2010

CAPTCHA challenges for massively multiplayer online games: Mini-game CAPTCHAs

Yang-Wai Chow
University of Wollongong, caseyc@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Hua-Yu Zhou
University of Wollongong

CAPTCHA challenges for massively multiplayer online games: Mini-game CAPTCHAs

Abstract

Botting or automated programs in Massively Multiplayer Online Games (MMOGs) has long been a problem in these networked virtual environments. The use of bots gives cheating players an unfair advantage over other honest players. Using bots, players can potentially amass a huge amount of game wealth, resources, experience points, etc. without much effort, as bot programs can be run continuously for countless hours and will never get tired. Honest players on the other hand have to spend much more time and effort in order to gather an equal amount of game resources. This destroys the fun for legitimate players, ruins the balance of the game and threatens the game developer's revenue base as discontented players may stop playing the game. Research efforts have proposed the incorporation of CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) challenges in games to prevent or detect potential cheaters, by presenting challenges that are easy for a human to solve but are difficult for a computer to solve. However, the incorporation of CAPTCHA challenges in games is often seen in a negative light, as they are deemed to be intrusive and that they destroy the sense of immersion in the game. This research presents an approach of using CAPTCHAs in MMOGs that is both secure and adds gameplay value to the game.

Disciplines

Physical Sciences and Mathematics

Publication Details

Chow, Y., Susilo, W. & Zhou, H. (2010). CAPTCHA challenges for massively multiplayer online games: Mini-game CAPTCHAs. 2010 10th International Conference on Cyberworlds, CW 2010 (pp. 254-261). Piscataway, New Jersey, USA: IEEE.

2010 International Conference on Cyberworlds

CAPTCHA Challenges for Massively Multiplayer Online Games

Mini-game CAPTCHAs

Yang-Wai Chow[1], Willy Susilo[2], Hua-Yu Zhou[1]
[1]Center for Multimedia and Information Processing
[2]Center for Computer Security Research
School of Computer Science and Software Engineering
University of Wollongong
Australia
{caseyc, wsusilo, hz285}@uow.edu.au

***Abstract*—Botting or automated programs in Massively Multiplayer Online Games (MMOGs) has long been a problem in these networked virtual environments. The use of bots gives cheating players an unfair advantage over other honest players. Using bots, players can potentially amass a huge amount of game wealth, resources, experience points, etc. without much effort, as bot programs can be run continuously for countless hours and will never get tired. Honest players on the other hand have to spend much more time and effort in order to gather an equal amount of game resources. This destroys the fun for legitimate players, ruins the balance of the game and threatens the game developer's revenue base as discontented players may stop playing the game. Research efforts have proposed the incorporation of CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) challenges in games to prevent or detect potential cheaters, by presenting challenges that are easy for a human to solve but are difficult for a computer to solve. However, the incorporation of CAPTCHA challenges in games is often seen in a negative light, as they are deemed to be intrusive and that they destroy the sense of immersion in the game. This research presents an approach of using CAPTCHAs in MMOGs that is both secure and adds gameplay value to the game.**

Keywords-bots; CAPTCHA; cheating; MMOGs; online games; security;

I. INTRODUCTION

Massively Multiplayer Online Games (MMOGs) are networked virtual environments in which multiple users can interact through a shared sense of presence, with other players and non-player characters as well as with the surrounding environment, within the same virtual game world. MMOGs are usually persistent virtual worlds, in which the game environment remains in existence and changes to the state of the game world will remain even after the player logs off.

In recent years, MMOGs have increased in both complexity and popularity [1]. As more and more players are playing these games nowadays, there is a growing demand for the development of new online games and continual maintenance of existing MMOGs. The rapid growth in the online gaming market world wide has seen the development of MMOGs evolving into a multi-billion dollar industry. This particular type of online games is typically subscription based and players have to pay a monthly subscription fee in order to enjoy the online game services. It has been reported the subscriptions for one of the most popular MMOGs, World of Warcraft [2] has surpassed 11.5 million subscribers in 2008, with an estimated US$150 million in monthly subscription fees [3].

The use of automated programs to play these games, known as botting, has long been a problem that has plagued online games. Botting is rampant even in the earliest MMOGs like Ultima Online [4] [5]. This form of cheating gives botting players an unfair advantage over other legitimate players in terms of time spent playing the game, the gathering of game resources, and so on. Using these automated programs, cheating players can potentially amass a huge amount of game wealth, resources, experience points, etc. in a short period of time without much effort. This destroys the fun for honest players who expect the game to be played on a level playing field, and that other players all play according to the set of rules laid out by the game developers.

In fact, the incentive to cheat using bots is not only limited to gaining advantages within the game world. In the gaming community, the process of killing large numbers of enemies to gain experience points and to earn gold is known as 'farming' [3]. It has been estimated that a large portion of the number of subscription in the MMOG, World of Warcraft, can actually be attributed to farmers rather than genuine players [1]. The term 'gold farming' has also been defined in [6] as 'the production of MMOG virtual currencies, items and services for financial gain'. Virtual resources and currencies in MMOGs can be sold for real world financial gain to players who do not want to spend the time and effort to collect the resources themselves. These players are willing to pay real money to acquire in-game resources or for someone else to level up their game characters. This is an added enticement for farmers to use bots in order to avoid the manual labor of farming, as bots can run for hours on end and never tire.

In that respect, botting can severely impact and disrupt the balance of the game. For example, botting can have a

978-0-7695-4215-7/10 $26.00 © 2010 IEEE
DOI 10.1109/CW.2010.48

significant effect on the level of inflation/deflation in the game economy due to the distribution of an unusual amount of items or resources obtained through botting [3]. The sudden influx of wealth may drive game item prices up because there is so much money in circulation, whereas flooding the game market with a particular resource will drive that resource's price down. This has an adverse effect on legitimate players, who will have to spend much more time and effort in order to gather as much wealth or resources, which will in turn affect their motivation to play the game at all [3].

In light of this, botting game exploits are not only problematic for the players, but game developers potentially stand to loose thousands of dollars in lost revenues from disillusioned players who stop subscribing to play their game [7]. To avoid loosing their customers and revenue base, game developers may have to spend additional time to devise and implement ways of rebalancing the game economy. As such many game developers ban the use of bots and try to implement strategies to prevent and detect the use of bots in their games. However, identifying bots from humans is not an easy task because bots are designed to simulate human behavior and to obey the game world rules completely [8] [9].

The concept of CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) was introduced as a way of distinguishing between humans and computer programs [10]. CAPTCHA challenges, or Human Interaction Proofs (HIPs) [11], are designed to be automated tests that a human can solve and pass the test, whereas a computer program will have difficulty solving it and therefore will not pass. Over the years, text-based CAPTCHAs have become almost ubiquitous on the internet and many companies have adopted its use on their websites for a variety of purposes, such as to prevent fraud and denial of service attacks in online registrations, ticket/event reservations, online voting, chat rooms and weblogs [11]. These CAPTCHAs typically appear as a distorted text string, where the user has to identify the appropriate text characters. Figure 1 shows an example of a text-based CAPTCHA taken from Google [12].

Figure 1. Example of a Google text-based CAPTCHA [12].

A number of researchers have proposed using CAPTCHAs as a security mechanism against bots in online games [7] [13] [14]. However, unlike the typical usage of CAPTCHAs on websites, in MMOGs the CAPTCHA challenge cannot be a one time test at login. Otherwise a human user can always log a bot into the game and let the bot take over after that. As such, multiple tests are required at various times throughout the game [5]. While this can succeed in combating the use of bots, it also has major drawbacks. In particular, periodic tests using traditional text-based CAPTCHAs in the context of online games are intrusive, they interfere with the flow of the game and they destroy the player's sense of immersion in the game.

This gives rise to the question of how to seamlessly incorporate CAPTCHA mechanisms into games without adversely affecting a player's fun or enjoyment of the game. This paper addresses this issue by presenting an approach of using CAPTCHAs in MMOGs in a way that is both secure and adds gameplay value to the game.

II. BACKGROUND

This section gives an overview of a number of different methods used for combating bots, which are implemented in current MMOGs. In addition, this section also presents a background on previous work in other proposed bot prevention and detection techniques including the use of CAPTCHAs.

A. Bot Countermeasures

One of the limitations of bot programs is that they are by no means intelligent enough to sustain meaningful conversations with humans. As such, some MMOGs are designed to maximize player-to-player interactions, and by monitoring inter-player conversations it is possible to identify clusters of human players and to try to distinguish them from bots. This approach does not require any additional human intervention, other than from the players themselves. However, as it may result in a high number of false positives and false negatives, additional detection methods still have to be used [13].

Other game developers rely on other forms of human interaction to detect bots, by encouraging and providing mechanisms for players to report suspicious in-game characters. Game moderators, also known as game masters, can approach these characters and try to start conversations with them in order to determine whether a human is actually playing the game [3]. Game masters can also roam around the game world and question suspicious players [9]. However this approach is subject to abuse, as disgruntle players can easily make false accusations by reporting on players that they do not like. In addition, it is a laborious task and inefficient especially for game worlds with thousands of simultaneously online characters [3].

Some developers attempt to incorporate process monitoring to check for known bot programs running on the player's system [15]. The developers of World of Warcraft included an automated program within their game client, which tries to determine whether the client system is cheating. This program, called Warden, is embedded in the game client and reads all sorts of data from the user's system [3]. The purpose of Warden is to try to check for suspicious programs, such as bot programs, by monitoring data and processes while the game client is being run and reports this back to the servers. However, there are several limitations with this approach. For instance, it can only perform signature checks for known programs and is always a step behind bot writers. Furthermore, players have developed a way to work around the Warden by starting the game in guest mode, thus preventing Warden from accessing

processes at higher privilege levels. Probably the most important drawback with the Warden that has raised many objections to its use is that it is a clear invasion of the user's privacy, to the extent that it has been labeled by some as spyware [3] [15].

B. Related Work

1) Monitoring Techniques

With the increase in the number and variety of bot programs as well as the problems that these program pose, researchers have proposed a number of ways to combat such programs using variety of automated approaches. Chen et al. [8] propose a technique of monitoring network traffic and identifying discrepancies in the traffic patterns. In their study, they examined the network traffic from a number of perspectives including the speed and regularity of response times and traffic burstiness (variability of traffic volume). By analyzing the network traffic of a particular online game, they observed that there are major discrepancies in the traffic patterns between human players and bots [8].

Other bot detection methods include player behavior analysis. In [9], the researchers speculate that the frequencies at which bots perform certain actions will be distinguishably higher than that of human players. In view of the fact that bots typically concentrate on repeated use of particular action types, their proposed methodology automates the process of classifying bots by analyzing game logs to identify discrepancies in action frequencies and action types from human players [9].

Bot detection based on movement analysis has also been investigated. This method is based on the assumption that bots typically follow a prescribed or previously recorded list of waypoints, while humans generally roam freely in the game world. Mitterhofer et al. [3] proposed a mechanism to reconstruct a character's route based on character movements, extract waypoint information by clustering high density locations and to use this information to identify repetitions in the movement sequence. In [16], the researchers perform a simple test to classify bots based on variations in the frequency of different movement angles between bots and humans.

Other researchers have examined anti-cheating approaches via the collection and analysis of user input data. Gianvecchio et al. [5] performed a series of experiments by collecting user input for a number of user input actions; namely, keyboard keystrokes, mouse pointing, clicking, point-and-click, drag-and-drop, and pauses, and comparing these with the input characteristics produced by bot programs. They analyzed this information using a neural network to identify bots. Based on their work, they put forward a bot defense system based on what they called human observational proofs [5]. A different approach to detecting the use of programmatically generated mouse clicks and keyboard strokes input data attacks has been investigated in [17].

2) CAPTCHAs

The use of CAPTCHA tests has also been proposed for preventing bots in online games. This approach is slightly different from the other approaches in that its purpose is to ensuring human presence. Human presence means that a bot cannot play the game completely unsupervised [7]. Unlike the previously discussed approaches, such as monitoring player behavior and user input, which are processes that run continuously during the game, each CAPTCHA test is a single instance test. In other words, this means that even though multiple CAPTCHA tests may be presented to the player at random intervals over the course of the game, it is still possible for the game to be played by a bot program, but every time a CAPTCHA test is presented a human would have to solve the CAPTCHA in order to pass the test. This would in effect cause the value of using bots to decrease considerably as every bot would have to be supervised by a human player or risk detection by consistently failing the tests [13]. An advantage of this approach is that it requires less computational resources as compared to methods that require the continual monitoring of the characteristics of thousands of players.

However, there are a number of usability issues with the current text-based CAPTCHAs [18]. In the attempt to make CAPTCHAs secure against automated machine attacks, a variety of techniques have been proposed to make CAPTCHAs harder to break. The example in figure 1 shows distorted text characters that are linked together to prevent segmentation attacks. This can sometimes make it difficult for humans themselves to decipher. Other forms of CAPTCHA tests like image-base CAPTCHAs [19] have also been developed along with 'physical' CAPTCHAs that require the use of specialized hardware input devices, which may be inexpensive but requires the player to purchase the appropriate device [13]. In [20], the researchers presented the design of a CAPTCHA system for the Nintendo DS handheld gaming device.

One of the most prevalent arguments against the use of CAPTCHAs in online games is that, in its traditional form, presenting CAPTCHA tests to players during the actual gameplay is both distracting and disruptive to the flow of the game. In addition, it destroys the suspension of disbelieve that game developers attempt to create in virtual game worlds.

III. MOTIVATION

The proposal to implement CAPTCHA challenges in games to prevent and detect bots is often seen in a negative light. The periodic presentation of these tests is deemed to be intrusive, irritating to legitimate players and spoils the fun of the game.

Nevertheless, some of the fairly recent games such as Fallout 3 [21], BioShock [22], Mass Effect [23], etc. have incorporated gameplay features such as 'lock picking' and/or 'hacking' as a part of the game. These are mini-games within the overall game that the player has to solve in order to gain some sort of reward upon successful completion. For example, the player might encounter a locked vault or a password protected computer system in the game, they are then presented with a mini-game challenge if they chose to pick the lock or hack the computer. If they complete the challenge successfully they can then access the contents of

the vault or the computer. However, if they are unsuccessful the game might invoke some sort of penalty. For instance, the player might have to expend some resources in order to attempt the challenge (e.g. the pick might break and there might be a limited number of these), other penalties could include the computer short-circuiting causing the player to loose health, the alarm going off, etc.

These mini-games can also include timing challenges in which the player has to solve the mini-game within a limited amount of time or even challenges that require the player to time their actions accordingly. Furthermore variations to the mini-games are added by randomizing certain aspects of the mini-game's challenge. These challenges are also typically graded from easy to difficult, with the rewards adjusted appropriately based upon the perceived level of difficulty. Some games may even allow the player to level up their skills as they progress through the game. In other games, the game developers design the game in a way that limits the skills and abilities required to solve these mini-game challenges based on the class of character that the player chooses to play.

Initially when the player first plays the game, he/she has to learn how to solve/complete the challenge. This might initially be tricky as they will not be familiar with the mechanics behind the challenge. However, once they grow accustomed to the controls/actions available and become more skilled at overcoming the challenge, to the player the mini-games then become a regular part of the game. In that respect, rather than being seen as a hindrance these mini-games actually add to the number of in-game actions and tasks that the player can perform and enjoy. Furthermore, it also adds to the overall complexity of the game and is seen as an attractive/unique gameplay feature. In fact, game developers are always looking for ways to add more gameplay value to their games and to distinguish their games from other game titles. These mini-game challenges require extensive design, creative visual and input interfaces in order to make them usable and enjoyable.

This research is based on the fact that it is possible to add value to a game by incorporate mini-game challenges. As such, it is also possible to use mini-games for a different purpose, which is to ensure human presence in online games. By designing appropriate CAPTCHA challenges as mini-games in the game, these can be seamlessly implemented in an online game without destroying the immersion or suspension of disbelief in the game.

IV. FRAMEWORK

A. Mini-game CAPTCHA Challenges

One of the common tasks present in most MMOGs is the collection and gathering of resources. Resources could be in the form of weapons, armor, money, experience or even material to craft items, etc. The developers of the game typically implement a number of ways in which resources can be gathered. This usually involves a series of repetitive actions or tasks in the game. For example, in a fantasy game like World of Warcraft, this might entail hunting and killing mobs/monsters which will drop resources for the player to pick-up and collect [3]. Bot programs are often written to automate this repetitive series of actions, in other words, to automate the 'farming' process. The resources that are dropped by the monsters are typically random. Most of the time the player will get normal drops of low value (e.g. various raw material required to craft an item), but occasionally there is a small chance of getting good drop which has a high in-game value (e.g. a whole item).

This is where the CAPTCHA challenge can be designed to come in. In view of the fact that there is a random chance of good drops, developers can implement CAPTCHA challenges as mini-games to coincide with this. This can easily be explained to fit and blend into the game setting and genre [13]. For instance, in something like a fantasy game, this can be explained along the lines of monsters occasionally securing their belongings using magical locks. This makes sense as the item may be of a higher value than normal drops. In order for the player to access the locked resources/items, they will have to break the spell currently in place. Deciphering the spell will involve completing a mini-game (i.e. solving a CAPTCHA test). If successful, the player will get the resource, if not the resource will be destroyed by the magical spell guarding the resource.

This is similar to the approach already implemented in some of the current games as mentioned in section 3 of this paper, but for the purpose of securing the game against bots. If implemented as part of the game, once players become accustomed to these mini-games, they will not directly break the flow of the game as it will be seen as a regular part of the gameplay. In addition, there is that excitement and anticipation in the reward that the player expects to receive upon successfully solving the CAPTCHA.

B. Design Considerations

The security mechanism for flagging whether a player is a potential cheat needs to give some allowance for some incorrect responses. It should only flag a potential cheat if a particular player consistently fails the CAPTCHA challenge. When that happens, other measures can be taken to investigate this further. For example, flagged players can then be approached by the game moderator, who will try to start a conversation with these players. In a game world with thousands of simultaneous players, this will help to filter out and narrow down potential botting cases to a small subset of the players.

To add value to the gameplay and to make the mini-games more appealing and interesting, mini-games can be made to increase in difficulty, proportional to the level of the player, and developers can also allow players to level up their mini-game skills, etc. which will aid them in solving the mini-game challenges. Gameplay has been defined as providing the player with a series of interesting choices [24]. So another approach will be to force the player to expend some kind of resource when attempting the CAPTCHA challenge. Using up more resources might make the challenge easier and faster to solve but costly for the player. In this manner, players will have to decide from themselves how they want to approach the problem. This could also have a bearing on the reward that the player can potentially

get. The faster the player solves the mini-game, the higher the chance of getting a better reward.

This timing mechanism can also be used to flag potential cheats. In situations where a bot is used to play the game, the bot program can be written to alert a human whenever the CAPTCHA challenge appears, so that the human can solve the challenge then allow the bot to continue the game. If the human is not actually at the controls, it will take some time for them to respond. This way, client systems that consistently take a long time to respond to CAPTCHA challenges can be flagged as potential cheats. In addition, the in-game reward that they will get will not be of a significant value. If the potential rewards for solving the CAPTCHA challenges fast are substantial, the incentive for using bots will decrease significantly, as it means that the 'supervised' bot approach will not yield any good resources.

To avoid players becoming irritated with the CAPTCHA challenges, developers might allow players the option of turning off these mini-games challenges or to occasionally not respond to them. However, again if the rewards for completing the challenges are potentially substantial, a normal legitimate player would always want to get the maximum benefit and would thus try to solve the mini-game to the best of their abilities. Therefore for the majority of legitimate players who choose to play the mini-games and who correctly solve these most of the time, these players can be ruled out of the suspicious players list.

Another important issue that needs to be considered is that these mini-game CAPTCHA challenges should not be presented to the player during fast pace game sequences (e.g. when the player is in the middle of battle) where the player may be in danger of dying or when the player is required to perform a time critical task. One of the general principles of game design is that the pace in a game should vary, and fast pace periods will generally be followed by slower paced phases [24]. CAPTCHA challenges should only be presented during these slow paced periods.

V. Implementation and Analysis

A. Implementation

A mini-game CAPTCHA challenge was designed and developed using the framework detailed in the above section. The basic idea behind this mini-game challenge is that the player is presented with a display area where the full contents are not completely revealed. As the player points, clicks and moves the mouse cursor, random pixel splats will appear around the mouse cursor. Sections hit by the splats will gradually be revealed. This is somewhat akin to spraying paint onto objects in a scene. Initially there does not appear to be anything in the display area, but as the player 'sprays paint' in the appropriate areas, the symbols/characters will start to be uncovered. Figure 2 gives a depiction of this.

This mini-game notion was inspired by the concept of 'emerging images' in [19], where seemingly random splats were used to render a scene. A computer program would not be able to obtain much useful information by segmenting the image and examining the local splats, but a human who sees the image as a whole will be able to perceive the subject in the scene. So in our case, even before the full scene is revealed, the player can still perceive the content if the player concentrates the splats in the appropriate locations.

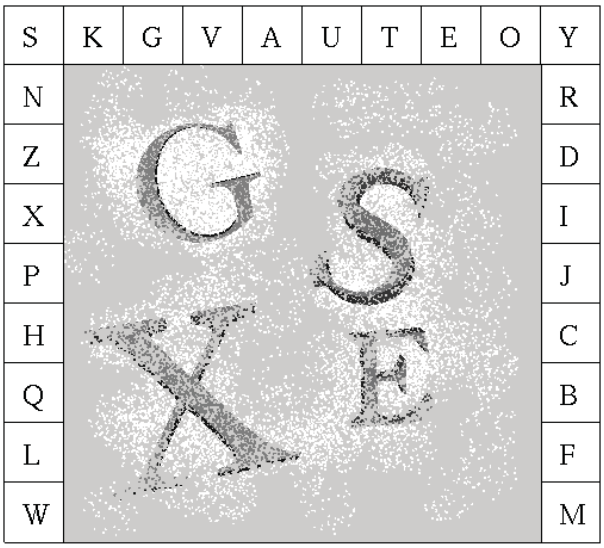

Figure 2. The player has to uncover the hidden characters.

The implementation of our mini-game CAPTCHA challenge is a general mini-game as we did not design it for any specific game or genre. As such we used a generic set of upper-case Roman letters as the set of possible symbols/characters. These characters were randomly generated and orientated. In an actual game, a professional game artist would be employed to design the symbols, interface, background, etc. to make it look more appealing, and the symbols would be designed to fit the appropriate genre. In order to blend in with the game world, this CAPTCHA approach can easily be explained as sprinkling magic dust on a scroll to reveal the hidden spell, or spraying a certain substance for a chemical reaction on the material surface to uncover the hidden combination, etc. For the rest of the paper, we will describe this approach as 'sprinkling magic dust' which is suitable for a fantasy game genre.

In our implementation, to correctly complete the CAPTCHA challenge the player has to work out the contents in the display area, then select which letters were present in the display area by clicking on the correct letters in the interface surrounding the display area. This can be seen in figure 2. The respective locations of the letters surrounding the display area were randomized to avoid predictability, especially for a pixel reading bot program.

To add gameplay value, players are also provided with the option of sprinkle completely random splats into the entire area. This will normally be done until they can identify areas of interest. Once they can identify sections of potential interest, they can then choose to refine their sprinkling to those sections. To make the mini-game more interesting, the player is timed as to how long it takes him/her to complete the challenge. The amount of resources expended is also recorded. In this manner, the more resource the player uses

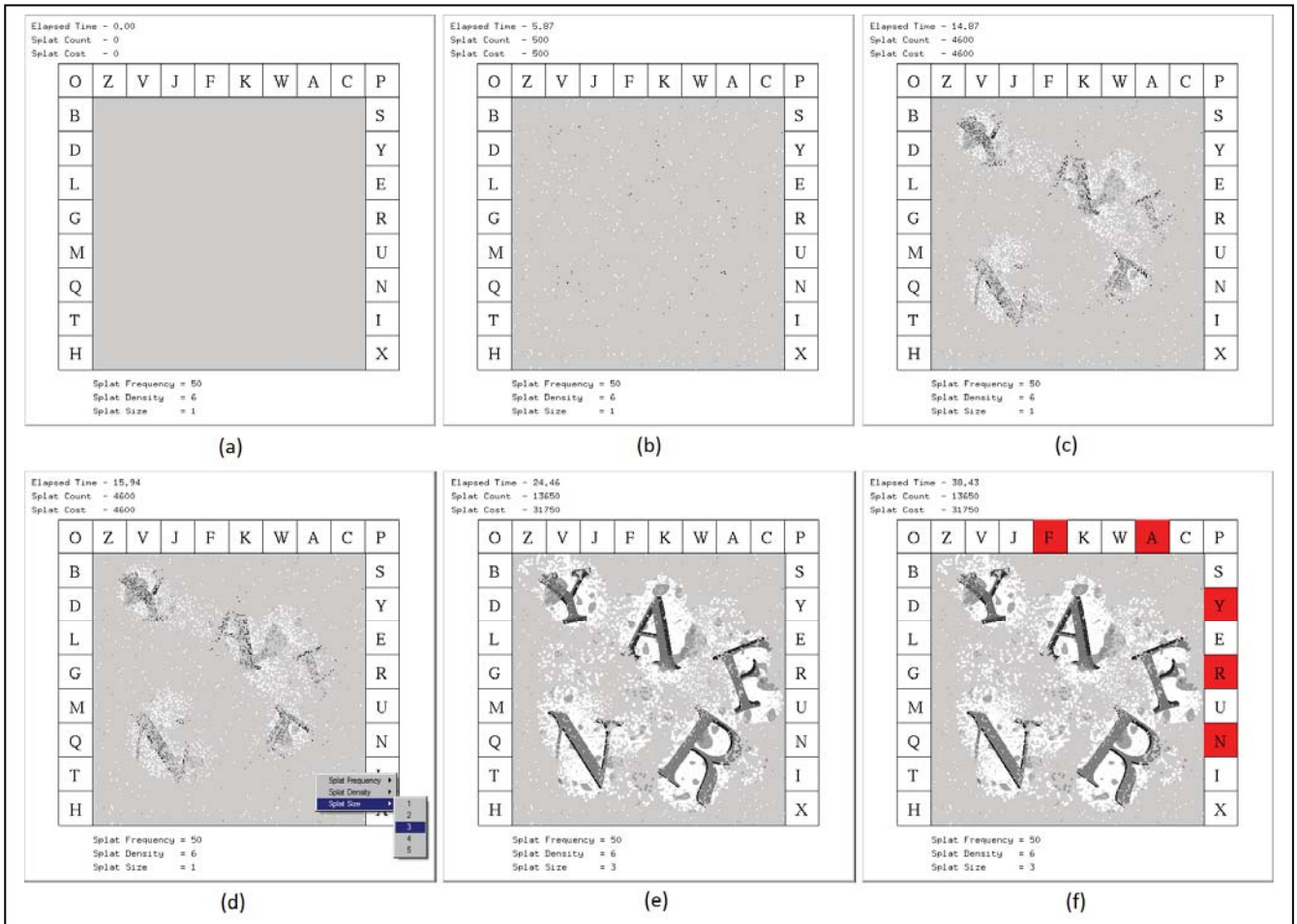

Figure 3. Example of player interaction: (a) characters are initially hidden from the player; (b) player chooses to use random splatting over the entire content area to identify areas of interest; (c) local splatting concentrated around perceived areas of interest; (d) player chooses to use larger splat sizes to speed up the process; (e) most of the text is revealed using the larger splats; (f) player selects the correct characters.

up the easier it might be to solve the CAPTCHA, but it will also be more expensive if a cost is assigned to the splats/magic dust.

The players can also upgrade their available tools by increasing the frequency of the splats (splats per unit time), the density of splats (splats per unit area) or the size of the splats. All of which fill a greater number of pixels more rapidly, thus making it easier and faster to solve the CAPTCHA. A variety of other variations are also possible such as using splats with larger color contrasts, 'magnetic' splats which will be attracted towards the important content and so on. All these can be explained in a game setting as upgrading the player's set of skills and abilities, or purchasing better spells or better grade magic dust, etc. As an in-game cost will be assigned to the different options, the total cost will be calculated based upon the developer assigned weightage formulas. Therefore the player will have to assess the pros and cons of using a particular approach, which makes for interesting gameplay choices. Figure 3 gives a step-by-step illustration of how a player might interact with the mini-game CAPTCHA. The actual contents of the display area in figure 3 are shown in figure 4. One can see the background clutter used in the CAPTCHA as per normal CAPTCHA practices.

Figure 4. Actual image of the CAPTCHA in figure 3.

The difficulty level of the challenge can always be increased using standard text-based CAPTCHA techniques, like randomizing the number of symbols/characters presented per challenge, distorting the content, transforming the contents in various ways, adding more background/foreground clutter, reducing color contrasts, etc. The set of possible symbols/characters can also be increased to make the recognition challenge more difficult. These approaches will also make the CAPTCHA more secure against automated machine attacks. Figure 5 shows examples of two variations of the image in figure 4 that can be used to make the mini-game more interesting. These will probably be more challenging to the player due to the hollow interiors and lower color contrasts.

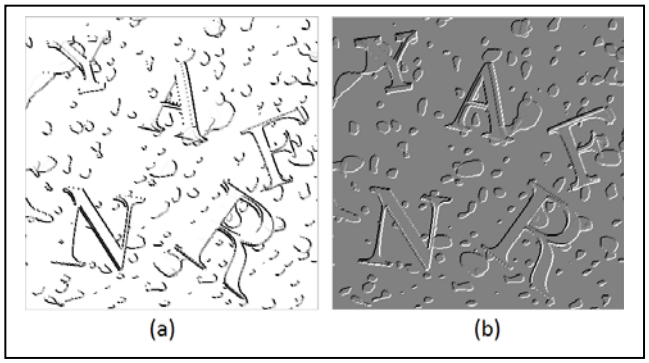

Figure 5. Other more challenging CAPTCHA image options, both are variations of the image in figure 4 (a) edge image; (b) emboss image.

B. *Security Analysis*

The implementation of mini-game CAPTCHA challenges in online games will force a human player to be present to solve the CAPTCHA. The CAPTCHA contents itself will be generated and randomized on the server. This will then be sent, at appropriate random intervals over the course of the game, to the client system which will present the challenge to the player. The player will have to solve the CAPTCHA and their answers will be sent back to the server which will validate their responses. Based on the client system's timing and responses, the behavior of the player can be monitored. If there are any abnormalities with the player's responses, the player will be flagged as a potential bot. This automates the process of bot prevention and detection rather than relying solely on the human game master interaction approach.

As the game client is run on the player's own system, one cannot assume that the game software is completely secure. It is entirely reasonable to assume that the player can hack the game client. In order to deter potential hackers, the response time that players take to respond to the CAPTCHA will also be recorded and sent to the server. In this manner, the response time will not only be used for timing gameplay challenges and affecting player rewards, but it will also be used for security reasons. If the player consistently solves the CAPTCHA too fast or too slow, they will be flagged as being suspicious.

Furthermore, while the data concerning the CAPTCHA challenge that is sent to the client system may be encrypted, it is possible for the player to find ways of intercepting and interpreting the contents of network packets. This will possibly allow them to use automated CAPTCHA machine attacks to solve the CAPTCHA (assuming that the image processing and recognition attacks can solve the CAPTCHA as fast as a human), or alternatively they can cheat by viewing the entire CAPTCHA display which is supposed to initially be hidden from the player.

To safe guard against these situations, a screenshot of the CAPTCHA's display area can be taken when the player solves the CAPTCHA and this can be sent back to the server. In addition, the player's mouse movements during the mini-game CAPTCHA challenge period can also be recorded, sent to the server and analyzed on the server's side. The server can then determine whether or not the player actually attempted the CAPTCHA challenge legitimately based on the screenshot and by correlating player mouse movements with the information in the screen capture. This way, even if the player manages to hack the network messages, they will still have to take the time to complete the mini-game by going through the splatting process. In the end, hacking the CAPTCHA will not give them much gain over legitimate players.

The advantage of doing this unlike in previous work is that relevant information regarding the player's behavior only needs to be collected and analyzed for short periods of time during the CAPTCHA test. Contrary to previous behavior analysis techniques that require continuous monitoring of a wide range of player activities, this approach will free up much more of the server's computational resources for performing other tasks.

C. *Limitations*

This research is concerned with the prevention of bots in MMOGs by ensuring human presence. In other words, it forces a human to be in the loop. While human presence might seem like a weak property to assert, as a bot can always be used to play the game with some human supervision, by adjusting the appropriate rewards for the mini-game CAPTCHA challenges, the value of using bots will decrease considerably. If every bot has to be supervised by a human player, and if the rewards obtained by a botting approach are not substantial, this would significantly defeat the purpose of using bots in the first place [13].

Nevertheless, this approach cannot prevent human gold farmers or 'gaming workshops' that employ human players to farm in-game resources then sell this for real world currencies [6]. These gold farms are typically hosted in countries where wage labor is cheap. In the case of these human gold farms, no amount of automated detection methods to distinguish humans for bots will be able to prevent such practices.

VI. CONCLUSION AND FUTUREWORK

The use of bot programs is becoming increasingly problematic in MMOGs as it adversely impacts the game for legitimate players. Game developers typically ban botting

practices and have tried many ways to prevent and to detect the use of bots. While CAPTCHAs can be used to combat bot programs in these online games, the incorporation of CAPTCHAs into games has often been seen in a negative light. This is certainly true in the case of directly assimilating the traditional text-based CAPTCHA approach into games. However, this paper presents an approach of integrating CAPTCHA challenges seamless into MMOGs in a way that does not completely destroy the sense of fun in the game. These mini-game CAPTCHA challenges introduce new gameplay features to the game in addition to providing security against the use of bots. Usability studies will be the topic of future work in order to evaluate this CAPTCHA approach from a user's perspective.

The authors are currently working on an animated CAPTCHA approach which uses 3D models in 3D scene. By using motion parallax, where objects at different distances in a 3D scene appear to move by different rates, users can distinguish the important content from the rest of the scene as the camera viewport moves around in the scene. This 3D CAPTCHA approach can easily blend in with the environment of a 3D game world.

REFERENCES

[1] B.S. Woodcock, "An Analysis of MMOG Subscription Growth," Presentation at ION Game Conference 2008. http://www.mmogchart.com/downloads/

[2] Activision Blizzard, World of Warcraft, http://www.worldofwarcraft.com/

[3] S. Mitterhofer, C. Kruegel, E. Kirda and C. Platzer, "Server-Side Bot Detection in Massively Multiplayer Online Games," IEEE Security and Privacy, vol. 7, no. 3, May/June 2009, pp. 29-36.

[4] http://www.exploitsrus.com/uo/bots.html

[5] S. Gianvecchio, Z. Wu, M. Xie and H. Wang, "Battle of Botcraft: Fighting Bots in Online Games with Human Observational Proofs," Proc. of ACM Computer and Communications Security Conference (CSS'09), Nov 2009, pp. 256-268.

[6] J. Heeks, "Understanding "Gold Farming" and Real-Money Trading as the Intersection of Real and Virtual Economies," Journal of Virtual World Research: Virtual Economies, Virtual Goods and Services Delivery in Virtual Worlds, vol. 2, no. 4, Feb. 2010.

[7] R.V. Yampolskiy and V. Govindaraju, "Embedded Noninteractive Continous Bot Detection," ACM Computers in Entertainment, vol. 5, no. 4, March 2008, Article 7.

[8] K.T. Chen, J.W. Jiang, P. Huang, H.H. Chu, C.L. Lei and W.C. Chen, "Identifying MMORPG Bots: A Traffic Analysis Approach," Proc. of ACM SIGCHI International Conference on Advances in Computer Entertainment Technology, 2006, article 4.

[9] R. Thawonmas, Y. Kashifuji and K.T. Chen, "Detection of MMORPG Bots Based on Behavior Analysis," Proc. of ACM Advances in Computer Entertainment Technology, 2008, pp. 91-94.

[10] L. von Ahn, M. Blum, N.J. Hopper and J. Langford, "CAPTCHA: Using Hard AI Problems for Security," Advances in Cryptology – Eurocrypt 2003, Lecture Notes in Computer Science, vol. 2656, pp. 294-311, 2003.

[11] K. Chellapilla, D. Larson, P. Simard and M. Czerwindski, "Designing Human Friendly Human Interaction Proofs (HIPs), Proc. of the SIGCHI Conference on Human Factors in Computing Systesm, 2005, pp. 711-720.

[12] https://www.google.com/accounts/UnlockCaptcha

[13] P. Golle and N. Ducheneaut, "Preventing Bots from Playing Online Games," ACM Computers in Entertainment, vol. 3, no. 3, July 2005, Article 3C.

[14] J. Yan, "Bot, Cyborg and Automated Turing Test (or "putting the Humanoid in the Protocol")", Security Protocols, Lecture Notes in Computer Science, vol. 5087, pp. 190-197, 2009.

[15] G. Hoglund and G. McGraw, Exploiting Online Games: Cheating Massively Distributed Systems. New Jersey: Addison-Wesley, 2007.

[16] M. van Kesteren, J. Langevoort and F. Grootjen, "A Step in the Right Direction: Botdetection in MMORPGs using Movement Analysis," Proc. of the 21st Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2009), 2009.

[17] T. Schluessler, S. Goglin and E. Johnson, "Is a Bot at the Controls? Detecting Input Data Attacks," Proc. of ACM SIGCOMM Workshop on Network and System Support for Games (NetGames '07), 2007, pp. 1-6.

[18] J. Yan and A.S.E. Ahmad, "Usability of CAPTCHAs – Or Usability Issues in CAPTCHA Design," Proc. of the Symposium on Usable Privacy and Security (SOUPS), 2008, pp. 44-52.

[19] N.J. Mitra, H.K. Chu, T.Y. Lee, L. Wolf, H. Yeshurun and D. Cohen-Or, "Emerging Images," ACM Transactions on Graphics, vol. 28, issue 5, December 2009, article 163.

[20] M. Shirali-Shahreza and S. Shirali-Shahreza, "A CAPTCHA System for Nintendo DS," Proc. of ACM SIGCOMM Workshop on Network and System Support for Games (NetGames '08), 2008, pp. 104-105.

[21] Bethesda Game Studios, Fallout 3, http://fallout.bethsoft.com

[22] Irrational Games, BioShock, http://www.2kgames.com/bioshock/

[23] BioWare, Mass Effect, http://masseffect.bioware.com/me1/

[24] E. Adams and A. Rollings, Game Design And Development: Fundamentals of Game Design. New Jersey: Pearson Prentice Hall, 2007.