School of Computing Science,
University of Newcastle upon Tyne

Security in Computer Games: from Pong to Online Poker

Jeff Yan and Brian Randell

Technical Report Series

CS-TR-889

February 2005

Published by the University of Newcastle upon Tyne,
School of Computing Science, Claremont Tower, Claremont Road,
Newcastle upon Tyne, NE1 7RU, UK.

Security in Computer Games: from Pong to Online Poker

Jeff Yan and Brian Randell

***Abstract*— Security can mean different things in a different application context. For decades, gaming has been a major computer application with its own distinct characteristics, and in fact, online gaming is now one of the most popular applications on the Internet. However, there are few systematic treatments of security concerns in gaming. In this paper, we briefly trace the history of computer games. Then, we examine the role that security has played in different games, from early mainframe-based games through arcade, PC and console games to the latest online games.**

Online cheating is widely considered a new security concern in computer games. However, it is not as well understood by security experts as one might expect. In this paper, we systematically investigate cheating in online games. We identify common forms of cheating as they have occurred or might occur in online games, and then we define a taxonomy of online game cheating with respect to the underlying cause (namely what is exploited?), consequence (what type of security failure can be achieved?) and the cheating principal (who can cheat?). One of our findings is that the four traditional aspects of security – confidentiality, integrity, availability and authenticity – are insufficient to explain cheating and its consequences in online games, and fairness can be a vital additional aspect.

I. Introduction

Computer games may initially appear to be merely source of entertainment for the younger generations in society. However, more than fun, they have been for decades one of the major computer applications, one with far reaching implications.

Few people might realise that computer games have been a driving force in the development of the IT industry, though they might understand that economies of scale can play a major role in the diffusion of a technology into a society. In the 1970s and early 1980s, the game business consumed so much silicon that it was a major funding resource for the semiconductor industry. The huge volume of memory chips and monitors consumed by early games also made it possible for manufacturers to produce them at much lower prices. All this contributed to make early personal computers considerably cheaper and thus more acceptable [3].

Nowadays, gaming still leads the development of consumer graphics hardware and software [27], and the computer game industry occupies a multi-billion dollar market. In 2001, the U.S. computer game market generated $9.4 billion in sales, outperforming for the first time U.S. film box office sales, which totalled $8.35 billion [30].

Computer games have also become a part of popular culture and have changed many people's life styles. People growing up in the 1960's have been often referred to as "the Beatles Generation" because of the huge influence that the Beatles and their music had upon them. Now many authors use "the Nintendo Generation" to refer to those who grew up while playing Nintendo games. On the other hand, just as the US has world-widely exported its culture through its movie and television programs, many oriental cultural concepts, with the rise of Nintendo and the Japanization of the game business that occurred in the late 1980s, have quickly entered into the world culture through games [3]. For example, the oriental legend of the Ninja Turtles is now common knowledge in many countries.

For decades, gaming has been a major computer application with its own distinct characteristics such as high quality graphics and strong emphases on playability. In fact, online gaming is now one of the most popular applications on the Internet [17]. It is well known that security can mean different things in a different application context. However, there are few systematic treatments of security concerns in such a representative application as gaming. In this paper, we examine the role that security has played in different computer games, from early mainframe-based games through arcade, PC and console games to the latest online games.

We first briefly trace the history of computer games in Section II. Section III is an overview of security concerns in various computer games. In Section IV, we systematically examine cheating in online games, which is widely considered to be a new security concern in computer games but not as well understood by security experts as one might expect. We identify common forms of cheating as they have occurred or might occur in online games. Then, we define a taxonomy of online game cheating with respect to the underlying cause (namely what is exploited?), consequence (what type of security failure can be achieved?) and the cheating principal (who can cheat?). We also present some results deduced from our taxonomy. Finally, Section V provides some summary conclusions.

II. Computer Games: A Brief History

Though there were earlier faked game-playing automata, the first ideas concerning how a game would truly be automated can be traced back to 1864, when Charles Babbage conceived how an automaton might play the game of chess. The common association of modern digital computers and games, in some sense, began in the 1940's when John von Neumann and Oskar Morgenstern studied the general theory of games and showed how their minimax algorithm theoretically applied to chess.

The earliest research into modern computer games was mainly concerned with computer intelligence and as a means for humans to understand their own intelligence. For the purpose of these areas of research, computer chess served as an excellent field of study. Many prominent scientists pioneered research into computer chess. For example, Claude Shannon published his fundamental paper [28] on computer chess in 1950, and Alan Turing proposed an approach to automating chess strategy [31] in 1953. Shannon's work proposed basic strategies for restricting the number of possibilities to be considered in a game, and outlined how computer chess would have to evaluate and choose future positions. Although he never wrote chess-playing software, almost all major chess programs that have ever been written were based on Shannon's ideas.

A. Early Mainframe Games

According to [19], the first computer game to materialise was a military simulation game developed by Bob Chapman et. al. at the Rand Air Defence Lab in USA during 1952, and the first non-military game to use a video display was probably a crude game of pool programmed at the University of Michigan in 1954.

In 1962, Steven Russell, Martin Graetz and Peter Samson wrote *Spacewar!* on a PDP-1 computer in MIT. Perhaps because PDP-1 was the world's first commercial *interactive* computer, which had a 15-inch CRT display and supported human-machine interaction in a way similar to current practice, *Spacewar!* is often cited as the first computer game in public media or popular readings.

No doubt most of the earliest computer games were ***mainframe games***, which ran on mainframe computers, and were mainly played by researchers or computer specialists.

B. Arcade Games

In 1971, a coin-operated game machine called *Computer Space* was set up in the corner of a bar in Silicon Valley. Unlike a computer, this game machine could do nothing but play the game, *Computer Space*, which, developed by Nolan Bushnell, actually was a simpler version of Steve Russell's *Spacewar!*. This was the first coin-operated ***arcade game*** with a video display[1]. In 1972, Bushnell produced another arcade game known as *Pong*, which, in distinction to *Computer Space*, was an instant success. *Computer Space* and *Pong* began the decades of arcade video games.

According to Nolan Bushnell [3], his *Computer Space* was not a computer game, but an electronic game based on a hard-wired state machine built from logical gates and counters. Similarly, most of the early arcade games were not computer games, but electronic games, since microprocessors were not invented until 1971. It is not known when the first computer arcade game was built, but it is believed that Nintendo started selling coin-operated video games using microcomputers from 1978 [23].

C. Console Games

Around the time Nolan Bushnell invented arcade games, which usually could not be played at home, the ***console game*** was invented so that people could play at home by attaching a game console to an ordinary television set. The first console of this kind was Odyssey, invented by Ralph Baer and marketed by Magnavox in 1971. The game console with replaceable cartridges was first introduced by Fairchild Instrument and Camera in late 1976, and it was named Channel F game system. This 8-bit system utilised an early microchip, the F8 invented by Robert Noyce, and its use of replaceable game cartridges offered consumers a theoretically inexhaustible variety of games. The game giant Nintendo made its fortune with its 8-bit NES console in the middle of 1980's. In 1995, Sony entered this console market by releasing its revolutionary Playstation console. Microsoft also entered this lucrative market in 2001 with the release of its Xbox console, a system born with huge amounts of investment. Now, the console market is dominated by Sony's Playstation 2, Nintendo's GameCube and Microsoft's Xbox.

D. PC Games

In some sense, personal computers were invented for fun by early hackers. Games were linked to the early personal computers in many ways. The first generation personal computer games were written for computers such as TRS-80 and Apple I in 1976. On the other hand, the arcade version of *Breakout*, a game machine designed by Steven Wozniak for Atari played a significant role in his creating Apple II which was first released in 1977 [33], [34].

Most early personal computer games were written for Apple II. But once IBM PCs and its compatibles dominated the personal computer market, they also became the dominating platform most games were targeted on. Consequently, people started to label games running on personal computers as ***PC games***.

Although Linux has been increasingly popular as a PC operating system, it has had little success wooing PC game players [15]. Instead, PC games have been largely designed for MS-DOS and MS-Windows platforms.

E. Online Games

Most computer games were for a single player only, namely the player played with or against a virtual player simulated by the game software. Multi-player games[2] designed for mainframe, arcade, console and PC platforms never became a major phenomenon until the emergence of *online games*, which have changed the way games are played. Online games are computer games played by one or more persons over the Internet, and the game device each player uses can be a computer, a game console or even a mobile phone powered by a microprocessor. Most online games allow many users, who may be in different places over the world, to play together over

[1] According to [16], the first commercial video game was *Galaxy Game*, another arcade version of *Spacewar!*. Developed by Bill Pitts and Hugh Tuck, it was installed in Stanford's student union a few months before Nolan Bushnell set up his *Computer Space*. Despite this disputed fact, Bushnell's *Computer Space* is still widely cited as the first such game in video game history.

[2] If a game is "multi-player", there are at least two alternatives: two or more people playing on the same machine in the same room, or two or more people playing over a network.

the network. Players can have more fun than before because they can compete with real human beings, and probably with different people each time. Moreover, each online game constitutes an exciting virtual community where people can socialise while playing games.

The first online game was probably the MUD1 (Multi-User Dungeon) developed by Roy Trubshaw and Richard Bartle at Essex University in England in 1979. MUD1 was the first adventure game to support multiple players, and it was so influential that it has inspired lots of variations (a comprehensive overview of major MUDs until 1990 can be found in [1]).

Nowadays, online gaming has become one of the relatively few profit-making e-commerce applications. Indeed it is developing into a multi-billion dollar business in its own right. The market force behind it is that players love playing games online with real human beings, and they are willing to pay to play. Sony's online game *EverQuest* [32] earns the company around $5 million per month from its monthly subscription fee, and the game's gross-profit margin is around 40 percent [10]. Many other online games such as *Ultima Online* [24], *Diablo II* [2] and Korea's *Lineage* [20] have also achieved huge commercial success.

III. Security Concerns in Computer Games: An Overview

In this section, we examine the role that security has played in different games, from early mainframe-based games through arcade, console and PC games to the latest console games.

A. Security in Mainframe, Arcade and PC Games

Security for computer games was simply ignored in the early years. Although security started to become a concern for computer scientists in the 1960's, it was only one of the many issues for operating system designers. There were no real security considerations that were specific to computer games, which were mainly mainframe games at the time.

The only plausible exception known was a unique access control feature implemented in the Cambridge Multiple Access System at Cambridge University in the middle 1960s. It controlled access to data by the identity of the program that was being used to do the access, as well as or instead of a user identity. Examples utilising this unusual access control feature included "the league table file that recorded people's relative standing in a competitive guessing game, which was accessible to the game program only" [21].

The security of coin-operated arcade games dealt mainly with the physical security of coin boxes locked in the arcade game machines. People tackled this security issue simply by using safer boxes, and putting game machines in trustworthy or guarded places.

As with the other content-critical industries such as music or video, when issues of content piracy expanded to threatening levels, copy protection became important, especially for PC games. From the beginning, PC games were especially vulnerable to piracy, because they have been typically shipped on removable storage media such as floppy disks and CD-ROMs. Many diskette or CD-ROM based copy protection techniques have been developed for games and other software, and they achieved only a mixed success because this is a co-evolutionary war between guardians and crackers.

B. Security in Console Games

There were also copy protection mechanisms in console games. However, the bulk of the security effort of console game vendors has been directed towards locking in customers to their hardware, and making strategic plays in the market. Nintendo appears to be the first console vendor who adopted security techniques for this purpose, and introduced practices that were to define the game console industry to this day.

The first game console released by Nintendo was Famicom (short for "Family Computer"), and it was an immediate hit in Japan in 1983. Famicom was purposefully designed as a closed system. Security techniques were introduced to an improved version of Famicom before it was released in USA under the name of NES (short for "Nintendo Entertainment System") in 1985.

The security system that engineers in Nintendo introduced to NES was a complex implementation of a simple lock-and-key concept. Each authentic NES console and cartridge contained a security chip, and these chips communicated with each other with a shared secret key. As long as the chip in the console and another in the cartridge were communicating, the system operated; otherwise, the system froze up. In this way, a NES console would reject any non-Nintendo game cartridges; if a Nintendo cartridge was inserted into a console that didn't know the key, the game would not run either.

While this lock-and-key system worked to reject counterfeits made for Famicom and stop NES counterfeiters, it stopped more than that.

Nintendo protected its security system via both copyright and patent. The company filed a patent for its lock-and-key system[3]. Meanwhile, Nintendo's code used for authentication, known as "10NES", was also registered at the US Copyright Office. Only Nintendo had access to this intellectual property[4]. And Nintendo periodically modified the keys inside NES consoles so that only Nintendo-approved games could play. Therefore, no one could manufacture their own games for the NES platform without Nintendo's approval.

Nintendo used this advantage to enforce strict licensee agreements. For example, licensees had to pay expensive

[3] US patent No. 4,799,635, "System for determining authenticity of an external memory used in an information processing apparatus", filed in 1985 and granted in 1989.

[4] There is only one legitimate way to get access to a copyrighted code: with an affidavit filed to indicate that a work was the subject of litigation, a copy of the concerned work could be legally obtained from the US copyright office. The rationale behind this was: a party being sued for violating a copyright couldn't defend itself unless it could review the copyrighted material [29]. This approach can be exploited as a "litigation attack". An infamous unsuccessful case was attempted by Atari Games, one of the major competitors of Nintendo at the time [29]. In early 1988, Atari Games failed to reverse engineer Nintendo's security system, but they managed to get a copy of 10NES from the Copyright Office by falsely alleging that it was a present defendant in a case with Nintendo. Atari Games was successfully sued by Nintendo for intellectual property infringement.

royalties. Nintendo also used this advantage to censor games. If a game developed by a licensee received poor comments from Nintendo, no matter how much the licensee might have spent in its development, it would not be released on any NES consoles. Nintendo's security chip was referred to in the industry as a "lock-out" chip. To further extend Nintendo's control, all cartridges for the NES console had to be manufactured through Nintendo, though it was the licensee's job to sell them.

All this combined to make Nintendo the number one video game manufacturer at the time, as well as one of the most successful companies in the world. Nintendo at its peak grabbed a near-monopoly position in the industry, which it retained for nearly ten years [29].

Following Nintendo's practice, Sega, a former major player in the console game market, also created security systems in its consoles to guard against software pirates and unlicensed publishers. While manufacturing its Genesis III console in 1990, Sega began to include a Trademark Security System (TMSS), a lock-out device that operated by searching each game cartridge inserted into the console for four bytes of data present at a particular location in all Sega-produced game programs. If the console failed to find the TMSS initialization code at the necessary location, it would not allow the game to operate.

Until now, it appears that all major console vendors still use security techniques to make strategic plays in the market, retaining the tradition initiated by Nintendo. Microsoft, the newcomer, has also employed security techniques in its Xbox to reject games developed for other platforms. Though some of its strategic plays enforced by security techniques are as yet still unclear, the MIT Xbox hack [9] may have revealed some hints.

C. Security in Online Games

The emergence of online games has not only changed the way games are played, it has also led to fundamentally changed security requirements for computer games.

Online game vendors distribute their game software free of charge or with a symbolic fee, and charge a user when he logs in to play on their servers. Thus the traditional headache of copy protection can be forgotten[5].

As (distributed) Internet applications, online games concern more complicated security issues than traditional computer games did. For example, security for online games includes security for game hosting systems in a networked environment, and it covers issues such as privacy and access control. For commercial online games, security also concerns the payment mechanism. But these are relatively well understood issues as they are shared by other Internet applications.

Recent research has suggested that cheating is in fact a major new security concern for online computer games [26], [35], [36]. A careful investigation of online cheating can benefit the study of security in this representative Internet application. However, cheating has not been studied as thoroughly as one might expect. For instance, although cheating is rampant in online games, there is no generally accepted definition for it.

Three reasons may explain this fact. First of all, cheating is a relatively new topic for security researchers, although many online game players have been familiar with it for a considerable time. Second, the variety of online games now in existence has made cheating a complicated phenomenon. For example, there are a number of entirely different game genres, and each may give rise to varied forms of cheating. Third, many novel cheats have been invented that are different from but often entangled with ordinary security attacks.

IV. Cheating in Online Games

In this section, we systematically examine cheating in online games while adopting the following definition for it, which refines the version used in [35].

> *Any behaviour that a player uses to gain an advantage or achieve a target in an online game is cheating if, according to the game rules or at the discretion of the game operator (i.e. the game service provider, who is not necessarily the developer of the game), the advantage is unfair to his peer players or the target is one that he is not supposed to have achieved.*[6]

Specifically, we present a classification scheme for online game cheating, in the expectation that by categorizing various online game cheats, our understanding of this phenomenon will be increased, useful patterns and conclusions can be established, and it will be possible to provide improved protection for online game systems against cheating using this knowledge. Our classification is intentionally reminiscent of the dependability taxonomy provided in [13].

A. Related Work

Several authors have attempted to define a framework for classifying and understanding online game cheating. For example, Davis [8] categorized traditional forms of casino cheating and discussed their potential counterparts in online games. However, a casino is not representative enough to reflect all forms of online game settings, in which cheating may occur with differing characteristics.

Pritchard [26] reported many real cases of online cheating that have occurred in various games, and classified them into a framework of six categories. However, his classification is ad hoc and not comprehensive. Indeed, a lot of online cheats do not readily fit into any of his categories.

Yan et al [35] reported a more thorough effort identifying eleven common cheating forms in online games. In addition, Yan [36] examined the cheating techniques that have occurred or might occur in online contract bridge communities, and organized them into a simple framework.

There is also a large amount of literature investigating the definition of taxonomies for security vulnerabilities, attacks or

[5] This business model appears to provide a good solution to the long-standing problem of software piracy.

[6] At present the preponderance of cheating in online games is carried out by male game players, so for linguistic convenience in the rest of this paper we will appear to imply that all cheaters are male.

intrusions in a general setting. For example, Landwehr et al constructed a classification of security flaws in software with respect to genesis (how did the flaw enter the system?), time of introduction (when did it enter the system?) and location (where in the system is it manifested?) [12]. Krsul conducted his PhD research on software vulnerability analysis and taxonomy construction [11]. Neumann et al gave a taxonomy of attacks with respect to the technique used to launch each given attack [22]. The MAFTIA project [6] proposed a taxonomy for intrusion detection systems and attacks. Lindqvist and Jonsson [14] conducted a brief but useful survey on the desired properties of a taxonomy, and defined a taxonomy of intrusions with respect to intrusion techniques and results. All these studies are relevant to our aims. In online games, a player might cheat by exploiting a "vulnerability", or by launching an "attack" or "intrusion". However, as will be discussed later, online game cheating also has some unique manifestations.

B. Common Cheating Forms

Before defining our taxonomy, we identify all cheating forms known to us, as they have occurred or might occur in online games.

Eleven common cheating forms were identified in [35]. However, we have seen the need of refining the framework given in [35] and now present a revised listing, which classifies cheats into 15 categories. Those that are new, or are significantly revised versions of the categories listed in [35], are marked with asterisks.

A:* **Cheating due to Misplaced Trust.** Much cheating involves modifying game code, data, or both on the client side. A cheater can modify his game client program, configuration data, or both, and then replace the old copy with the revised one for future use. Alternatively, the modification or replacement of code and data can be done on the fly.

This form of cheating is really due to misplaced trust. Too much trust is placed on the client side, which in reality cannot be trusted at all because a cheating player can have the total control over his game client. Countermeasures based on security by obscurity approaches such as program obfuscation will eventually fail in the fight against this form of cheating, because they try to protect the wrong thing.

B: **Cheating by Collusion.** Players collude to gain unfair advantages. Representative cases, including various collusive cheats in online contract bridge communities and the "win trading" collusion in the WarCraft game, were discussed in detail in [36].

C:* **Cheating by Abusing Game Procedure.** This form of cheating may be carried out without any technical sophistication, and a cheater simply abuses the operating procedure of a game. One common case is *escaping*: a cheater disconnects himself from the game system when he is going to lose [35], [36].

Another example is *scoring cheating* [35] in online Go games, which abuses the scoring procedure as follows. When a game is finished, "dead" stones must be identified and then removed by hand before the system can determine which side wins this game. During this scoring process, however, a cheating player may stealthily remove "alive" stones of his opponent, and then "overturn" the game result. (When the size of territory occupied by each side is close, this cheating may easily escape the awareness of the cheated player, especially when he is not a strong player.)

D: **Cheating Related to Virtual Assets** Virtual characters and items acquired in online games can be traded for real money. Lots of cheating related to these virtual assets can then occur.

E:* **Cheating due to Machine Intelligence.** Artificial intelligence techniques can also be exploited by a cheating player in some online games. For example, the advancement of computer chess research has produced many programs that can compete with human players at the master level. When playing chess online, a cheater can always look for the best candidates for his next move by running a strong computer chess program.

This is in fact cheating due to the superiority, in this particular situation, of machine intelligence over that of an ordinary human being. It can happen in many other online games, depending on two factors: 1) the properties of the game: whether the game can be modeled as a computable problem, and 2) the maturity of AI research into such games. For example, online Go players do not worry about this form of cheating, since the state of the art of AI research can produce only very weak computer Go programs – the strongest one at present can be easily beaten by an amateur human player [18].

F:* **Cheating via the Graphics Driver.** By modifying the graphics driver installed in his operating system, a cheating player can, for example, make a wall transparent in some online games so that he can see through the wall and locate other players who are supposed to be hidden behind the wall [4].

G: **Cheating by Denying Service to Peer Players.** A cheating player gains advantages by denying service to his peer players. For example, a cheater can delay the responses from one opponent in a real-time game by flooding his network connection. Other peer players will then be cheated into believing that there is something wrong with the network connection of the victim, and agree to kick him out from the game in order to avoid the game session being stalled.

H:* **Timing Cheating.** In some real-time online games, a cheating player can delay his own move until he knows all the opponents' moves, and thus gain a huge advantage [7]. This *look-ahead cheat* is one kind of *timing cheating*.

Other timing cheating includes the *suppress-correct cheat*, which allows a cheater to gain an advantage by purposefully dropping update messages at the "right" time [7].

I: **Cheating by Compromising Passwords.** A password is often the key to much of or all the data and authorization that a player has in an online game system. By compromising a password, a cheater can have access to the data and authorization that the victim has in the game system.

J: **Cheating due to Lack of Secrecy.** When communication packets are exchanged in plain text format, one can cheat by eavesdropping on packets and inserting, deleting or modifying game events or commands transmitted over the network.

K: **Cheating due to Lack of Authentication.** If there is no proper mechanism for authenticating a game server to clients, a cheater can collect many ID-password pairs of legitimate players by setting up a bogus game server. Similarly, if there is not a proper mechanism authenticating a client, a cheater can also exploit this to gain advantages. For example, it is critical to re-authenticate a player before any password change is executed for him. Otherwise, when a player leaves his computer temporarily unattended and his game session unclosed – in countries such as China and Korea, many people play online games in internet cafes – a cheater who can physically access the player's machine may stealthily change his password, and exploit the changed password afterwards.

L:* **Cheating by Exploiting a Bug or Loophole.** This form of cheating exploits a bug or loophole in game programs or the game design itself, without involving any modification of game code or data. Once discovered, such a bug/loophole will give knowledgeable players a major advantage. An early case of such cheating can be traced back to an incident, that occurred in Lucasfilm's Habitat, one of the first multi-user virtual environments. Due to an inadvertent pricing error, people in the game could sell virtual items to a pawn shop at a higher price than they paid to get them from a vending machine. By shuttling back and forth between the vending machine and the pawn shop, some players become millionaires overnight [5].

If a player has to modify the game program or data in order to exploit a bug or design loophole to gain unfair advantages, according to our definition, his cheating behaviour will not be covered by this form, but by *cheating due to misplaced trust* or the following form of *cheating by compromising game servers*.

M:* **Cheating by Compromising Game Servers.** A cheater can tamper with game server programs or change their configuration once he has obtained access to the game host systems.

N: **Cheating Related to Internal Misuse.** A game operator usually has the privileges of a system administrator. It is easy for an insider – an employee of the game operator – to abuse this privilege. For example, he can generate super characters by modifying the game database on the server side.

O: **Cheating by Social Engineering.** Often cheaters attempt to trick a player into believing something attractive or annoying has happened to him and that as a result his ID and password are needed.

1) Nature of Cheats: Atomic vs. Non-Atomic: The list given above attempts to be comprehensive but not necessarily disjoint. Therefore, a given cheat might fall into more than one category. It would be ideal to define a list of common cheating forms that is disjoint, but unfortunately this has proved to be a very challenging task.

Although each listed form can be an independent cheat, an actual case of cheating may be complex and involves multiple forms of cheating. For example, the Pogo cheat discussed in [36] involved two dishonest players who collusively abused a voting protocol to gain advantages. It is in fact a cheat due to collusion, which abuses the game procedure, and at the same time also exploits a loophole in the game system design.

Another example is the *hit-then-run* cheat [35] in Internet Go games, which can occur as follows.

Go is a time-critical game played between two people. The Go server counts the time spent by each player in a game, and the player who runs out of time will automatically lose the game. Many online players choose to play 25 moves in 10 minutes or less, and it is usual for one to play 5 stones in the last 10 seconds. Therefore, a cheating player can easily defeat an opponent by timing him out with a well timed flooding attack. This is a form of cheating by denying service to peer players.

The above *timeout* cheat can be used together with cheating by abusing the game procedure. Some Internet Go services implemented a penalty rule to fight against the *escaping* cheat: players who disconnect themselves will lose their unfinished game unless they return to finish it within a limited period. A *hit-then-run* cheater can take advantage of this rule in the following way. He floods one opponent so that the game is recorded as disconnected by the opponent. Then he does not log on until the penalty period has passed. The game cannot be finished in time, and the opponent will automatically lose points for it.

2) Generic vs. Specific Cheats: Table I classifies all the above fifteen cheating forms into two divisions. The "generic" division includes seven forms of common cheating in online games, which are also generic to all network applications but may appear with different names such as "attacks" or "intrusions" in different contexts. The "specific" division includes both cheating specific to online games, and cheating that may also occur with different names in other network applications but has some interesting features or implications in the context of online games.

In fact, some cheating forms even appear to be unique to specific game genres. For example, *cheating due to machine intelligence* would seem to be unique to online versions of the traditional board or card games, and *cheating related to virtual assets* has occurred only in multiplayer role-playing games. This can be explained by the unique characteristics of such game genres.

C. A Taxonomy of Online Cheating

In this section, we define a taxonomy for online game cheating. This is a three dimensional taxonomy, and online cheating is classified by the underlying cause (what is exploited?), the

Type	*Label*	*Cheating Form*
Specific to online games	A	Cheating due to Misplaced Trust
	B	Cheating by Collusion
	C	Cheating by Abusing Game Procedure
	D	Cheating Related to Virtual Assets
	E	Cheating due to Machine Intelligence
	F	Cheating via the Graphics Driver
	G	Cheating by Denying Service to Peer Players
	H	Timing Cheating
Generic	I	Cheating by Compromising Passwords
	J	Cheating due to Lack of Secrecy
	K	Cheating due to Lack of Authentication
	L	Cheating by Exploiting a Bug or Design Loophole
	M	Cheating by Compromising Game Servers
	N	Cheating Related to Internal Misuse
	O	Cheating by Social Engineering

TABLE I

COMMON CHEATING FORMS IN ONLINE GAMES

System Design Inadequacy	In the Game System	Cheating due to Misplaced Trust
		Cheating due to Lack of Secrecy
		Cheating due to Lack of Authentication
		Timing Cheating
		Cheating by Exploiting a Bug or Design Loophole
		Cheating by Denying Service to Peer Players
	In the Underlying Systems	Cheating via the Graphics Driver
		Cheating by Compromising Game Servers
		Cheating by Denying Service to Peer Players
Operational Failure	Cheating by Collusion	
	Cheating Related to Internal Misuse	
	Cheating by Abusing Game Procedure	
	Cheating by Compromising Passwords	
	Cheating by Social Engineering	
	Cheating due to Machine Intelligence	
	Cheating Related to Virtual Assets	

TABLE II

ONLINE GAME CHEATING TAXONOMY: BY CAUSE

cheating consequence (what type of security failure can be caused?) and the cheating principal (who can cheat?).

Tables II – IV shows the details of the taxonomy by cause, consequence and cheating principal, respectively. Note that the same cheating form will appear at least once in each of these categories. Divisions and, where appropriate, subdivisions are provided within the categories; these and their motivations are described in detail later.

1) By Cause: Online cheating may or may not exploit system design inadequacies. For example, *cheating by exploiting a bug or loophole* exploits inadequacies in the game design, implementation or both. However, social engineering does not involve exploitation of any technical design inadequacies. Therefore, we classify the causes of online cheating to two divisions: *system design inadequacy* which concerns a technical design failure arising in the process of system development, and *operational failure*, which is largely due to a failure of human-computer interaction during the operational phase of a game system. (Some operational failures can be ultimately a design failure: they arise due to "the inability to foresee all the situations of the system will be faced with during its operational life, or the refusal to consider some of them" [13] for reasons such as a concern for time-to-market.)

There are two subdivisions in system design inadequacy: *inadequacy in the game system* and *inadequacy in the underlying systems*. Online games are applications running on top of an underlying networking and operating system. A cheater can exploit a flaw in a game system, a flaw in its underlying networking or operating system, or both.

Cheating due to misplaced trust, lack of secrecy or authentication, timing cheating, cheating by exploiting a bug or design loophole exploit technical inadequacies in the game system, and they belong to the first subdivision.

Two common cheating forms, namely *cheating via the graphics driver* and *cheating by compromising game servers*, belong to the second subdivision. Specifically, the first cheating form occurs on the game client side. However, rather than exploit the game system itself, it modifies a system driver that is part of the operating system. Similarly, a cheater compromising a game server usually breaks into the server by exploiting an operating system or network flaw on the server side[7].

In addition, *cheating by denying service to peer players* usually exploits some inherent weakness of the network layer, but

[7] A game server program may have flaws that can be remotely exploited by a cheater, but we have not yet seen such cases in real life.

it can also be committed by exploiting a design inadequacy in the game system alone. For example, a cheat that occurred in the Firestorm game [13] exploited a buffer-overflow condition in the game program to disconnect all players. Therefore, this form of cheating is included in both subdivisions.

A lot of cheating techniques in online games, such as collusion, social engineering, game procedural abuse, password compromising, cheating related to internal misuse or virtual assets, are only weakly related to any technical design inadequacy. Instead, they largely exploit "the human side" of computer security [25]. Therefore, they are classified as operational failures.

2) By Consequence: We largely base our classification of cheating consequences on the four traditional aspects of computer security: confidentiality (prevention of unauthorized disclosure of information), integrity (prevention of unauthorized modification of information), availability (prevention of unauthorized withholding of information) and authenticity (the ability to assure the identity of a remote user regardless of the user's host). A breach of confidentiality results in *theft of information or possessions*, a breach of integrity results in *code or data modification*, a breach of availability results in *service denial* and a breach of authenticity results in a *masquerade*.

Cheating by collusion, compromising passwords or *social engineering*, or *cheating due to lack of secrecy* result in the theft of information or possessions in a game. *Cheating due to lack of authentication* results in a *masquerade*. *Cheating by denying service to peer players* involves selective service denials, but *cheating by compromising game servers, due to misplaced trust, or related to internal misuse* usually involve integrity failure.

However, these traditional aspects of computer security are insufficient to cover all the consequences of online game cheating. For example, the cheat exploiting the erroneous pricing in Habitat violated none of the issues of confidentiality, availability, integrity or authenticity. And the list goes on.

We introduce "fairness" between peer players as an additional aspect for understanding online game cheating, and a breach of fairness results in a *fairness violation*. Either *cheating by abusing game procedure, timing cheating, cheating by exploiting a bug or design loophole*, or *cheating due to machine intelligence* can result in a *fairness violation*. Although *cheating related to virtual assets* may result in theft of possessions, it is hardly the result of confidentiality failure. Therefore, *cheating related to virtual assets* is also categorized as a *fairness violation*.

3) By Cheating Principal: A player can cheat independently either in single player or multi-player online games, whereas in multi-player games two or more players can cheat via malicious cooperation. Furthermore, a player can also collude with an insider to cheat. The identity of the cheating principal is used as the third dimension in our classifications, and it provides a way of distinguishing cooperative cheats from their independent counterparts.

Regarding the cheating principal, there are three divisions: by *player*, by *game operator* and by *operator-player* (i.e. the cooperation of player and game operator).

The by *operator-player* division accommodates cheating committed through the cooperation of a player and an insider, which typically involves collusion as well as internal misuse that are specific to the game.

The by *game operator* division accommodates cheating related to internal misuse, where no collusion between player and insider is involved, however. One example is that of an insider who is also a player. As discussed in [36], house cheating orchestrated by a game operator alone is likely to occur. However, it is beyond the scope of our online cheating definition used in this paper.

There are two subdivisions in the cheating by player category, namely by *single player* or by *multiple players*. Collusion between players is covered by the second subdivision, whereas, as indicated in Table IV, 13 other cheating forms belong to the first subdivision.

D. Discussion

Our taxonomy brings out a systematic view of online cheating, from which a number of observations can be made.

First, it is interesting to examine the distribution of each common cheating form in the two orthogonal dimensions of causes and consequences.

Table V constructs such a distribution matrix, where the cheating cause and consequence are displayed in rows and columns respectively, and cheating forms in the cells are represented with their labels assigned in Section 3. The matrix in Table V shows that most types of online game cheats have been about information theft, code or data modification, or fairness violation, and they largely exploit either operational failures or flaws in the game systems.

However, the distribution of cheating forms in the cause-consequence matrix may not remain stationary as online games and the cheating phenomenon co-evolve. Therefore, any observation based exclusively on this matrix may have to remain tentative. For example, it is not yet clear whether cheats exploiting flaws in the underlying networking and operating systems will increase in the future.

It is also interesting to note that as a result of taxonomic analysis using Table V, we have corrected a mistake in a previous version of this paper. Namely, we found that we carelessly missed a type of cheating by denying service to peer players, which involves exploitation of design inadequacies in the game system only.

It appears that we can also use this table to suggest novel additional forms of cheating that will likely occur in the future while arguing why some blank squares in the table are and whether they will remain empty. For example, it appears that cheating leading to service denial due to operational failure will never occur, since seemingly there is no other way to deny service to peer players other than by exploiting technical design inadequacies in the game system, the underlying systems, or both. However, it is very likely that cheats which lead to masquerade, information theft or fairness violation and which are due to design inadequacies in the underlying systems, will occur in the future, although it is not yet clear in which forms they will manifest themselves.

Second, as the classification by cheating principal in Table IV shows, the majority of current game cheating can be

Theft of Information or Possessions	Cheating by Collusion
	Cheating by Compromising Passwords
	Cheating due to Lack of Secrecy
	Cheating by Social Engineering
Service Denial	Cheating by Denying Service to Peer Players
Code or Data Modification	Cheating due to Misplaced Trust
	Cheating via the Graphics Driver
	Cheating by Compromising Game Servers
	Cheating Related to Internal Misuse
Masquerade	Cheating due to Lack of Authentication
Fairness Violation	Cheating by Abusing Game Procedure
	Timing Cheating
	Cheating by Exploiting a Bug or Design Loophole
	Cheating Related to Virtual Assets
	Cheating due to Machine Intelligence

TABLE III

ONLINE GAME CHEATING TAXONOMY: BY CONSEQUENCE

Player	Single Player	Cheating due to Misplaced Trust
		Cheating by Abusing Game Procedure
		Cheating Related to Virtual Assets
		Cheating by Compromising Passwords
		Cheating by Denying Service to Peer Players
		Cheating due to Lack of Secrecy
		Cheating due to Lack of Authentication
		Timing Cheating
		Cheating by Exploiting a Bug or Design Loophole
		Cheating by Compromising Game Servers
		Cheating by Social Engineering
		Cheating due to Machine Intelligence
		Cheating via the Graphics Driver
	Multiple Players	Cheating by Collusion
Game Operator	Cheating Related to Internal Misuse (No collusion involved)	
Operator-Player	Cheating Related to Internal Misuse (Collusion involved)	

TABLE IV

ONLINE GAME CHEATING TAXONOMY: BY CHEATING PRINCIPAL

	Info Theft	Service Denial	Code or Data Modification	Masquerade	Fairness Violation
Design inadequacy in the game system	J	G	A	K	E, H, L
Design inadequacy in the underlying systems		G	F, M		
Operational failure	B, I, O		N		C, D

TABLE V

DISTRIBUTION OF CHEATING FORMS IN THE CAUSE-CONSEQUENCE MATRIX

committed by a single player independently, although some others involve collusion between one and his peer player(s) or an insider. However, for similar reasons to those given above, this observation also remains tentative.

Third, re-examining the taxonomy by consequence in Table III, in fact, no matter whether a cheating form results in either information theft, service denial, code or data modification, or masquerade, a fairness violation is caused and it gains a cheater some advantages over his peer players in the game. Therefore, the perspective of fairness appears to be essential in understanding security in applications such as online games. This echoes the result of [36] and can be easily explained as follows. On the one hand, fair play is essential to any game. Online gaming is not an exception, and fairness should be an inherent concern in its design. On the other hand, online players usually do not know each other, and they are often scattered across different physical locations. Therefore, the social structures preventing cheating in the non-electronic world are no longer in place for online games. It is security that can help provide an alternative mechanism for fairness enforcement.

Nonetheless, some game cheating problems cannot be solved by security techniques alone. For example, security mechanisms that usually can mitigate collusion in one way or another do not work well in the setting of online Bridge, in which collusive players can illicitly exchange card information via out-of-band channels such as telephone and instant messengers. Instead, a collusion detection approach based on artificial intelligence techniques appears to be essential in mitigating this devastating threat in online bridge communities [36]. Therefore, security plays an important but non-exclusive role in enforcing the fair play in online games.

V. Conclusion

Just as in other content-critical industries such as music or video, security was simply not an issue in the early years of computer games. Later on, security became an important technique for copy protection, and it still plays this role in some game market segments. However, we have found that much security effort of console game vendors has been directed towards locking in their customers to their hardware, and making strategic plays in the market.

The emergence of online games has led to fundamentally changed security requirements, and an important new security concern is online cheating. To understand cheating in online games, we have developed a taxonomy for it, in which the classification is made with respect to the underlying causes, consequences and the cheating principals. We have found that traditional aspects of computer security such as confidentiality, integrity, availability and authenticity are insufficient to explain cheating in online games. Fairness is a vital additional aspect, and the problem of its enforcement provides a convincing perspective for understanding the role of security techniques in online games.

References

[1] R Bartle, “Interactive Multi-User Computer Games”, Report commissioned by British Telecom Research Laboratories, December, 1990. Available at `http://www.mud.co.uk/richard/imucg.htm`.

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

[3] N Bushnell, “Relationships between fun and the computer business”, Communications of the ACM, Volume 39 Issue 8, August 1996.

[4] Christopher Choo, “Understanding Cheating in Counterstrike”, Nov. 2001. Available at `http://www.fragnetics.com/articles/cscheat/print.html`.

[5] C Morningstar and FR Farmer, “The Lessons of Lucasfilm’s Habitat”, in *Cyberspace: First Steps*, M Benedikt (ed.), MIT Press, Cambridge, 1990.

[6] D Alessandri (ed.), “Towards a Taxonomy of Intrusion Detection Systems and Attacks”, MAFTIA deliverable D3, Version 1.01, September 6, 2001. Available at `http://www.newcastle.research.ec.org/maftia/deliverables/D3.pdf`.

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

[8] 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.

[9] A Huang, “Keeping Secrets in Hardware: the Microsoft Xbox Case Study”, AI Memo 2002-008, AI Lab, MIT, May 2002.

[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] IV Krsul, “Software Vulnerability Analysis”, Ph.D. Thesis, Purdue University, Computer Sciences Department, 1998.

[12] CE Landwehr, AR Bull, JP McDermott and WS Choi, “A taxonomy of computer program security flaws”, ACM Computing Surveys, Vol.26 No.3, Sept. 1994. pp211-254.

[13] JC Laprie (ed.), Dependability: Basic Concepts and Terminology, Springer-Verlag, Vienna, 1992.

[14] U Lindqvist and E Jonsson, “How to Systematically Classify Computer Security Intrusions”, in *Proceedings of the 1997 IEEE Symposium on Security & Privacy*, Oakland, California, May 4-7, 1997. IEEE Computer Society Press. pp154-163.

[15] M Macedonia, “Will Linux be computer games’ dark horse OS?”, IEEE Computer, Vol. 34 No. 12, Dec. 2001. pp161-162.

[16] J Markoff, “A Long Time Ago, in a Lab Far Away”, The New York Times, Feb. 28, 2002.

[17] S McCreary and K Claffy, “Trends in Wide Area IP Traffic Patterns: A View from Ames Internet Exchange”, in *Proceedings of the ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management*, Monterey, CA, USA, Sept. 2000.

[18] M Müller, “Computer Go”, Artificial Intelligence, Vol.134, No.1-2 (Special issue on Games, Computers and AI), January 2002, pp145-179.

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

[20] NCsoft Corp., Lineage homepage, `http://www.lineage.com`.

[21] RM Needham, Personal communication, Oct. 2002.

[22] PG Neumann and DB Parker, “A Summary of Computer Misuse Techniques”, in *Proc. of the 12th National Computer Security Conference*, Baltimore, MD, 1989, pp. 396–407.

[23] Nintendo, “The History of Nintendo”, Available at `http://www.nintendo.com/corp/history.jsp`.

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

[25] K Poulsen, “Mitnick to Lawmakers: People, Phones are Weakest Links”, SecurityFocus.com News, March 2000. Available at `http://www.politechbot.com/p-00969.html`.

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

[27] T-M Rhyne, “Computer games and scientific visualization”, Communications of the ACM, Volume 45 Issue 7, July 2002.

[28] C Shannon, “Programming a digital computer for playing Chess”, Phil. Mag. 41, 1950. pp356-375

[29] D Sheff, Game Over: Nintendo’s Battle To Dominate An Industry, London:Hodder & Stoughton, 1993.

[30] K Tran, “U.S. video-game industry posts record sales”, Wall Street Journal, Feb. 7, 2002.

[31] A Turing, “Chess”, a subsection of chapter 25, Digital Computers Applied to Games, of Faster than Thought, ed. B. V. Bowden, Pitman, London (1953). Also available online at `http://www.turingarchive.org/catalogue.php3?category=AMT/B/7`.

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

[33] S Weyhrich, Chapter 3 in *Apple II History*, `http://apple2history.org/history/ah03.html`.

[34] S Wozniak, “Letters - General Questions Answered”, available at `http://www.woz.org/letters/general/91.html`.

[35] J Yan et al, “Security Issues in Online Games”, *The Electronic Library*, Vol. 20, No.2, 2002.

[36] J Yan, “Security Design in Online Games”, in *Proc. of the 19th Annual Computer Security Applications Conference*, IEEE Computer Society, December, 2003.