Security Design in Online Games

Jeff Yan
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
Email: jyan@cse.cuhk.edu.hk

Abstract

The emergence of online games has fundamentally changed security requirements for computer games, which previously were largely concerned with copy protection. In this paper, we examine how new security requirements impact the design of online games by using online Bridge, a simple client-server game, as our case study. We argue that security is emerging as an inherent design issue for online games, after graphics and artificial intelligence, which have become important issues of the design of most games for decades. The most important new security concern in online game design is fairness enforcement, and most security mechanisms all contribute to a single objective, namely, making the play fair for each user.

1. Introduction

Online games are computer games played by one or more persons over the Internet. They have changed the way computer games are played. Nowadays, online games are one of the few profit-making e-commerce applications. For example, Sony's massive multiplayer role-playing game *EverQuest* [19] earns the company around $5 million per month from its monthly subscription fee alone, and this game's gross profit margin is around 40 percent [10]. Many other online games such as *Ultima Online* [16], *Diablo II* [4] and Korea's *Lineage* [15] have also achieved huge commercial success. Online gaming is developing into a multibillion-dollar business, and the market force behind it is that people love playing computer games with real human opponents, and they are willing to pay to play them.

For console and personal computer (PC) games, which have dominated the market, the security concern has been largely copy protection, i.e., how to make it hard to produce pirated copies. Online games, however, have fundamentally different security issues. Some vendors distribute their game client programs freely or with a symbolic fee, and they charge a user when he logs in to play on their servers. Thus, the traditional headache of copy protection can now be forgotten. Other problems have replaced it, such as password security, payment security and the availability of systems that host online games. But these are relatively well understood as they are shared by other networked e-commerce applications.

A new security concern in games is online cheating [17, 22]. Cheating was very popular in single player console or PC games. For example, it has been common for a player to make game missions easier by using cheating tools. For single-player games, cheaters were cheating the computer and nobody else would care. For online games, however, cheaters are now cheating human players sitting elsewhere on the network, and the cheated players do care. On the other hand, unlike single-player games, an online game is not (only) a discrete product, but (also) a service. This transforms the economics of the business, and player retention is critical for success [6, 20, 17]. It is commonly believed that online cheating ruins good games, and results in users, in particular new users, giving up. Therefore, online game operators should take online cheating seriously.

Cheating in online games was first discussed in [11]. In an excellent online article [18], Raymond discussed two cases of online cheating and their security implications. In [17], Pritchard reported more cases of real cheating occurred in various online games, and defined a framework for understanding them. However, his framework was ad hoc, and much online cheating couldn't fit into any of his categories. Later on, Yan et al. [22] discussed more security issues in online games, and started to build a taxonomy for online game cheating in a way structured to help security specialists understand the threats and look for countermeasures. Davis [6] categorized traditional casino cheating (e.g. Poker cheating, dice and slot machine cheating) and discussed their potential counterparts in an online environment. In addition, Baughman et al. [3] examined a type of look-ahead cheating in some real-time online games, where a dishonest player could hold his move until he knew all the

opponents' moves, and proposed lock-step protocols to mitigate such threat.

In this paper, we use a different approach to further understand cheating in online games. Instead of analyzing one cheat from this game and another from that, we systematically examine online cheating that has occurred or might occur in a single online game, and discuss how they impact the design of the game system. We do not consider cheating caused by issues such as password, payment and host security, which have been widely discussed in other contexts and are relatively better understood.

Online Bridge is chosen as our case study for the following reasons. First, both the design of an online Bridge system and the game itself are simple to explain and easily understandable, which helps focus on the important aspects of online cheating and system design while minimizing the disturbance from the complicated game details as in other online games. Second, it is a game largely based on a centralized client-server architecture, which is used by most commercial online games. Therefore, many lessons learned from online Bridge can be easily applied to many other games. Third, it hinges on collusion, a challenging security problem that imposes a severe threat in many scenarios but has unfortunately been a topic of little serious security research.

We first introduce the game of online Bridge in Section 2. Section 3 examines security failures that have occurred in online Bridge or might do, and discusses proper system designs that address some of these failures. Section 4 highlights collusion mitigation and its impact on the game design. In Section 5, we summarize that security has become an essential element of designing a decent online Bridge system. We also argue that security is emerging as an inherent issue for online game design, and the new security concern in online game design is about fairness enforcement, i.e., making the game play fair for each user. Section 6 concludes.

2. Online Bridge: Rule, Game Play and Architecture

In this section, we briefly introduce the game of contract Bridge, its rules and the common design of an online Bridge system.

2.1. The Game of Bridge

We first introduce contract Bridge for readers who are not familiar with this game. The introduction given here and the explanation elsewhere in footnotes where necessary should be enough to make this paper comprehensible even to people who know nothing about Bridge, since the ideas that we will discuss are more generally applicable.

Bridge is a four-person card game played with a deck of 52 cards, which has four suits, namely, spades ♠, hearts ♡, diamonds ♢ and clubs ♣, each containing the 13 cards A, K, Q, J, 10, 9, 8, 7, 6, 5, 4, 3, 2. Four players traditionally named as East, South, West and North play in two fixed partnerships: North and South are partners playing against West and East (i.e. N-S vs. W-E).

After the cards have been shuffled and dealt, the dealer starts and each player takes turns to **bid**. A bid is usually of n**M** form, where $1 \leq n \leq 7$ and **M** is a suit denomination (♠, ♡, ♢, ♣) or **NT**, and it can also be a *pass*, a *double*, or a *redouble*. Bidding stops once there are three *pass*es bid in succession, and the highest bid thereby is a *contract* for this round of play. A suit contract specifies the contracted suit as *trump suit*, whose cards are called *trump*s and rank above all cards of all other suits in this round; an NT contract means no trumps for this round. The partnership committing to a contract are *declarer* and *dummy*, and their opponents are *defenders*. The declarer partnership aims to make their contract, while the defenders will try to defeat it.

Card play starts with the first lead of the left hand opponent of declarer. Immediately after this *opening lead*, dummy exposes all his cards - he takes no further active part in this round of play, and the declarer will play cards of his own hand and those in dummy. Like bidding, card play also proceeds clockwise. Each player, if possible, must play a card of the suit led, but he is allowed to play any card when he cannot follow that suit. Each round of four cards is called a *trick*, which is won by the highest card (ranked by $\mathrm{A} > \mathrm{K} > \mathrm{Q} > \mathrm{J} > \mathrm{T} > \ldots > 2$) of the suit led, or the highest trump in it. The winner of a trick leads to the next.

To make a contract of n**M** and thus earn points accordingly, declarer must win $6 + n$ tricks. For example, he must win at least 9 tricks to make a contract of 3♡ by playing hearts as trumps, and at least 10 tricks for a contract of 4NT by playing with no trump suit. If the defenders prevent declarer from taking this many tricks, then they have successfully *set* him, and they earn points from each of his undertricks.

There are two types of competitive play of contract Bridge. The first scheme, also the original one, is Rubber Bridge, where a rubber ends when a partnership has won two games out of three. Sometimes luck plays an important role in Rubber Bridge, since some good hands are unbeatable. Duplicate is the second way to compete in Bridge, and it removes most of the luck in its Rubber counterpart. In Duplicate Bridge, a partnership is compared with other pair(s) playing the exact same hands under the same conditions. Final scores are calculated by comparing one pair's score to that of the other pair.

2.2. The Common Design of Online Bridge

A client-server architecture is typically adopted by an online Bridge system, no matter whether the system supports Rubber, Duplicate or both. The brain of the whole system is a server, which handles card shuffling and dealing, scoring and player ranking, and other coordinating activities. A user logs on to the server with his pre-assigned ID/password pair by running a client program. He can either start a new table with three other players and then play, or join an ongoing table as a kibitzer, who can observe the games played at this table while seeing cards held by one player.

The implementation of an online Bridge system is straightforward. It is similar to build any typical client-server application of a small or modest size, except that once a table becomes active (i.e., the play is about to start at the table), the interaction of people participating in the same table can be implemented as a peer-to-peer architecture.

3. Security Failures in Online Bridge

In this section, we discuss security failures that have occurred or might occur in online Bridge, most of which are due to online cheating and can be observed in many online Bridge communities. We also discuss countermeasures that address some of these failures.

We analyze security failures in online Bridge by summarizing them into two categories: 1) failures because of single-player cheating, and 2) failures due to collusion between multiple players. For the sake of completeness, not all security failures discussed in this section are new and significant.

3.1. Single-Player Cheating

A single player may cheat by either exploiting incompetent design or breaking the rules of fair play without doing anything technical.

3.1.1. Card Eavesdropping When a Bridge server deals cards to four players at a table, the card information is usually transmitted across the network in the form of plain text. A cheater may eavesdrop on cards held by his partner or opponents by intercepting the network traffic. Encryption is an obvious solution.

3.1.2. Client Hacking When cards are dealt to four players at a table, it is sufficient to transmit to each player only the 13 cards assigned to him. If the client program was designed to receive all four players' cards, then a cheater could manage to peek at cards held by his partner and opponents by hacking his own copy of the client. We learned that this design flaw had occurred to online Bridge before, but we have not yet identified any literature that could establish the evidence.

However, it is evident that a similar design did occur in another online game, which has some properties similar to Bridge. SIGUO (or *Four States* in English) is a popular oriental game which simulates war among four states. Four persons play the game on four displays, each coordinating an army, and players at the opposite side of the game board are allied. Like Bridge, SIGUO is also an *incomplete information* game: though four players place all cards on the game board, each is supposed to only see his own (chess and Go are *complete information* games, where each side can see all the pieces on the board). In an implementation of SIGUO, each client program maintained a copy of card information for all four players, though only the cards of one player were displayed to their owner, and the other three hidden. The consequence was that a cheat program displaying cards of four players was developed, and it became the nightmare of honest players. Although the designer tried many times to hide the sensitive data through methods such as code obfuscation and encryption, his effort was quickly defeated by crackers until he changed the design by keeping sensitive card information on the server side, and the client only updating the game board display as informed by the server [12, 23].

3.1.3. Exploiting Bad Randomness Lack of randomness will lead to either biased card distribution, or predictable consecutive dealswhich make the play much less fun and is unacceptable for online tournaments. A good pseudorandom number generator is needed for card-dealing in online Bridge.

3.1.4. Escaping When a player is going to lose, he disconnects himself from the system before finishing his ongoing game. This is *escaping cheating*. A cheater, usually called an "escaper", uses this cheat to avoid losing his rank points, since the ranking system usually does not score an unfinished game.

A common method for mitigating the escaping cheat is to track and display the number (or ratio) of unfinished games for each player.

3.2. Player Collusion

The interaction between players in online games can lead to new security problems. In online Bridge, the most visible security failure due to user interaction is collusion cheating, where two or more players are involved. We have observed three forms of collusion in online Bridge. One is exploiting a deadlock condition through collusion, the second is "friend in foes", and the third involves passing *unauthorized information* (i.e. information that a player is not enti-

tled to use, e.g. cards held by another player). While the first form appears to be a special case only, the others are general to all online Bridge systems.

3.2.1. Deadlock The fact that two players collude to deadlock the system was observed in a free online Bridge service provided by *Pogo* [7]. Pogo implemented a procedure, called "boot out", to expel a player who might be stuck due to the network lag on his side or had annoyed other players, and it worked as follows. Assume that N, S, W, E were playing at the same table. N thought that W should be expelled, and initiated the "boot out" procedure which sent N's request to both S and E. If both S and E agreed, then W would be automatically kicked out of the game; otherwise, W remained. Two cheaters playing as partners, say N and S, could misuse this feature as follows to stall the game when they were going to lose. One of the cheaters, say N, first launched a request to expel W. E refuses to do that because W behaved well and they're going to win. Meanwhile S ignores the "boot out" request and never casts his own vote. Realizing that they might be playing with cheaters, either W or E might like to expel the cheaters. But neither of them could do that because they were involved with one ongoing vote, and no further vote request could be accepted. Honest players W and E could do nothing but wait. None of the four players would quit the game because nobody would like to incur the penalty for an escaper. Their game therefore could not proceed; after a predefined amount of time, Pogo automatically destroyed the game (to save system resources).

Some dishonest players use this cheat when they are going to lose. Although it needs a partnership, they do not need to be familiar with each other. If they know this trick and have the same intention (i.e., avoid losing points), deadlock can be done without agreement beforehand or explicit information exchange.

This collusion exploits a design flaw; it can be prevented by fixing the "boot out" protocol.

3.2.2. Friend in Foes Collusion does not necessarily happen only between a partnership. A player may also collude with one of his opponents, and their collusion is not involved with illicit information passing either. For example, North could collude with East to beat West. No matter whether North is declarer or defender, East can intentionally lose to North his own winning tricks or West's. This form of collusion is commonly detectable to players at the same table.

3.2.3. Illicit Information Passing The stealthier collusion cheating in online Bridge is about illicitly passing information. Bridge is a game with incomplete information in both the bidding and card play stages. A player cannot tell any card or intention to his partner other than through the commonly known bidding or play *convention*s to convey information. Information exchanged in this legitimate way is usually imperfect, and it may be plausible or wrong. Therefore, most of the play is involved with uncertainty, reasoning and judgment. However, through illicitly informing card information, collusive cheaters can easily grasp a huge asymmetric advantage over honest players.

This form of collusion has been the most popular and devastating form of cheating in online Bridge, and it can occur in many different ways. A player can collude with different people, and he can pass or receive unauthorized information through different communication channels.

With whom to collude? Collusion of the form of illicit information passing can happen between a partnership, between a player and a kibitzer, and even between two players at different tables.

The most popular collusion form is that one colludes with his partner sitting at the opposite position of the table, and unauthorized information is passed between the partnership. A special case of this form is that a cheater logs on to a server with two distinct user IDs - he may own two IDs, or has got the second one from his family, friend or hacking - and then he partners these two IDs at the same table, both played by himself. Therefore, the cheater knows two hands of cards.

Since an online Bridge system usually allows a kibitzer to observe ongoing play at an open table, one playing at a table can also collude with one kibitzing at the same table. The kibitzer may tip off his colluding partner with cards held by anyone of three other players.

In addition, in Duplicate Bridge, the same deals will be played by different people. Therefore, collusion can also happen between two players who are at different tables.

For each of these collusion forms, cheaters do not necessarily always know all cards held by their partner or opponent, and they may only exchange "mission critical" information.

Collusion channels. When cheaters are physically together, they can collude using spoken or body language. Online, the communication channels that collusive players have exploited most is to pass unauthorized information by telephone, email, instant messenger or online chat.

There are also at least two covert channels[1] that collusive players can exploit in online Bridge. The first one is a timing covert channel. Denote by t_2 the time when a cheater follows his right hand opponent (RHO), and by t_1 the time when this RHO bids or plays. The cheater can always control the time interval $\Delta_t = t_2 - t_1$ and modulate Δ_t to encode unauthorized information, and his colluding partner can calculate each Δ_t and interpret it to proper informa-

1 A covert channel uses entities not normally viewed as a communication channel to transfer information.

tion by following a predefined protocol. Though this timing covert channel also exists in face-to-face Bridge, it may have a higher bandwidth in online Bridge, where a cheater can make use of precise timing with his computer clock.

A text-based chatting facility built in most online Bridge systems provides the second covert channel. One may encode covert information into texts, from which only his colluding partner can meaningfully decode the embedded information. For example, he may use a 4-word sentence starting with 'S' to mean "I have 4 spades", or type texts in a predefined way, e.g. by use of common typos, additional spaces or punctuation to transfer other hidden information, although his chosen texts and typing patterns may appear innocent.

It is a complicated issue to mitigate collusion of the form of illicit information passing in online Bridge, which will be discussed in detail in Section 4.

4. Collusion Mitigation and System Design

In face-to-face Bridge, collusion of the form of illicit information passing occurs largely between partners, and the unauthorized information is passed via spoken or body language. A committee of experienced experts is used to detect collusion after the fact by analyzing a complete record of cards, bids and tricks for each suspicious deal.

The timing covert channel also exists in face-to-face Bridge, and it is exploited in the form of player tempo or hesitation. There is no way to eliminate this covert channel in face-to-face Bridge, and instead a human (tournament director) is introduced to arbitrate whether collusion has occurred.

It is hard, however, to follow the practice in face-to-face Bridge to mitigate collusion in its online counterpart except for very few serious online tournaments, since human intervention (committee review in particular) is time-consuming and very expensive, and it cannot cope with thousands of games played online everyday in an effective and economic way.

In this section, we discuss countermeasures that can be taken to tackle collusion of the form of illicit information passing in online Bridge. There is no "silver bullet" for such collusion mitigation. We will discuss all possible options, including both preventive and detective means. For each countermeasure, both its advantages and disadvantages will be discussed. Since collusion mitigation is largely if not totally ignored, these countermeasures will introduce new components or other changes to the original system design.

4.1. Collusion Prevention

There are two types of preventive countermeasures: one tries to cut off the collusion channel, and the other to separate one from his colluding partner(s). For example, by disallowing kibitzing, it is technically easy to prevent collusion where a kibitzer is involved[2]. However, it appears that there is no way to prevent the "friend in foe" collusion. Meanwhile, for collusion in Duplicate Bridge where people at different tables are involved, by guaranteeing that the same deal be played at different tables at the same time, it provides a partial preventive solution only.

4.1.1. Overt Channels and Collaborative Monitoring One natural thought about collusion prevention is to cut off communication channels that are exploited. It is, however, very hard if not impossible for online Bridge designers to cut off communication channels provided by phone, email or the like.

In face-to-face Bridge, especially for serious tournaments, four players at a table are usually separated to two groups by an opaque screen, which is placed diagonally across the table, so that each player can see only one opponent but not his partner. This kind of *collaborative monitoring*, when each player has a video camera installed in his computer, can also be implemented online as a collusion-preventing means as follows. Once four players start a new table, the system will activate two video links, say, one between North and West (N-W) and another between South and East (S-E). Therefore, each player can monitor one opponent.

Nonetheless, many online players may not be willing to attach such an intrusive camera to their home computer, paying to sacrifice their own privacy. More important, it is hard to make this video monitoring system hack-proof at low cost.

4.1.2. Tackling Covert Channels It is easy to prevent collusion that exploits the timing covert channel to pass unauthorized information. To randomize the intervals of bidding and card play that each player can sense is a simple but effective solution. An insignificant amount of code will do the job.

Although it is technically easy to cut off the text covert channel by disabling the built-in chat facility, this solution does not make sense for online Bridge. A built-in chat facility is an inevitable element of any decent online Bridge service. First, players use this facility to inquire about and confirm conventions used for bidding and card play before the start of the game. Second, there are 200+ bidding and play conventions [9], and not all players are familiar with all of them. The official Bridge rules (e.g., both [1] and [2]) allow one to ask a full explanation of any bid or card play of his opponent, and require these questions to be answered

2 Kibitzing is a tradition in Bridge, and many online players enjoy the company of kibitzers. This may partly explain why many online Bridge providers do not enforce this simple technical measure.

honestly. This interaction is facilitated by this channel. Furthermore, each online Bridge service constitutes a virtual society, where people also socialize by chatting while playing.

It appears that the only sensible way of tackling the text covert channel is to minimize its bandwidth. There are various methods to do that. For example, text formatting can eliminate covert information encoded in additional spaces, punctuation, or predefined patterns such as combining upper and lower case letters in a certain way. Automatic spelling correction can sweep away information hidden in common typos. Moreover, when the meaning of a bid or card play is asked or explained, questions and answers often follows some similar patterns, because this kind of interaction concerns similar contents. For example, the explanation of a bid often includes the range of High Card Points (HCP)[3], suit distribution[4] and the strength of a specific suit. Therefore, such interactions between players may be done in a standardized way by using fixed verbal terms, or graphic user interface (GUI) components such as check boxes, so that versatile covert information that can be trickily hidden in human languages can be avoided in the first place. Though it is still likely to create a covert channel, e.g. by manipulating GUI-based interactions, it will be easier to spot such cheating behavior.

People may criticize that this customized chat will lead to an impersonal way of interaction, and thus it is user unfriendly or anti-social. Nonetheless, those covert information filtering mechanisms can be enabled only for people participating in active game sessions.

4.1.3. Randomized Partnering Randomization can be used to prevent collusion in some scenarios. For example, a scheme of randomized partnering is used in WarCraft III, a popular online game, to tackle "win trading", one form of collusion cheating. In previous WarCraft versions, two players might collude to climb up the rank ladder by win-trading as follows. Player A partnered with his friend B, and then each lost to the other alternately. The loss that A took would give B a victory point and raise B's ladder rank, and vice versa. Therefore, both A and B could climb to top positions in the ladder without playing a legitimate game[5]. In WarCraft III, the latest version, each player cannot choose at will his opponent, but instead an opponent of an equal or approximately equal rank will be randomly assigned to him by the system in each ladder game [21].

This randomized partnering scheme is useful, but it does not solve the problem of collusion such as win trading. For example, when only a few players log on to a game server, and all of them happen to be friends as well as cheaters, a randomized partnering does not make sense at all. This is not an unrealistic argument, especially when the popularity of cheating in online games is considered: 35% players admitted to online cheating in a survey [8].

Similarly, randomized partnering at most provides a partial solution in online Bridge. By coincidence, we were told by some veterans of a popular online Bridge community, who have been playing there since its initial release, that around one third of players in this community have cheated. In case the minority of people on a same server are honest players, the probability of cheaters being partnered together can be high.

Even though two randomly partnered players do not know each other, they still can collude in online Bridge if they want, since collusion will equally benefit each of them at any time, unlike in ladder games where much stronger mutual trust is required between two colluding players because each of them could easily slip away after grasping victory points for himself, let alone that the randomized partnering scheme cannot exclude the possibility that two randomly partnered players have established a good rapport on collusive play.

Furthermore, Bridge is a game requiring good cooperation between a partnership. Many people prefer to play with a familiar partner, instead of one assigned randomly, and some even prefer a fixed partner. This is especially common in tournaments. Although it is technically feasible to forbid familiar people to play as frequent partners, it is offensive and unacceptable to most Bridge players.

4.2. Collusion Detection

4.2.1. Rank Tracking Some collusion detection means are useful in other games, however, they do not work well in online Bridge. An example is rank tracking, which monitors the rank promotion of each player, and alerts abnormal patterns such as unusually quick rank promotion that may indicate collusion cheating. When used together with session logging, rank tracking can help detect the aforementioned win-trading collusion. For example, the quick ladder climbing of A and B can be easily alerted by rank tracking, then a fairly precise detection of their collusion can be achieved by analyzing session logs with a simple algorithm.

Rank tracking, however, is not a good method for alerting collusion in online Bridge. First, you cannot say that a partnership winning often (and thus climbing the rank list quickly) must be cheaters. A long-term partnership helps

3 The sum of A, K, Q and J each calculated with a predefined weight.

4 Number of cards in each suit. For example, one may have a 4432 suit distribution.

5 Yes, the ranking algorithm is also to blame for this win trading. If it was properly designed, say, to count one's loss points as well while calculating his rank, A and B couldn't climb up the ladder at the same time. A pair of cheaters then could not profit by collusion without harming one of themselves, unless they had hacked user IDs as victims - this would at least make it harder for them to collude.

establish rapport that is very useful for cooperative team games such as Bridge. It is not unusual that one wins more often when partnering with a familiar player than with others. Second, colluding players do not necessarily always win in Bridge, especially when they are novice players or cheaters, or when they play against very strong opponents. Additionally, a cheater whose rank promotion shows no suspicious pattern will stay safe. For example, rank tracking typically recognizes an abnormal pattern only after a cheater has played a sufficient number of hands. Thus, cheaters who have played hands less than the threshold value will not be detected.

4.2.2. An AI-based Detection Approach We have designed a new collusion detection scheme for online Bridge. Its core is a collusion detection robot which, armed with an inference engine and expert knowledge, automatically analyses the bidding and play of each partnership. The detection robot monitors three critical points in the process of a game, i.e., the contract bid, the penalty double bid[6] and the opening lead, that colluding cheaters have exploited most often, and registers as a suspicious signal each action (bid or card play) that is based on a decision too good to be drawn from incomplete information. That is to say, when information that one player can collect from his own hand and others' bids cannot logically lead to such a contract or penalty double bid, or such an opening lead as played, a suspicious signal is registered.

Such signals, however, are not exclusively a result of collusive play, and they contain noise arising from error, luck or even honest but ingenious play. While several methods have been proposed to force real cheating signals to stand out in this noisy environment, the detection robot registers signals to different suspicion levels. A cheater is identified only after enough signals of a *higher* suspicion level have been registered to him. For each player who is not labeled as a cheater, an *opportunity index* will be calculated by summing his registered signals with predefined weights. Thus, while achieving reasonable accuracy for cheater identification, our detection scheme enables a player to evaluate the risk of playing with any other people before making his own table choice. Moreover, this collusion detection scheme also works as an effective screening means that narrows serious attention down to only a few highly suspect players.

It appears that only this new scheme can provide a wider coverage of collusion mitigation. This, however, inevitably introduces a whole sophisticated subsystem and other major changes to the design of an online Bridge system. The design details of our collusion detection scheme are beyond the scope of this paper. Only some design issues introduced by this scheme are briefly discussed as follows.

First, the opportunity index of each player should be easily accessible.

Second, as the first step towards collusion detection, collection of raw data such as bids, tricks and player names should be properly addressed. The common practice in online Bridge is to discard such data after the game ends.

If our collusion detection robot is chosen to run offline, then raw data has to be recorded for each session. If it runs online, data can be collected in real time, and only sessions triggering suspicious signals are needed to be stored as evidence.

Clearly, it is cheaper for the client, in terms of usage of server CPU and network bandwidth, to submit bids or tricks to a data-collecting server in batches than one by one in real time.

One simple dual control method is recommended to guarantee the validation of all bids, tricks and session results: two copies are required for the same data, one from a player and another from his opponent; the system accepts the data only when these two copies match. As each deal is generated by the Bridge server, there is not much of a security worry about validation of cards assigned to each player if the server is secure.

Moreover, since the same bid or play may have a different meaning under a different convention, in order to identify collusion accurately, it is essential for the detection robot to analyze the bid or play with the very convention that a player has used. In Pogo, each player has a profile including conventions that he feels comfortable to play with. Each profile, however, can be modified by its owner at any time! Sometimes, players may use conventions that are not included in their profiles but temporarily agreed at the table. Therefore, it is necessary to record each bid or play together with the exact convention in real time. An easy way to do this is to annotate each *alerted* [7] bid or play with its corresponding convention. For example, a double bid can be recorded as *Dble/Lightner*, meaning a lead directing double that follows the *Lightner* convention. This requires adding additional fields to the data structure used for recording raw data.

4.3. User Interface Design

Proper design of the user interface is also relevant to collusion mitigation in online Bridge.

When a player makes a mistake, e.g., miscalculating his HCP points or clicking a wrong button, his bid or card play may appear to be the result of collusive play. On the other hand, a cheater can also use these as his excuses to defend his collusive play. A good user interface design for the client

6 A double bid intended as an attempt to obtain a penalty against the opponents' contract.

7 The Bridge rule requires that tournament players draw opponents' attention to bids and play following unusual agreements.

software helps reduce this kind of trouble. For example, big enough bidding buttons, big enough separating space between bidding buttons and between cards all help reduce these mistakes (or excuses). That the client calculates HCP points for a player and displays it in an eye-catching place is also useful.

5. Fairness Enforcement: Security Design for Online Games

What is the security concern for a *single-player* computer Bridge game? The answer is largely to make it hard to duplicate the binary program by copy protection techniques. For such a product, the design of its copy protection and the design of the game system can be independent.

We have seen in Section 4 that lots of security issues have to be considered in order to design a decent online Bridge system, and surprisingly, complicated security design is needed for such a simple game. We can summarize that with a strong impact on system design, security has (or should have) become an essential element of developing a decent online Bridge system, and it has (or should have) been tightly coupled with the game design itself. Meanwhile, it is clear that the main security concern in the design of such an online game is, by avoiding or minimizing online cheating, to enforce fair play between users.

These observations hold for the design of other online games. First, although online cheating may occur in different forms in each game, most of them can be organized into an analytical framework similar to the one used in Section 3. Theoretically, a commercial game service provider is also in a good position to orchestrate cheating, although all online cheating discussed in this paper is done by players. One example is gambling games provided by an online casino, where house cheating is likely to occur. For instance, a house might stealthily implement a payout rate which is much lower than claimed [6]. In such cases, regulation and third-party auditing might be necessary. However, most if not all online game operators treasure the trust that players have in their services. Believing that a cheating house will face the severe discipline of the market, we doubt that intentional house cheating would be a serious concern in most online games.

Second and more important, most online cheating in games can be addressed by the common security principles or mechanisms, including those that have been suggested to online Bridge.

Some issues that are particularly relevant to online game security and deserve further clarification are highlighted as follows.

Trust the Trustworthy Although it is (or should be) a straightforward issue, trust location is often handled in an imprudent way by online game designers.

To cheat by modifying game code or memory data has been a traditional method since the beginning of the PC game era, and many tools are available to help. This type of cheating also prevails in online games, and code or memory modification typically happens on the client side. Countermeasures of security by obscurity, including those discussed in [17], however, will eventually fail because they try to protect the wrong thing. The *information exposure* cheat, as defined in [17], as the client hacking case discussed in Section 3 shows, is really due to misplaced trust. Too much trust is placed on the client side, which cannot be trusted if the client owner is a cheater.

To avoid this, sensitive data has to be kept on the server side unless the client side is tamper resistant (which is very hard if not impossible in real life). It would be prudent for game developers to throw away security-by-obscurity, and instead, design their systems as if they are open source [18].

For some real-time games where network lag is critical and it may severely damage performance or even render the game play impossible when sensitive data is on the server side, a trusted proxy [13] sitting between the server and game clients may provide a reasonable solution. For example, the sensitive data is stored in the proxy, and it will be presented to the client only when necessary.

Collusion Mitigation Online cheats discussed in the literature, e.g. in [11, 18, 17, 3], were about single-player cheating, where a single player abused the system design or game rules. Unlike traditional single-player computer games, however, many online games allow multiple human players to interact in the game environment. As shown by online Bridge, this interaction can lead to collusion cheating that earns collusive players unfair advantage over honest ones.

On the other hand, collusion can also occur between a player and an insider (i.e., an employee of a game service provider). It typically involves internal misuse specific to the game.

Therefore, it is necessary for online game designers to consider all possible collusion scenarios and implement proper countermeasures in their game system.

Intrusion Detection The collusion detection robot discussed in Section 4.2.2 can be considered as one type of intrusion detection system (IDS). There are two special properties that differentiate it from common IDS systems. First, no concrete signature is extracted or stored, and instead, only abstracted models (determining whether "being logical or not") are used to detect collusive signals. Second, it is an application-dedicated IDS which provides protection to a specific application only, whereas common IDS systems, either network or host based, protect a wider system infrastructure but cannot detect misuses that have occurred inside each application. It is obvi-

ous that these two types of IDS systems can complement each other.

Just as with the collusion detection system designed for online Bridge, an application-dedicated IDS system can be an inherent part of security design for an online game where cheating prevails but preventive means cannot provide enough protection.

Reputation System In online Bridge, both the opportunity index and the number (or ratio) of unfinished games can contribute to determine the reputation of each user. This actually works as a reputation system that aids a player to evaluate the risk of playing with any other people, and thus regulate each other's behavior.

Such a reputation system can be implemented in many other online games, though parameters contributing to the determination of reputation may vary in each game.

To sum up, cheating in online games can be roughly summarized in a simple analytical framework as shown in Figure 1. The most important security consideration in online game design is to enforce fair play between users by avoiding or minimizing online cheating, and common security mechanisms such as randomness, encryption, security protocol (e.g. [3]), intrusion detection and reputation systems all contribute to this single objective.

This may appear to be a surprising observation, but it is a natural consequence of the intrinsic properties of such games. It is curious that in order to get to this point we had to wait around fifty years after the appearance of the first computer game[8] – in fact, until computer games were played between people in the same way as their counterparts in the non-electronic world.

It is clear that computer graphics and AI have been important elements in the design of most games for decades [5], but security is emerging as another inherent issue of computer game design.

6. Conclusion

This paper has examined the impact that security has on the design of online games by using online Bridge as a case study. For traditional single-player games, their security concern (mainly copy protection) could be independent of the game design. With the emergence of online games, however, security is tightly coupled with the game design, and it is emerging as another inherent design issue after computer graphics and AI.

8 According to [14], the first computer game was a military simulation game developed by Bob Chapman et. al. at the Rand Air Defense Lab in USA during 1952, and the first non-military game using a video display was probably a crude game of pool programmed at the University of Michigan in 1954.

- Single-player cheating
  - client hacking (misplaced trust)
  - information eavesdropping (lack of confidentiality)
  - escaping (rule violation)
  - ...
- Collusion (two or more persons involved)
  - collusion between players
  - collusion between a player and an insider
  - ...

Figure 1. Cheating in Online Games – An Analytical Framework

The most important new security consideration in online game design is fairness enforcement, i.e., making the game play fair for each user, and most security mechanisms contribute to this single objective. But the fairness enforcement perspective for online games is not just another straightforward security application; it may present challenging research problems which call for novel use of existing technology and the invention of new techniques.

Acknowledgements

The author thanks Ross Anderson and Will Ng for valuable discussions and comments. Comments from annoynimous reviewers also improved this paper.

References

[1] American Contract Bridge League, "Law 20. Review and Explanation of Calls", in *Laws of Duplicate Contract Bridge (American Edition)*, Effective May 27, 1997.

[2] American Contract Bridge League, Laws of Contract Bridge (Rubber Bridge Laws, American Edition), effective January 1, 1993.

[3] N Baughman and B Levine. "Cheat-proof Playout for Centralized and Distributed Online Games", in *Proc. of the Twentieth IEEE INFOCOM Conference*, Apr. 2001.

[4] Blizzard, Diablo homepage, `http://www.blizzard.com/worlds-diablo.shtml`.

[5] S Cass, "Mind games: to beat the competition, video games are getting smarter", IEEE Spectrum, December 2002.

[6] SB Davis, "Why Cheating Matters: Cheating, Game Security, and the Future of Global On-line Gaming Business", in *Proc. of Game Developer Conference 2001*, 2001.

[7] EA.com Inc, Pogo homepage, `http://www.pogo.com`.

[8] R Greenhill, "Diablo, and Online Multiplayer Game's Future", GamesDomain Review. At `http:`

//www.gamesdomain.com/gdreview/depart/jun97/diablo.html.

[9] A Kearse, Bridge Conventions Complete, London:A and C Black, 1977. ISBN: 0713617446.

[10] G Keighley, "The Sorcerer of Sony", in *Business 2.0*, August 2002. Available at http://www.business2.com/articles/mag/0,1640,42210,FF.html.

[11] A Kirmse and C Kirmse, "Security in Online Games", *Game Developer*, Vol. 4, no. 4, July 1997.

[12] Lianzhong, "Introduciton to SiGuo". Available at http://www.ourgame.com/srvcenter/game-intro/junqi.html.

[13] M Mauve, S Fischer and J Widmer, "A Generic Proxy System for Networked Computer Games", the First International Workshop on Network Games, 2002.

[14] Nature Publishing Group, Encyclopedia of Computer Science, the 4th edition, 2000.

[15] NCsoft, Lineage homepage, http://www.lineage.com.

[16] Origin, Ultima Online homepage, http://www.uo.com.

[17] M Pritchard, "How to Hurt the Hackers: The Scoop on Internet Cheating and How You Can Combat It", Information Security Bulletin, February 2001.

[18] Eric Raymond, "The Case of the Quake Cheats", unpublished manuscript, 1999. Available at http://www.catb.org/~esr/writings/quake-cheats.html.

[19] Verant, Everquest homepage, http://www.everquest.com.

[20] E Todd, "Quality or Death: How Business Model and Player Psychology Collide in For-Pay MMP Games", in *Proc. of Game Developer Conference 2002*, 2002.

[21] WarCraftIII.net, "Cheat Prevention: Steps Blizzard Will Take to Protect the WarCraft World", Aug 8, 2001. Available at http://www.warcraftiii.net/articles/cheat-prevention.shtml.

[22] J Yan et al, "Security Issues in Online Games", *The Electronic Library*, Vol. 20, No.2, 2002. A previous version appears in *Proc. of International Conference on Application and Development of Computer Games in the 21st Century*, Hong Kong, Nov. 2001.

[23] X. Yang, "Online Board Games", China Computer Weekly, Jan 15, 2001. Also available at http://media.ccidnet.com/media/ciw/990/b1201.htm.