FreeRank: Implementing Independent Ranking Service for Multiplayer Online Games

Li Tang
Tsinghua University, Beijing, China
tangli99@tsinghua.org.cn

Jun Li
Tsinghua University, Beijing, China
junl@tsinghua.edu.cn

Jin Zhou
Tsinghua University, Beijing, China
zhoujin@gmail.com

Zhizhi Zhou
Renmin University, Beijing, China
zhizhi.zhou@gmail.com

Hao Wang
singhua University, Beijing, China
freizo@gmail.com

Kai Li
singhua University, Beijing, China
likai@datangmobile.cn

ABSTRACT

Ranking is necessary for multiplayer online games to provide players with self-complacence and reference for choosing game counterparts. Most existing ranking solutions are tightly coupled with game applications of client-server architectures. In this paper, a novel scheme named FreeRank is proposed as a ranking service independent to specific architecture and detailed implementation of each individual game application. Based on a certificate-based framework and a reputation-based score-computing algorithm, FreeRank resolves the challenge of cheating prevention. Preliminary analysis and simulation results show that FreeRank is feasible and effective.

Categories and Subject Descriptors

K.8.0 [**General**]: Games, C.2.1 [**Network Architecture and Design**]: Ranking Service.

General Terms

Management, Design, Security

Keywords

Multiplayer Online Games, Ranking Service, FreeRank

1. INTRODUCTION

Ranking is the scheme to calculate (accumulated game) scores and evaluations of players according to their game histories. Almost all multiplayer online games (MOGs) require ranking service to provide players with self-complacence and reference for choosing game counterparts. Traditional ranking solutions are designed tightly coupled with or embedded in the client-server architecture of each individual game application, which suffer several deficiencies. Firstly, a player community of the same game is isolated into multiple unrelated parts by different game service providers. A player cannot combine his or her scores and histories on several different battle nets. Secondly, existing ranking solutions are unable to count game results played in local area networks (LAN). Many popular MOGs, such as Starcraft and Counter Strike, support game play in LAN, but players complain that their efforts in LAN do not contribute to their rank or overall scores on battle nets. Last but not least, existing ranking solutions work only in the game applications of client-server architectures but fail to support other application architectures such as peer-to-peer or hybrid architectures which are proposed in literatures to improve the scalability and availability, and to reduce investments of MOGs.

It is noted that MOGs are moving towards a large-scale virtual world [1] in which interconnected players are expected to be able to play games anywhere with all sorts of instruments, e.g. PC, X-BOX, PDA, and mobile phones. As existing ranking solutions are no longer capable for these situations, an independent ranking service is desired to take all game results into account for the evaluation of players, no matter the games are played through a game server on certain battle net, in LAN, on Internet, or just with mobile phones using Blue-tooth technology. Obviously, such a ranking service has potential to significantly improve the availability and scalability of MOGs.

With these problems in mind, we designed FreeRank, a novel scheme for a ranking implementation as a platform independent service for MOGs. The rest of this paper is organized as follows: Section 2 presents the problem and identifies technical challenges. Section 3 describes the certificate-based framework to implement FreeRank. Section 4 details the reputation-based score-computing algorithm for cheating prevention. Section 5 evaluates FreeRank's feasibility and effectiveness by simulation. Section 6 discusses several application scenarios of FreeRank and other related issues. Section 7 concludes the paper and presents some future work.

2. PROBLEM STATEMENT

The main focus of this paper is the design and implementation of FreeRank, an implementation architecture of independent ranking service for MOGs, which is not involved in or restricted by the application architecture or detailed implementation of any individual game application. It works well at the absence of any trusted arbiter or central authority when games are being played. It can count all game results played anywhere with various instruments as long as the players are interconnected with each other. In existing ranking solutions, there are always game servers authenticating players and handling ongoing game states. In contrast, FreeRank does not require central authoritative entities all the time, which implies more data have to be cached on

NetGames'05, October 10–11, 2005, Hawthorne, New York, USA.

player's game-instrument. The requital is that FreeRank can improve the availability and scalability of MOGs and provide uniform rank or overall scores to player communities.

The key challenges to FreeRank design are: (1) avoiding misuse of the ranking service, and (2) preventing malicious players from obtaining unfair advantages by cheating. Although this seems peddling by security experts, it is critical for a specific game to retain its players. As a malicious player could be powerful enough to crack and tamper anything stored on his or her computer, including files, memory, and drivers [2], FreeRank has to be capable to prevent cheating by impersonation, escaping during games, and modifying, forging, deleting or reusing local data. Additionally, FreeRank should also prevent particular cheating strategies against the ranking service, ensure fairness among players, and reflect the player's actual ability measured with his or her game achievement.

3. CERTIFICATE-BASED FRAMEWORK

FreeRank is based on a certificate-based framework to meet the challenges. In this section, a high level overview is presented firstly to introduce the precondition and some main terms used in the framework. Then pivotal processes are then described in details. At last, cheating threats and security issues related to the framework are analyzed.

3.1 Overview of Framework

Fig.1 shows a sketch of the certificate-based framework. It contains two parts, the global ranking server or *FreeRank Server* (*F-Server*), and the agent at each player's terminal or *FreeRank Agent* (*F-Agent*).

Figure 1. FreeRank's Architecture.

Every player is identified by a *playerID* and possesses a pair of public key and private key. There are two kinds of certificates for each player, *identity certificate* and *ranking certificate*, both issued by *F-Server*. The former certifies player's profile, e.g. *playerID* and public key, and the latter certifies player's game achievement.

FreeRank can be assimilated to a financial system of MOGs, where *F-Server* plays the role of a bank and *F-Agent*s play the role of players' accountants. In order to decouple from specific rules for each game application, FreeRank leaves the task of judging game results to game applications and separates a bout of multiple-player game into several two-party deals. For example, if player *A*, *B*, *C*, *D* decided to play a game together, each player's *F-Agent*, say *A*'s *F-Agent*, will treat the game as three separated deals with each of the other players, say *B*, *C*, and *D*, respectively.

Whenever some interconnected players want to play a game together, their *F-Agent*s reveal profiles and achievements of the other players. If all the players are satisfied, *F-Agent*s help them to negotiate an agreement confirming the game's validity. If the negotiation is successful, each *F-Agent* notifies its associated game program residing in the same terminal to start the game. When the game is over, the game application informs *F-Agent* who should pay or be paid how much according to the game result. *F-Agent* reckons accounts by signing and collecting bills. If there were players escaping during the game, others have privilege to accuse escapers with the agreement as proof. *F-Agent* integrates all bills and accusing information of the same game play into a *statement*, which will be submitted to *F-Server* immediately, or cached locally if communication to *F-Server* is unavailable at the time and resubmitted when connection to *F-Server* is established later. On receiving the *statement*, *F-Server* processes it and certifies up-to-date *ranking certificate* for the submitting player.

The player's game achievement in his or her *ranking certificate* is represented by *score*, *rank* and the number of deals in which he or she was involved, won, lost, ended with a draw, accusing others and accused by others respectively in tuple format: *(total, win, lose, draw, accusing, accused)*. While *rank* represented by a series of grades, such as junior, senior, and professional, gives a most general evaluation of the player's ability, *score* and tuple *(total, win, lose, draw, accusing, accused)* demonstrate the player's playing history in statistics. *F-Agent* provides reference of another player's reputation by revealing his or her *accusingRate* ($^{accusing}/_{total}$) and *accusedRate* ($^{accused}/_{total}$), where a high *accusingRate* implies the player is aggressive and a high *accusedRate* implies the player is likely to act shamelessly.

In order to prevent a player from reusing the same bill or agreement repeatedly, every bill and agreement contains a unique *gameID* corresponding to each player. It may be most straight forward for *F-Server* to record all ever used *gameID*s of every player, which however is obviously unscalable as the number of *gameID*s keeps growing up. Instead, in FreeRank, every *F-Agent* only maintains an *AgentIDThreshold* of its own largest *gameID* the player ever used, and the *F-Server* also maintains a *ServerIDThreshold* for each player recording the largest *gameID* of the *statement*s ever counted for the player.

3.2 Agreement Negotiation

To negotiate an agreement, every participant's *F-Agent* increases its *AgentIDThreshold* and then sends the value to all others as the player's *gameID* to be used in the agreement. When a player's *F-Agent* has received all other participant's *gameID*s, it sorts *(playerID, gameID)* tuples in the descending order of *playerID* and composes an agreement. Afterwards, every participant's *F-Agent* signs the agreement with its private key and exchanges signature. Then *F-Agent*s store the signed agreements and notify the game program to start the game.

It is quite important to keep fairness during the signature exchange, because a malicious player may swindle another's signature but refuse to offer his or her own and then slander honest players. Despite many fair exchange protocols have been proposed [10], it is proved that there is no perfect solution without a trusted third party [4], implying that it is impossible to completely prevent slandering behavior in FreeRank's scenario. To restrain slandering players, two effective mechanisms can be employed. Firstly, the signatures are exchanged in the descending order of participant's *accusingRate*, where it assumes player's behavior possesses some consistent characteristics (those who seldom accuse others are more likely to behave well during the exchange). Secondly, we propose a reputation-based score-computing algorithm to be detailed in the next section.

3.3 Reckoning Accounts

When a game is over, *F-Agent* is informed of the result by the game program. Every renter (the benefiting player's *F-Agent*) requires each of its debtors (the losing player's *F-Agent*) to sign a bill confirming the game result. The bill contains the renter's *playerID*, the same *gameID* as in the agreement, the debt count (in this paper, it indicates whether the renter wins or has a draw) and the debtor's signature. If some dishonest players cheated, escaped during the game or refused to sign their bills, it is exactly the agreement that the offended players can use to accuse dishonest players. *F-Agent* composes bills and accusing information of the same game into a final *statement* and signs with its private key. Then *F-Agent* attempts to submit all cached *statement*s to *F-Server*. Note that a malicious player will get no advantage by deleting his or her locally cached *statement*s before submitting them to *F-Server*, because *statement*s are cached by renters who are going to be benefited.

3.4 Issuing Ranking Certificate

Besides after each game, *F-Agent* also periodically attempts to submit cached *statement*s to *F-Server* and requests the latest ranking certificate for the player. The cached *statement*s must be submitted in the same order as they are created. On receiving the batch of *statement*s, *F-Server* first checks the submitter's signature to prevent malicious players impersonating others; then it checks the signatures in bills and agreement to avoid forged *statement*s; at last, *F-Server* checks whether the submitter's *gameID* in the *statement* is larger than his or her *ServerIDThreshold* to prevent repeated counting. After verification procedures, valid *statement*s are passed to *F-Server*'s score computing engine, which afterwards updates achievements of the players referred to in the *statement*s. The score computing engine also computes the submitter's latest score and updates his or her *ServerIDThreshold*. Finally, *F-Server* signs a new ranking certificate and sends it along with a confirming message back to the submitter's *F-Agent*, which then replaces old ranking certificate and removes cached *statement*s that matches the confirming message.

3.5 Cheating Threats and Security Analysis

So far, the certificate-based framework has carefully considered mechanisms to prevent the malicious players impersonating others or obtaining unfair advantages by modifying, forging, deleting or reusing locally cached data. Compared with existing ranking solutions, however, the most difficult problem of FreeRank is how to prevent the escaping behavior without a trusted entity during the game.

Despite the proposed the accusing mechanism punishes escapers, it, on the other hand, is likely to be misused by malicious players to slander others and leads to a rather unexpected aftermath. To slander other players, a malicious player pretends to negotiate an agreement with those players, but accuses them maliciously as soon as he or she gets their signatures, though the defendant players did nothing wrong. Although slandering behavior is restricted by the signature exchange regulation proposed earlier in this paper, it is not a perfect solution. Imagine the scenario where two malicious players, say *P* and *Q*, are playing games and assume *P*'s *accusingRate* is lower. Unfortunately, *Q* can hardly be alerted of being slandered by *P* due to confusion of other cached *statements* or temporarily unable to connect to *F-Server*. In this case, *P* is able to slander *Q* until their *accusingRate*s become equal. Afterwards, *P* stops playing with *Q*, finds another player *R* whose *accusingRate* is higher and continues to slander *R* in the same way. As long as *P* can find another player whose *accusingRate* is higher, he or she can get unfair advantages by slandering. Theoretically, this problem may lead to an unpleasant situation where every player possesses a rather high *accusingRate* and keeps looking for chances to slander others. In worse case, when *accusingRate* is near to 100%, it increases slower and slower as accusing time increases. The situation will misguide behavior of the whole player community. To solve the problem, we propose a reputation-based score-computing algorithm in the next section to restrict and discourage escaping and slandering behaviors and lead the player community towards favorable behaviors.

4. SCORE-COMPUTING ALGORITHM

A player's score in ranking certificate is calculated in our algorithm according to his or her achievement tuple *(total, win, lose, draw, accusing, accused)* with the following equation:

$$score = W \cdot win + L \cdot lose + D \cdot draw + L \cdot Punish + W \cdot Compen \quad (1)$$

where *W*, *L* and *D* denote the mark of each game won, lost, or ended with a draw respectively, *Punish* is punishment due to being accused by others, and *Compen* is the player's compensation to the player due to accusing others. *W*, *L* and *D* are configured by the game operators according to the game's specific rules, and they usually hold $W > D \geq 0 \geq L$.

Existing ranking solutions usually just contain the first three factors (win, lose, and draw), because game servers are able to judge and punish escaping players explicitly. However, as there is no trusted arbiter like game servers in FreeRank, the last two factors (*Punish* and *Compen*) are necessary to discourage escaping behavior and compensate offended players. *Punish* and *Compen* should be carefully defined according to the following

principles of the score-computing algorithm. Firstly, behavior of the whole player community should be controllable towards pleasant trend; Secondly, those who are frequently accused should be punished harder to discourage escaping behavior; Thirdly, malicious players cannot get persistent advantages by slandering others; Fourthly, results should be proportion to the game numbers, e.g., if the number of games player *P* has played is twice than that of player *Q*, and they have the same *accusingRate*, *P*'s compensation should be twice than *Q*'s. According to these principles, *Punish* and *Compen* can be defined as follows.

Given a player *P* possesses his or her current achievement tuple as *(Pt, Pw, Pl, Pd, Pag, Pad)*, we define player *P*'s *Punish* denoted by *Ppn* in equation (2):

$$Ppn = \begin{cases} Pad \cdot (\frac{Pad}{Pt \cdot T})^S, & if \ \ Pad < Pt \cdot T \\ Pad, & otherwise \end{cases} \qquad (2)$$

where *S* (*S*>0, usually selected as a natural number) and *T* (0<*T*<1) are constant parameters. It is understandable that equation (2) alleviates punishment against those who are seldom accused and possess smaller *accusedRate*s than *T*.

To make behavior of the whole player community controllable, we introduce two parameters that can be configured and adjusted by the game operator according to characteristics of particular game rules and the practical situation, namely, *E* denoting expected *accusingRate* of player community, and *U* denoting upper limit of *accusingRate*, which means any player whose *accusingRate* is larger than *U* would be taken as completely slandering player thus denied from obtaining any compensation. Given a recent time window, let *Av* denote average *accusingRate* of all players, *M* denote an intervenient value between *E* and *Av* which may be calculated with certain interpolating splines, e.g. median of *E* and *Av* used in this paper, we define the player *P*'s *Compen*, denoted by *Pcp* in equation (3), where *K* (*K* >0, usually being a natural number) is a constant.

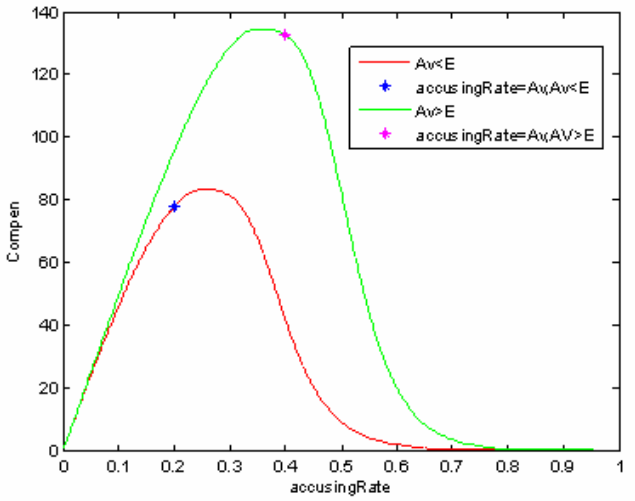

Figure 2. Variation of Compen.

When player *P*'s *accusingRate* is no larger than *M* i.e. $\frac{Pag}{Pt-Pag} < \frac{M}{1-M}$, it implies that *P* has behaved well so far, his or her current accusing will be trusted and adds to *Compen*. As player *P* keeps on accusing others, *Pag* and *P*'s *accusingRate* grow up, and the increasing speed of *Pcp* slows down gradually till stopping when his or her *accusingRate* equals to *M* i.e. $\frac{Pag}{Pt-Pag} = \frac{M}{1-M}$. If player *P* continues to accuse others, his or her *Compen* decreases on the contrary because he or she is considered to be slandering. In this case, even if player *P* is a malicious player, he or she has to stop slandering as we expected in the third principle. At last, because every player tends to adapt his or her *accusingRate* towards *M* to make *Compen* reach the peak, the average *accusingRate* of the whole player community *Av* would also tend to reach *M*. Additionally, *M* is the intervenient value between *E* and *Av*, thus *Av* tends to reach *E*. It is exactly what is expected according to the first principle that we expect. In fact, if *E* is set to be a little less than *Av* (e.g. 0.9 *Av*), the algorithm becomes self-adaptable.

$$Pcp = \begin{cases} (Pt - Pag) \cdot \left[\frac{M-1}{2 \cdot M} \left(\frac{Pag}{Pt-Pag} - \frac{M}{1-M} \right)^2 + \frac{M}{2 \cdot (1-M)} \right], & if \ \ \frac{Pag}{Pt-Pag} \le \frac{M}{1-M} \\ \frac{Pt-Pag}{1+(\frac{Pag-M \cdot Pt}{M \cdot (Pt-Pag)})^K} \cdot \left[\frac{(1-U)^2 \cdot M \cdot (1-M)}{2 \cdot (U-M)^2} \left(\frac{Pag}{Pt-Pag} - \frac{M}{1-M} \right)^2 + \frac{M}{2 \cdot (1-M)} \right], & if \ \ \frac{M}{1-M} < \frac{Pag}{Pt-Pag} < \frac{U}{1-U} \\ 0, & otherwise \end{cases} \qquad (3)$$

It is easy to find that given player *P*'s *accusingRate* is determined, his or her *Compen* defined in equation (3) i.e. *Pcp*, is proportion to the number of games he or she has played, thus the fourth principle proposed above is satisfied. Assume player *P* is going to explore advantages by keeping slandering others, and the value of *(Pt-Pag)* is kept constant. Because *Av*, the average *accusingRate* of all players, changes little as *P* continuously accuses against others, we can ignore the effect on variation of *M* caused by *Pag* increasing and treat *M* as a constant. Fig.2 shows the variation process of *Pcp* as *P*'s *accusingRate* increases.

5. SIMULATION EVALUATION

We performed preliminary simulation experiments, and the results demonstrated the feasibility, effectiveness and capacity of our approaches.

5.1 Simulation Setup

We categorize all players into *G* grades, with each grade containing *S* players, of whom *Ks* percent are slandering players and *Ke* percent are escaping players; every player is randomly designated an accusing probability and an escaping probability; the two probabilities of common players follow normal distribution N(0.05, 0.1) and N(0.1, 0.2) respectively but limited in [0, 1] range, while those of slandering and escaping players follow uniform distribution in [0.6, 1]. *N* games are randomly played among players. As FreeRank takes a bout of game as several two-party deals, every game in simulation just contains two players.

Table 1. Simulation Parameters

Symbol	Description	Value
G	# of grades	5
R	# of players in each grade	200
N	# of games played	100000
Ks	% of slandering players	10%
Ke	% of escaping players	10%
W, L, D	win, lost, draw mark/game	3, -1, 1
T, S	constants of *Punish* in Eq.(2)	0.15, 3
E, U, K	parameters of *Compen* in Eq.(3)	0.9*Av*, 0.99, 3
Rg	# of games in each round	10

Table 1 shows parameter values used in the simulation, and the playing procedure is designed as follows.

1) For each game, randomly selecting two players, say player *P* and player *Q*;
2) The player with lower *accusingRate* say *P* attempts to slander the other one say *Q* according to his or her accusing probability; if *P* determines not to slander *Q*, *Q* attempts to slander *P* according to his or her accusing probability on the contrary; if both *P* and *Q* determine not to slander, the game is played;
3) The game ends with a draw in certain probability (0.05 in our experiment). Otherwise, *P* or *Q* wins the game in proportion to their grades; the loser determines whether to escape or not according to his or her escaping probability; if the loser escapes, he or she is surely accused by the winner, and the loser slanders the winner according to his or her accusing probability; otherwise, the loser pays the bill to the winner;
4) After *N* games have been randomly played among players, every one possesses his or her game achievement. Then, every player starts to take turns to play games with others and plays *Rg* games in each round. At the beginning of each round, the player attempts to slander others, he or she keeps slandering until cannot get benefits any longer, and then the rest of the games in that round are played honestly.

5.2 Experimental Results

5.2.1 *Feasibility to Reflect Player's Actual Ability*

To evaluate feasibility of FreeRank to reflect player's actual ability with his or her *ranking certificate*, we plot score distribution of all players and average score of each grade in Fig.3.

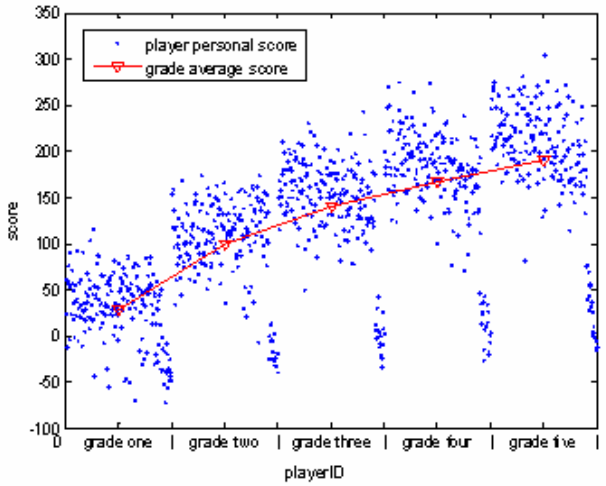

Figure 3. Score distribution of players.

It shows that players of higher grade have obtained higher score, and the average score grows up as the grade increases. We arrange players of the same grade in the ascending order of their accusing probabilities, and find that malicious players get much lower scores than common players of the same grade (see the tail of each cluster). This is mainly because of the following reasons. Firstly, as malicious players frequently attempt to swindle other's agreement for slandering, they waste most of their chances to win than common players, especially for those of high grades, which is similar in real life where malicious players with high *accusingRate* are more likely to be refused by others. The second reason is because *Compen* equation defined in equation (3) notably eliminates the compensation to malicious players.

5.2.2 *Effectiveness to Distinguish Malicious Players*

Fig.4 shows the proportions of average compensation to common players (Cc) and slandering players (Cs), as well as average

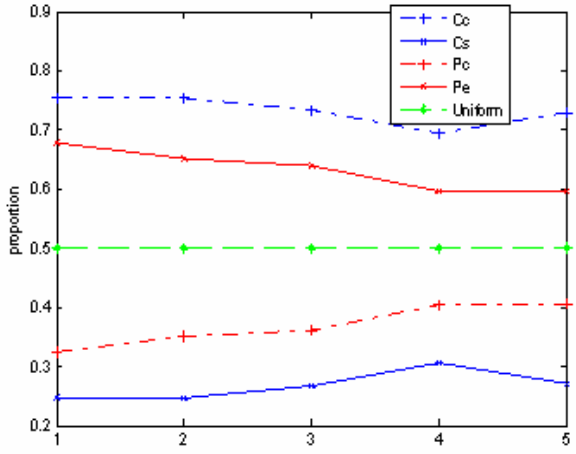

Figure 4. Proportion of compensation and

punishment to common players (Pc) and escaping players (Pe) on average. It demonstrates that our reputation-based algorithm with the definitions of *Punish* in Eq.(2) and *Compen* in Eq.(3) is effective to punish escaping players and reduce compensation to slandering players.

5.2.3 Capacity to Lead Player Community towards Favorable Behavior

We show the evolvements of player's personal *accusingRate* and community's average *accusingRate* in Fig.5. The result proves capacity of the score-computing algorithm to lead the whole player community towards favorable behaviors. Despite we assume all players attempt to explore strategies to obtain advantages in FreeRank, their *accusingRate*s rapidly converge to a rather low value, which indicates all including common players as well as malicious players tend to seldom misuse accusing mechanism and seldom slander others maliciously. It is worthy pointing that the experiment also proves self-adaptability of the score-computing algorithm, because we have set the expected accusing rate of player community E to be $0.9Av$, and the result shows the average *accusingRate* of player community converges to a favorable value as expected gradually.

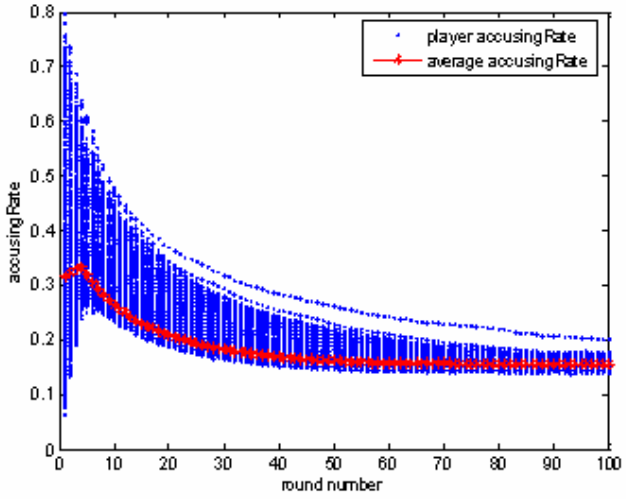

Figure 5. Evolvement of layer's *accusingRate*

6. DISCUSSION

There has been considerable work on adapting MOGs to a large-scale and highly available virtual world. Some researchers improved existing implementation or server distribution by analyzing performance and traffic features of MOGs [5,6,7], and others proposed alternative computing and communication architectures [8,9]. FreeRank proposed in this paper can be used to improve MOGs in but not limited to the following scenarios. Firstly, it can complement existing ranking solutions to account game results played in isolated environment hosted by different service providers. Secondly, FreeRank can cooperate with non-client-server game architectures to improve scalability and availability and reduce investments of MOGs. Moreover, FreeRank makes it possible for game producers developing cabinet games, e.g. funny desktop games, to easily build online player communities.

Compared to existing ranking solutions, security mechanisms proposed in this paper are necessary to FreeRank to prevent cheating strategies and misbehaviors. Although one may argue that most players are honest, we believe every player desires a higher evaluation and has incentive to explore shortcuts, and it is more straightforward for players to cheat against ranking service than to utilize complicated methods [3] against game applications. It is important to note that we are not claiming FreeRank is resistant to all cheating methods. In fact, FreeRank suffers some of the same cheating threats as existing ranking solutions such as cheating by collusion. The security efforts in this paper just aim to enable FreeRank to provide an independent ranking service as fair as existing ranking solutions. Actually, FreeRank allows a player to evaluate his or her opponent in personal opinion, which implies that a player's score reflects the reputation on playing games to some extent. In this case, FreeRank can also be used as a reputation-based player evaluation system for some emerging massively multiplayer online role play games (MMORPG) using peer-to-peer architectures [11,12].

Moreover, *F-Server* will not become a performance bottleneck like MOG servers in existing ranking solutions. *F-Server* only calculates scores and stores player profiles, which requires much less computing capacity and bandwidth than MOG servers that are used to handle ongoing game states. Besides, *F-Server* is not involved when players are playing games, while MOG servers are necessary all the time to existing ranking solutions. Even if *F-Server* is unavailable sometimes, FreeRank continues to work because game results can be cached locally and submitted later. Furthermore, the certificate-based framework and score-computing algorithm are carefully designed to require no much computing and storage.

When a large number of players play the same bout of game simultaneously, FreeRank's signature exchange in the agreement negotiation procedure may cause a scalable problem. Designing more efficient exchange protocols and standard APIs between *F-Agent* and game applications are some interesting directions for future work.

7. CONCLUSION

To the best of our knowledge, FreeRank is among the earliest efforts towards implementing an independent ranking service, which possesses better availability and scalability than existing ranking solutions of MOGs. We identified the challenges of such a ranking service and propose a certificate-based framework to achieve it. We analyzed specific cheating threats against FreeRank and proposed a reputation-based score-computing algorithm to prevent misbehaviors. We also performed initial simulation experiments, demonstrating feasibility, effectiveness and capacity of our approaches. Finally, we discussed several scenarios in which FreeRank can be used to improve or complement existing ranking solutions and some related issues.

We are making efforts on incorporating FreeRank into Freegame [13, 14], a project intending to build a generic network platform based on peer-to-peer technologies for various desktop games and providing ranking, auditing and chatting services. We will attempt to verify effectiveness of FreeRank against more complicated cheating methods in practice.

8. ACKNOWLEDGMENTS

This work was supported by a research grant from NEC Laboratories China. The authors are grateful to all reviewers who provide many valuable suggestions to improve this paper.

9. REFERENCES

[1] What's This World Coming To? The Future of Massively Multiplayer Games. *Game Developers Conference 2002*. http://archive.gamespy.com/gdc2002/mmog/. March, 2002.

[2] Matt Pritchard. How to Hurt the Hackers: The Scoop on Internet Cheating and How You Can Combat It. http://www.gamasutra.com/features/20000724/pritchard_pfv.htm. July, 2000.

[3] Jeff Yan et al. A Systematic Classification of Cheating in Online Games. http://www.cse.cuhk.edu.hk/~cslui/STUDY_GROUP/howgame.pdf. June, 2004.

[4] Even, S. and Yacobi, Y. Relations among public key signature systems. Tech. Rep. 175, Computer Science Department, Technicon, Haifa, Israel, 1980.

[5] Wu-chang Feng, Francis Chang, Wu-chi Feng, and Jonathan Walpole. Provisioning On-line Games: A Traffic Analysis of a Busy Counter-Strike Server.In *ACM SIGCOMM Computer Communication Review.* July, 2002.

[6] Ahmed Abdelkhalek, Angelos Bilas, and Andreas Moshovos. Behavior and Performance of Interactive Mult-player Game Servers. In *Proceedings of* 2001 *IEEE International Symposium on Performance Analysis of Systems and Software.* November, 2001.

[7] Wu-chang Feng and Wu-chi Feng. On the Geographic Distribution of On-line Game Servers and Players. In *Proceedings of Second Workshop on Network and System Support for Games.* May, 2003.

[8] Knutsson B., Lu H., Xu W., and Hopkins B. Peer-to-Peer Support for Massively Multiplayer Games. In *Proceedings of IEEE Infocom* 2004 *Conference.* March, 2004.

[9] Sean Rooney, Daniel Bauer and Paolo Scotton. Building Infrastructures for Very Large Multi-Player Games. Research Report, IBM Zurich Research Lab. December, 2002.

[10] Henning Pagnia, Holger Vogt, and Felix C. Gartner. Fair Exchange. In *Computer Journal, volume 46, number 1, Oxford University Press.* January, 2003.

[11] Solipsis. http://solipsis.netofpeers.net/wiki/HomePage.

[12] Second Life. http://secondlife.com/.

[13] Jin Zhou, Li Tang, Kai Li, Hao Wang, and Zhizhi Zhou. A Low-Latency Peer-to-Peer Approach for Massively Multiplayer Games. In *Proceedings of Fourth International Workshop on Agents and Peer-to-Peer Computing*. July, 2005.

[14] Jin Zhou, Li Tang, Kai Li, and Zhizhi Zhou. FreeGame: A Testbed for Peer-to-peer Techniques in Massive Multiplayer Online Games. Research Report, NEC Laboratories China. November, 2004.