one with just two back-up sites.

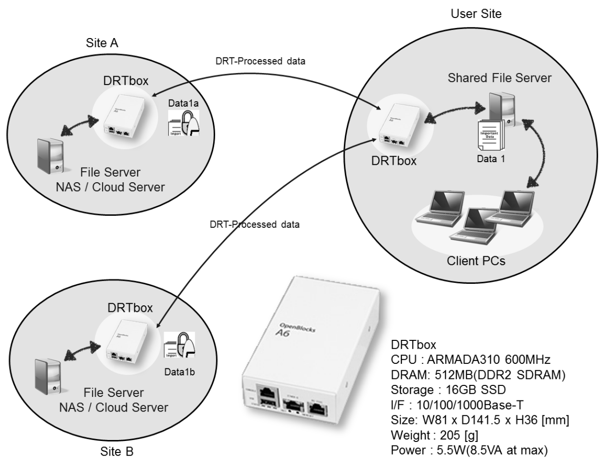


Figure 6. Implemented system using DRTBox

VI. Insider Attack on Block Ciphers

A. *Background*

We need to consider case of an insider attack in which a malicious vender of a security system attacks his clients by using his cipher program, which is generally a block cipher. It can be shown that the Initial Value (IV) mode [11][12] of the block cipher is vulnerable to an insider attack. Moreover, we will propose a countermeasure to this attack.

While we are planning to provide a safe and secure system for the data backup, the client may be uneasy that the data might be stolen through an unexpected insider attack. To prevent this situation, we need to introduce the original block cipher mode, only for this purpose.

B. Newly occurring model of attack on block ciphers

In an attack on the block cipher based on the IV mode, the vendor as a supplier of the cipher program is assumed not to have any malicious intentions towards clients. The safety of block ciphers like the Advanced Encryption Standard (AES) has been discussed under such assumptions.

Cipher Block Chaining (CBC) mode or counter mode are recommended as safe methods to take the place of Electronic Code Book (ECB) mode. Since AES actually exhibits steady performance and reliability, it is now adopted as a world standard. However, one may often have a sub-conscious motivation for an insider attack. What sort of program could a vendor produce if the malevolent intentions and which might succeed in an insider attack? Such a situation raises a serious concern in the disaster recovery service environment which makes full use of cloud computing systems. We need to resolve this kind of problem when we offer disaster recovery services.

C. The model of an insider attack against block ciphers

First of all, let us define the model of an insider attack on a block cipher.

1) The vendor of the cipher program attacks the user of that program.

2) The cipher program outputs only the cipher text through the client user's input of key and message data. In particular, the cipher program is inaccessible to the network.

3) The user can preserve the cipher text, and he can always confirm the content by decoding according to the publicized coding method.

4) The attacker can obtain only the cipher text.

The IV mode of the block cipher is not safe, as follows. The attacker sends the client user encryption and decoding programs, called E(K,M), and D(K,C), respectively, as described below. Here K is a private key that only the client user knows, and M is a message.

A pseudo-random number IV adheres to the head of the cipher text C_0, such that $C=E(K,M) = (IV,C_0)$.

Here C is a cipher text that looks quite normal, and M can be computed by the normal decoding program D(K,C). However, the attacker can obtain the key K and so succeeds in the attack.

E(K,M) is composed as follows.

1. The attacker prepares internal code e and d in E(K, M).
2. Here, $e(M) = e(K_0, M) = C$, $d(C) = d(K_0, C) = M$. K_0 is a key that only the attacker knows.
3. The program E(K, M) receives the user's key K and plaintext M.
4. Let IV = e(K).
5. Calculate C_0 in a usual IV mode.
6. Output $C = (IV, C_0)$.
7. The attacker acquires C from the network.
8. The attacker can calculate K = d(IV).

When this method is implemented with hardware, it is especially difficult detect such an attack. This type of attack is not addressed by protocol analyzers such as AVISPA or Scyther since in their treatment of cryptographic primitives they adopt the so-called black box approach [14]. Even if we could detect the insider attack by using a protocol analyzer by reverse engineering a cryptographic primitive, we could hardly identify the same attack for another primitive. But, we propose an integration mode of block ciphers, as described below, which mode can avoid all insider attacks.

D. Vulnerability of double encryption method of block ciphers

Although the IV mode is not safe, it might be thought that it is safe if it is encrypted twice, including the appended pseudo-random number. However, it is understood that it is not safe if the attacker uses the following approach.

If $IV=D(K,IV_0)$, then , $E_0(K,C)=E_0(K,(IV,C_0))$
$=E_0(K,(D(K,IV_0),C_0))=(IV_0,C_1)$, where E_0 is the corresponding ECB mode encryption .

Hence, it is assumed $IV_0=e(K)$ by using a secret internal code e that only the attacker knows.

E. Integration mode of block cipher

Now, we propose an encryption mode that is not vulnerable to an insider attack. This encryption mode does not have redundancy, and has the specific characteristic of not being vulnerable to an insider attack. Moreover, it has security more than equal to that of the block cipher which was originally used.

We can provide a proof for this using a non-trivial one to one correspondence

$f:M \rightarrow f(M)$ (integration transform).

Let E(K,M) and D(K,C) be the encryption and decryption, respectively, of ECB mode of a safe block cipher. In E(K,M) we adjust the length of M by zero padding. At this time, the encryption and decryption are defined by $C = E(K, f(M_0))$ and $f^{-1}(D(K, C))$, respectively, where M_0 denotes the zero padding of M. In what follows, let Length(M) be the number of bits of any message M.

Theorem 1. If the block cipher $E(K, M_0)$ is safe then $E(K, f(M_0))$ is also safe.

(Proof) For some algorithm A, suppose $A(E(K, f(M_0))) = M_0$, then we have

$f(A(E(K, M_0))) = f(A(E(K, f(f^{-1}(M_0))))) = f(f^{-1}(M_0)) = M_0$. Therefore the original block cipher E is also unsafe.

Theorem 2. $E(K,f(M_0))$ is safe against an insider attack.

(Proof) Let $C = E(K,f(M_0))$ and we assume this is not safe against insider attacks. Note $\text{Length}(M_0) = \text{Length}(C)$. Since $\text{Length}(K) > 0$, we have $\text{Length}(M_0) + \text{Length}(K) > \text{Length}(C)$ and note that C contains the information of K and M_0. Therefore, the attacker must use a compression algorithm, so we cannot decrypt C by the compatible decryption program. This is a contradiction.

As mentioned above, client users of the disaster recovery system are generally vulnerable to potential insider attack by the system vendor.

However, by making use of the method proposed here, this kind of vulnerability can also be avoided by using the integration mode of AES or ECB mode.

VII. Conclusion and Future Issues

We have presented several types of 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. In the conventional disaster recovery system mentioned above, we assumed the use of push-type client nodes which have ID information registered in advance in both the data center and the supervisory center. However, it would be preferable to increase the number of potential users to support the corresponding network services. For this purpose, client nodes such as those in a mobile cell-phone network can be utilized for the network services, even if their IP addresses are changeable with time. To effectively realize this scheme both the data center and the supervisory center register each list of fragmented file information relating to each client node ID with a changeable IP address, in preparation for a recovery request from the user. We should further examine the performance of the Web server as the number of fragments increases.

Since the formal protocol analyzers cannot include the reverse engineering for all the cryptographic primitives, they cannot correspond to all the proposed insider attacks.

In contrast, we have proposed the integration mode of block ciphers, and this provides verifiable security against all the proposed insider attacks.

Acknowledgment

This work has been partially supported by the study (Issue number:151) of the National Institute of Information and Communications Technology (NICT) of Japan.

References

[1] N. Miyaho, Y. Ueno, S. Suzuki, K. Mori, and K. Ichihara, "Study on a Disaster Recovery Network Mechanism by Using Widely Distributed Client Nodes," ICSNC 2009, pp. 217-223, Sep., 2009.

[2] Y. Ueno, N. Miyaho, S. Suzuki, and K. Ichihara, "Performance Evaluation of a Disaster Recovery System and Practical Network System Applications," ICSNC 2010, pp. 195-200, Aug., 2010.

[3] S. Suzuki, "Additive cryptosystem and World Wide master key," IEICE technical report ISEC 101(403), pp. 39-46, Nov., 2001.

[4] N. Miyaho, S. Suzuki, Y. Ueno, A. Takubo, Y. Wada, and R. Shibata, "Disaster recovery equipments, programs, and system," Patent. publication 2007/3/6 (Japan), PCT Patent :No.4296304, Apr., 2009.

[5] K. Kokubun, Y. Kawai, Y. Ueno, S. Suzuki, and N. Miyaho, "Performance evaluation of Disaster Recovery System using Grid Computing technology," IEICE Technical Report 107(403), pp. 1-6, Dec., 2007.

[6] S. Kurokawa, Y. Iwaki, and N. Miyaho, "Study on the distributed data sharing mechanism with a mutual authentication and meta-database technology," APCC 2007, pp. 215-218, Oct., 2007.

[7] S. Tezuka, R. Uda, A. Inoue, and Y. Matsushita, "A Secure Virtual File Server with P2P Connection to a Large-Scale Network," IASTED International Conference NCS2006, pp. 310-315, Mar., 2006.

[8] R. Uda, A. Inoue, M. Ito, S. Ichimura, K. Tago, and T. Hoshi, "Development of file distributed back up system," Tokyo University of Technology, Technical Report, No.3, pp. 31-38, Mar., 2008.

[9] Y. Deswarte, L. Blain, and J. C. Fabre, "Intrusion tolerance in distributed computing systems,"Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on , pp. 110-121, May, 1991

[10] R. A. Fisher and F. Yates, Statistical tables for biological, agricultural and medical research (3rd ed.). London: Oliver & Boyd. pp. 26–27, 1948.

[11] J. Katz and Y. Lindell, "Introduction to modern cryptography", Principles and Protocols, Chapman & Hall/CRC, pp. 96-106, 2008.

[12] D.R. Stinson, "Cryptography, -Theory and Practice- 3rd edition",Chapman & Hall/CRC, pp. 73-112, 2006.

[13] N.Nojima, "The DRTbox", pp. 1-2, Aug., 2013. http://www.nogic.net/files/AtCloudDRTbox.pdf.

[14] C.J.F. Cremers, "Scyther-Semantics and Verification of Security Protocols", Ph.D. Thesis, Eindhoven University of Technology, pp.11-12, 2006, ISBN 90-386-0804-7. ISBN 978-90-386-0804-4.

[15] R.J. Anderson, "The eternity service", Jun., 1997. http://www.cl.cam.ac.uk/~rja14/eternity/eternity.html, 2013.8.22.