A Collusion-resilient Hybrid P2P Framework for Massively Multiplayer Online Games

Kazuma Matsumoto
Graduate School of Informatics
Kyoto University
Kyoto, Japan
kazuma@net.ist.i.kyoto-u.ac.jp

Yasuo Okabe
Academic Center for Computing and Media Studies
Kyoto University
Kyoto, Japan
okabe@i.kyoto-u.ac.jp

***Abstract*—Massively Multiplayer Online Games (MMOG), on which many users play simultaneously in a large scale virtual space on the Internet, have long been popular services. Most of MMOGs work based on Client / Server (C/S) model. Although C/S model is easy to manage games, it lacks scalability since the capacity of storages and the performance of servers that needs to be maintained by administrators increases in proportion to the number of users. In order to solve the problem of scalability, it is expected that development of a P2P-based MMOGs framework takes the place of the conventional C/S model.P2P MMOGs has a disadvantage that cheats are easily performed compared to the C/S model. There are various studies to prevent cheats on P2P models, but research on frameworks considering the possibility of collusion cheats has been rarely done so far.**

In this paper, we propose a collusion-resilient hybrid P2P framework for MMORPG. We first analyze the features of RPG and we clarify the requirements. We investigate cheats which may be caused in a P2P framework and we categorize cheats into three. Then, we consider about the influences and detectability of them. In order to prevent the three types of cheats, it is necessary to calculate data concealed from users on the process of events. Therefore, we suppose that there are "scramble" that encrypt data in order to conceal what the data means and that we can calculate the scrambled data without decryption. We have devised a method using secret sharing based on Chinese Remainder Theorem that has almost all properties for scramble, and discuss its use in the hybrid P2P framework. We have compared the proposed framework to conventional frameworks and a framework with ideal "scramble," with respect to the resistance against cheats.

Keywords—MMORPG; P2P; secret sharing; cheat-proof; collusion

I. INTRODUCTION

Massively Multiplayer Online Games (MMOG), on which many users play simultaneously in a large scale virtual space on the Internet, have long been popular services. MMOG is defined as" Many users connect to a server or another user's client machine via the Internet and play game service in the same virtual world" [1].

It is one of typical game service models served on the Internet. Attractions of MMOG includes, playing with many other users in one virtual world, feeding character for long time and collecting items, and cooperation and competition with other users. Requirements are derived as, simultaneously play by many users and synchronization in real time, strong tamper resistance of each user's data, and fair competition.

In most commercial MMOGs, systems are constructed based on the client server (C/S) model. In this model, servers perform the process of game progress, and clients only receive input from users and process output video and audio to users. Servers also store and manage each user's personal data and various data which constructed a virtual world. In the C/S model, a manager can manage game data and ensure security easily. But the C/S model has serious problem of scalability. Loads tend to be concentrated on servers and there is concern about suspension of service. Further, storage capacity increases linearly as the number of users increases. We can compensate these problems by large-scale investment, but the cost to maintain facilities for game service would become too much for individuals or grass-roots providers.

To solve the scalability problem, P2P based MMOGs have been researched [2][3]. On P2P model, each user performs the process of game progress and send other users processed data at each step. We could solve the scalability problem if each user performed the process of his own game progress and stored his personal data. However, there is a fatal shortcoming that a malicious user can perform cheats easily on P2P model. Thus currently there is no commercial MMOGs based P2P model. Studies to prevent cheats in the P2P model are also being conducted [4]. However, there is no framework whose robustness is equivalent to the C/S model. P2P framework has been required which has resistance for cheats equivalent to C/S model.

According to our survey of previous researches, it is found that three types of cheats, tampering and obstructing events of processing game progress, knowing unauthorized information, and tampering storage data at local clients, are serious problems on MMOGs based P2P framework. We are no longer able to ignore the threat of organized cheats due to the underground spread of Real Money Trade (RMT), which violates the regulations, for selling and buying game items or accounts. Hence assuming collusion of malicious users, we considered influence and detectability of those cheats.

In this paper, we revealed that a hybrid P2P framework is effective to prevent those three cheats. We need a protocol that can be calculate while keeping data concealed to P2P clients (we

0730-3157/17 $31.00 © 2017 IEEE
DOI 10.1109/COMPSAC.2017.10

called this feature as "scramble") to realize the framework. We have evaluated the resistance of the framework using scramble against cheats. We propose a hybrid P2P framework using secret sharing which has similar characteristics of scramble. Comparison of the resistance against those cheats among frameworks of previous research, the proposed framework and the ideal scramble framework indicates that the proposed framework is superior to frameworks of previous researches against some important cheats.

The structure of this paper is as follows. In Section II, we explain related researches about P2P MMOGs. In Section III, we explain scramble and considered resistance against cheats of framework using scramble. We describe the proposed framework using secure function evaluation based on secret sharing in Section IV. In section V we evaluate and compare the proposed framework with frameworks of previous research.

II. Previous Research

A. Research on P2P MMOGs

Frameworks proposed in previous research are classified into the following two models in general.

- Server-centric model
- User-centric model

Server-centric model is a method to let a small number of privileged users manage and change game data. Privileged users act the server role, namely, get action message from other users, update game status and send the status to users who need it. A privileged user is called a super node. Selection of super nodes is often decided according to specific requirements such as preferentially selecting users with a large network bandwidth [5] [4]. Super nodes are in charge of game state in smaller world (be called region) than the whole virtual world. Each node is commonly in charge of each region [5] [6]. There are some disadvantages in this model like concern of overload, failure of super nodes, etc., but among them, the concern is notable that information is gathered on the super node, where it is easy to conduct illegal activities.

User-centric model is a method that each user gets messages directly from other users who make actions and update game state data that each user holds. In this method, all users are responsible for both server and client [7]. Disadvantages of this method are that network load of each user is heavier than C/S model. It has also a disadvantage that malicious users can tamper game data easily and it is hard to detect such cheats.

B. Research on Cheat Countermeasures

There has been many cheats even with conventional MMOGs. These cheats cannot be prevented even if we use C/S MMOGs. Although it is difficult for game administrators to prevent cheats in real time, it might be possible to detect cheats by verifying game logs and user reports afterwards, so that the administrator can give cheat user punishment to suppress cheats. The purpose of cheats often was just mischief or to gain advantage in a game several years ago, but the situation is changing. According to the spread of RMT, malicious users often do cheats to get financial gains. There are many groups which do cheats efficiently. Currently there are no effective countermeasure to suppress the increase of cheats. We considered following cheats can be easily done with P2P MMOGs.

1 Cheats by knowing unauthorized information

2 Tampering and interfering normally game event processes

3 Tampering game data in local storages

Cheats in [2] can be categorized as follows.

2A) Attack to change proxy calculation results to another intended value

2B) Attack to interfere game process by sending incorrect proxy calculation results

2C) Attack to interfere game process not to send proxy calculation results

Cheats by knowing unauthorized information is different from other cheats in the point that it cannot be detected. That cheat is never logged because malicious users only scan packets or data. Thus, we cannot detect this cheat. We consider it is suboptimal method to construct a framework to prevent cheats by knowing unauthorized information. In this paper, we target these three cheats.

C. Related Research Using a Majority Vote

In [4], they prevent a user from tampering and interfering correct game event process by storing and calculating same game data on some other clients and by determining results of game process using majority votes. This framework is similar to the sever-centric model. Unlike the server-centric model, some users manage the same data. A super node managing a region is selected among users who are playing in other regions. By doing this, the influence of "Cheats by knowing unauthorized information" is relieved.

However, malicious users may contact each other via SNS and may collude to do organized cheat. We think that it is not necessarily secure to set the threshold for preventing cheat to the half of super nodes managing same game data. Super nodes can know unauthorized information. There is a not a little possibility of cheating that malicious users collude and contact each other across regions.

Table I shows the resistance to cheat we mentioned in Section II-B.

TABLE I. Consideration on Cheats: Framework Used in a Majority Vote

Cheat	Evaluation	remarks
1	weak	Some node can know unauthorized information
2A	fair	It is possible if majority collaborate
2B	fair	It is possible if majority collaborate
2C	fair	It is possible if majority collaborate
3	fair	It is possible if majority collaborate

D. A Proposal of Distributed Management Method for Preventing Malicious Use of Game Information

In [8], the authors clarified malicious use of game information, stated as cheats by knowing unauthorized information. They classified objects in game and information about objects. They also proposed a method in which information and game data are distributed on other nodes distributed so as to prevent the cheats.

However, malicious users can tamper game data in this method. There is also a fear that some users collude so that they can use game information maliciously. Consideration on the resistance against cheats are shown in Table II.

TABLE II. CONSIDERATION ON CHEATS: FRAMEWORK USED IN THE DISTRIBUTED MANAGEMENT

Cheat	Evaluation	remarks
1	fair	Attackable by collusion
2A	weak	Attackable by one user
2B	weak	Attackable by one user
2C	fair	Attackable by collusion
3	fair	Attackable by a user but the user cannot know what the data means

III. PROPOSED FRAMEWORK

In this section, we propose a collusion-resilient hybrid P2P framework. It is based on the framework using a majority vote described in Section II-C. We describe stepwise the details of our idea as follows.

A. Simple Use of Majority Votes

Figure 1 shows a simple framework which is a modification of the one stated in [4]. X is data for User A, and is encrypted as $enc(X)$ by a key a, which the control server holds. Each step of game progress at User A is processed by multiple proxy clients which calculates the same game events and that result is decided by a majority of them. Even if one of the proxy clients tries to tamper the game data or the result of processing events, or even when he does not send results other clients, he cannot disturb the progress of the game.

1. User A accesses to the control server to login game service.
2. A sends $enc(X)$ to the control server.
3. The control server decrypt $enc(X)$ to X by the key a.
4. The control server sends X to proxy clients B, C, and D.
5. A sends input data to B, C, and D.
6. B, C, and D process input data and return the results to A
7. A decides the majority as the correct result.

There are two problems in this method. If majority of the multiple clients collude, they can cheat easily. The proxy clients can know unauthorized information of the game process for A.

B. Concealing Data by Scrambling

We consider *scrambling* to prevent cheats by knowing unauthorized information. Here scrambling means making it difficult to know what the data indicates. It is desirable if we can perform calculation without descrambling data. Figure 2 shows the framework using scramble.

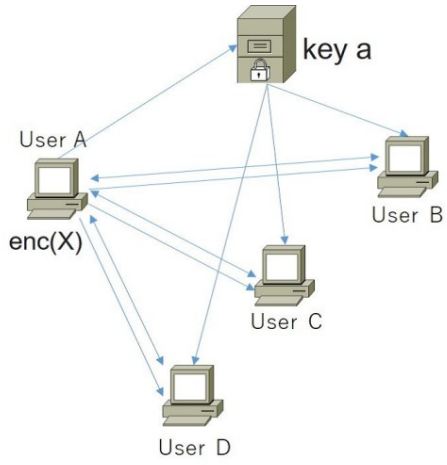

Fig. 1. A simple framework using a majority vote

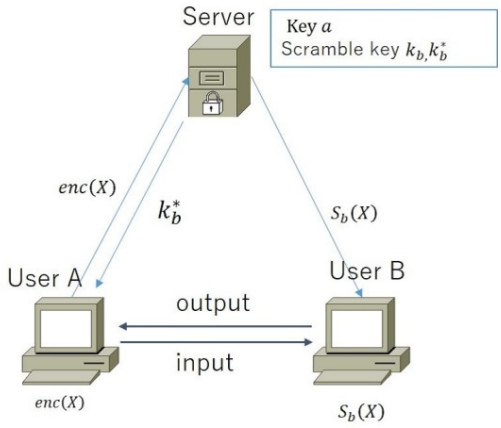

Fig. 2. A framework using scramble

User A stores his own data as $enc(X)$ in encrypted form. $enc(X)$ is decrypted at the control server and then X is scrambled by scramble-key k_b. $S_b(X)$ is sent to User B. User B receives scrambled input data from User A, calculates game events and sends back scrambled results to User A. The control server sends descramble-key k_b^* to User A. User A can scramble input data and decrypt scrambled results by descramble-key.

1. User A accesses to the control server to login game service and send $enc(X)$ to the control server.
2. The control server creates a scramble-key k_b and descramble-key k_b^* and send k_b^* to user A.
3. The control server decrypts $enc(X)$, scrambles data X and send $S_b(X)$, which is scrambled data of X, to

User B

4. User A scrambles input data and sends it to User B
5. User B receives input and calculate it. User B sends scrambled results to User A. User A descramble the results from B by descramble-key k_b^*.

In this framework, User B cannot do cheats by knowing unauthorized information and tampering game data in local storages.

C. Combination of Scramble and Majority Vote

The framework in the previous section can prevent User B from tampering with data into intended value since User B cannot know what the data sent from User A means. However, that framework cannot prevent User B from interfering game process by sending incorrect proxy calculation results nor disturbing game process by not sending proxy calculation results. Thus we consider a framework combining scramble and majority vote. We show concept of the framework in Figure 3.

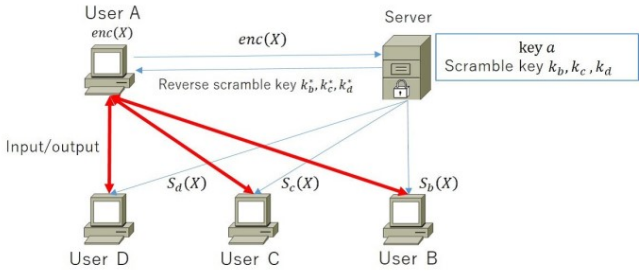

Fig. 3. A framework using scramble and majority vote

The control server sends scrambled data by each different scramble-key to User B, C, and D, and ask each proxy client to calculate events. User A scrambles input data and sends it to User B, C, and D. Even if majority of the proxy clients collude, malicious clients can hardly change scrambled data into some other intended value since User B, C, D have differently scrambled data by distinct scramble-keys each.

Table III shows the resistance against cheats described in section II-B.

TABLE III. CONSIDERATION ON CHEATS: COMBINATION OF SCRAMBLE AND MAJORITY VOTE

Cheat	Evaluation	remarks
1	strong	Unable to attack
2A	strong	Unable to attack
2B	fair	It is possible if majority collude
2C	fair	It is possible if majority collude
3	strong	Unable to attack

D. Consideration on Scramble

Ordinary encryption satisfies the requirement of scramble on concealing what the data means. Additionally scramble requires the following properties.

- Calculating data while data are kept concealed
- Maintain low latency which does not impeding game play

Minimum set of calculations required for MMORPGs are as follows.

- IF-THEN expression
- Updating game variable value
- Addition and multiplication

Homomorphic encryption is an encryption method on which calculation can be done secretly [9] [10]. Fully homomorphic encryption, one of homomorphic encryption schemes, can calculate addition and multiplication on. Thus fully homomorphic encryption satisfies the requirement of scramble on concealing what the data means [11]. However, homomorphic encryption need a lot of time in calculation. Within the authors knowledge, there looks no fully homomorphic encryption scheme which can maintain low latency not impeding game play.

In next section, we modify the framework shown in Section III-C. Using secret sharing and multi-party protocol, we indicate that we can constructed a framework which is as secure as the framework using ideal scramble. Further, the framework used secret sharing and multi-party protocol takes lower time than the framework using homomorphic encryption.

IV. SECURE FUNCTION EVALUATION BASED ON SECRET SHARING

A. Secret Sharing

Secret Sharing Scheme is an encryption method which keep information secret by distributing fragments called shares among clients. In secret sharing, splitting and sharing a secret give a high resistance against loss and leakage of secrets [12]

Supposing there is a secret s. Secret holder wants to avoid lost and leakage the secret. To reduce the possibility of loss of secret s, we share s by multiple coworkers. However, this increases the possibility of leakage. To prevent leakage of secret s, we suppose that secret s was split into sA, sB , sC as shares. s cannot be restored unless all shares are collected. This reduce the possibility of leakage of the secret. However, this again increases the possibility of lost. Thus secret s is split into n of shares so that one can restore s if and only if he collects at least k of the shares. This is called (k, n)-threshold secret sharing.

B. Secure Function Evaluation

Secure function evaluation is a calculation method on an encryption scheme. Suppose we hold secrets S_A and S_B. Secure function evaluation let one perform addition S_A+S_B or multiplication $S_A S_B$ while S_A and S_B are kept secret to him.

We explain secure function evaluation based on secret sharing. Suppose share holders A, B and C have each share

s_A, s_B, s_C, t_A, t_B, and t_C, where a secret s is split into s_A, s_B, s_C and secret t is split into t_A, t_B, t_C. Value u is calculated $s+t$ and split to u_A, u_B and u_C as shares. If we can calculate $u_A = s_A + t_A$, $u_B = s_B + t_B$, $u_C = s_C + t_C$, this is called additive homomorphic encryption on the secret sharing. The multiplicative homomorphism is similarly defined as additive homomorphic encryption.

C. Secret Sharing on Chinese Remainder Theorem

1) Chinese Remainder Theorem.

Chinese Remainder Theorem is stated as follows. Let m_1, m_2,, m_r be positive integers which are relatively prime and a_1, a_2, , a_r be any integer. The following expression holds.

$$x \equiv a_1 \bmod m_1$$
$$x \equiv a_2 \bmod m_2$$
$$\ldots$$
$$x \equiv a_r \bmod m_r$$

This expression has only solution x. Furthermost, under these conditions, the system has a unique solution in Z/MZ, where $M = m_1 m_2 \cdots m_r$.

D. Secret Sharing on Chinese Remaindering

We can use Chinese Reminder Theorem for secret sharing. Suppose the number of share is n. Let $m_0 < \cdots < m_n$ be positive integers which are relatively prime. Let k satisfy $m_0 \cdot m_{n-k+2} \cdots\cdots m_n < m_1 \cdot m_2 \cdots m_k$ and $2 < k \leqslant n$ (k, $n \in N$). k is the threshold. Secret S is an integer in Z/m_0Z. Let an integer α be $S + \alpha m_0 < m_1 \cdots m_k$. Let S_i be $S + \alpha m_0$ mod m_i. $I_i = (S_i, m_i)$ for i=1, …, n are shares of S.

$$x \equiv S_{i_1} \bmod m_{i_1}$$
$$x \equiv S_{i_2} \bmod m_{i_2}$$
$$\ldots$$
$$x \equiv S_{i_k} \bmod m_{i_k}$$

By the Chinese remainder theorem, since $m_{i_1}, \cdots; m_{ik}$ are relatively prime, the system has a unique solution S_0 modulo $m_{i_1} m_{i_2} \cdots m_{ik}$. The secret S is the reduction modulo m_0 of S_0

In this method, the integer α make difficult to infer secret [13].

E. Scramble Based on Secret Sharing

The framework used scramble and multiparty-protocol described in Section III-C can be modified by using secret sharing based on Chinese remainder theorem. Scramble-key and descramble-key become unnecessary and the control server sends a share of the data of User A to each share holder (SH). Each share holder calculates events and the shares of results is sent to User A. User A restores the results from shares not less than the threshold k. Some SH may collude each other but they cannot tamper the secret to an intended result unless the number of malicious SH is not less than the threshold k. The framework was showed in Figure IV.

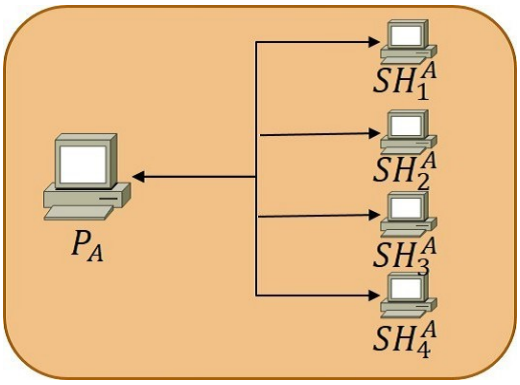

Fig. 4. A framework based on secret sharing

TABLE IV. CONSIDERATION ON CHEATS: COMBINATION OF SCRAMBLE AND MAJORITY VOTE

Cheat	Evaluation	rmarks
1	fairly strong	Attackable with collusion of SHs not less than threshold k
2A	fairly strong	Attackable with collusion of SHs not less than threshold k
2B	weak	Incrrect shares may cause error in restoration of the results.
2C	fairly weak	Attackable with SH collusion os SHs more than threshold $N-k$
3	fairly strong	Attackable with collusion of SHs not less than threshold k

TABLE V. COMPARIZON OF THE FRAMEWORKS

Cheat	Proposed framework	Majoriti vote	Distributed management	Scramble
1	farly strong	weak	fair	strong
2A	farly strong	fair	weak	strong
2B	weak	fair	weak	fair
2C	farly weak	fair	fair	fair
3	fairly strong	fair	fair	strong

The resistance to cheats about this framework was shown in Table IV. It should be noted on Cheat 2B. If incorrect value is mixed in shares, restoration of a correct secret is not easy. It may take a lot of time in examining the combination of correct shares.

V. COMPARISON

Comparison of the resistance of the frameworks described in Section II-C, II-D. III-C, and VI-D against cheats is shown in Table V.

Talking of Cheat 1, 2A and 3, the proposed framework is better than the frameworks based on previous researches with a threshold difference. In contrast, resistance against Cheats 2B and 2C is weaker than that based on previous researches. However, those cheats are not attacks by which malicious users can get advantages directly but just potential concern that the

game progress is interfered or disturbed. Thus the priority of resistance against Cheat 2B and 2C is relatively lower than Cheats 1, 2A and 3. The proposed framework is superior to frameworks based on previous researches with respect to the three serious cheats.

VI. Conclusion

In this paper, we have analyzed the features of MMORPGs and we have clarified the requirements of them. Requirements of MMORPGs are characterized as, many users simultaneously play and synchronize in real time, strong tamper resistance to data of each user, and fair competition. We have classified cheats in P2P MMORPGs. We have clarified the fear of organized cheats in MMOGs by malicious users who are driven money making. Then, we have argued a collusion-resilient P2P Framework needed to prevent cheats. Survey of previous researches leads consideration that requirements of calculating data with concealment and of maintaining low latency not impeding game play for such a framework. With an assumption of "scramble," we have shown that we can construct a collusion-resilient hybrid P2P framework for MMORPGs. We have also proposed a framework based on secret sharing and multi-party protocol which have almost the same strong resistance against collusion cheats.

We have evaluated the resistance of the proposed framework using secret sharing and compared with that of frameworks based on previous researches and that of the framework using scramble We have shown that the proposed framework is superior to previous researches in three important cheats.

There are some other related works. In [14], which is an enhancement of [2], the authors propose techniques (1) to reduce end-to-end event delivery latency by dynamically replacing nodes in the tree, and (2) to efficiently deliver events to users who have visible areas over multiple game areas. In [15] the authors presents techniques for trust enforcement that use cryptographic methods and are adapted to the dynamic membership and resources of P2P systems. There have been many known techniques for distributed calculation under P2P environment where malicious users may collude. For example calculation of random numbers can be implemented by applying so-called global coin flipping protocols[16].

Acknowledgment

We would like to express our sincere gratitude to Dr. Shuichi Miyazaki at Kyoto University for his comments and suggestions on this work.

References

[1] Kengo Nakajima. The Technology of Online Games, Gijutsu-hyohuron Co. Ltd., 2011. ISBN-10: 4774145807.(in Japanese)

[2] Keiichi Yasumoto, Katsuhiko Kouhara, Minoru Ito. A Load Distribution Mechanism for Multi-player Games on P2P Networks. In Proceedings of IPSJ .DPS Workshop, Vol. 2003, No. 19, pp. 79–84, Dec 2003.

[3] Ignasi Barri, Concepció Roig, Francesc Giné. Distributing Game Instances in a Hybrid Client-Server/P2P system to Support MMORPG Playability. Multimedia Tools and Applications, Vol. 75, No. 4, pp. 2005–2029, 2016.

[4] Minoru Kawahara, Keiichi Endo, Yutaka Takahashi, Cheat Prevention for Massively Multiplayer Online Distributed Services. IPSJ Journal, Vol. 47, No. 4, pp. 1087–1098, 2006.

[5] Marios Assiotis and Velin Tzanov. A Distributed Architecture for MMORPG. In Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support for Games, NetGames '06, New York, NY, USA, 2006. ACM.

[6] Hiraku Yoshioka Hiroyuki Ebara Raito Matsuzaki. Dynamic Region Decomposition Method in Large-scale Virtual Space. IPSJ Technical Report. SIG, Vol. 2012, No. 7, pp. 1–8, 2012.

[7] Shintaro Enomoto, Naoshi Sakamoto. An Agreement Protocol for P2P MMOG That is Tolerant of Entering and Leaving . IEICE Technical Report, NS, NetworkSystem, Vol. 111, No. 408, pp. 41–46, 2012.

[8] Yuzo Taenaka, Kotaro Yamazaki, Yasushi Wakahara, Proposal and Evaluation of Distributed Management Method for Preventing Malicious Use of Game Information in P2P Massively Multi-player Online Role Playing Game. IEICE Technical Report. IN,InformationNetwork, Vol. 111, No. 469, pp. 79–84, Mar 2012.

[9] Michael Naehrig, Kristin Lauter, Vinod Vaikuntanathan. Can Homomorphic Encryption Be Practical? In Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop, pp. 113–124. ACM, 2011.

[10] Junfeng Fan, Frederik Vercauteren.. Somewhat Practical Fully Homomorphic Encryption. IACR Cryptology ePrint Archive, Vol. 2012, p. 144, 2012.

[11] Tomoki Kaminabe, Takahiro Sawaki and Masahiro Tsuboi. Implementation of Operational Functions on the Cloud Using Fully Homomorphic Encryption. Master's thesis,Faculty of Information Sciences and Engineering, NANZAN University, 2014.(in Japanese)

[12] Hiroshi Doi. Secret Sharing and Application of It. Integrated Science of Information Security , Vol. 4, pp. 137–149, 2012.(in Japanese)

[13] Sorin Iftene. General Secret Sharing Based on the Chinese Remainder Theorem. IACR Cryptology ePrint Archive, Vol. 2006, p. 166, 2006.

[14] Shinya Yamamoto, Yoshihiro Murata, Keiichi Yasumoto, Minoru Ito, An Event Delivery Mechanism for Multi-player Games on P2P Networks with Load Balancing and Short Latency, IPSJ Journal, Vol 47, No. 2, pp. 475-483, Feb. 2006..

[15] Adam Wierzbicki, Trust Enforcement in Peer-to-Peer Massive Multi-player Online Games, Lecture Notes in Computer Science, Vol 4276, pp.1163-1180, 2006.

[16] Kunikazu Yoda, Yasuo Okabe, Masanori Kanazawa. An Unbiased Global Coin Flipping Protocol on Synchronous Distributed Systems, IEICE Transactions on Information and Systems Vol.E84-D No.1 pp.40-47, 2001,