Detecting Cheating in Online Multiplayer Video Games

Zeke Krug
krug0102@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

Abstract

We explore methods to detect cheating in massively multiplayer online games with an emphasis on the use of blockchain technology. Cheaters have continued to beat anti-cheat software and tools implemented by developers designed to block cheaters. Currently, the most common approach for detecting cheating in video games is the kernel-level anti-cheat, which is software installed on the kernel. However, players have expressed their distaste for kernel-level anti-cheats as they believe it gives the developer too much access to their system. They can also introduce instability into a system. Thankfully, there are new methods for detecting cheating in video games being developed that address both of these problems. The method we will be focusing on is the use of blockchain technology to monitor player inputs to detect anomalies. Researchers did a study on their blockchain approach to see the effects on latency. They found that using blockchain resulted in an average latency of 254.5 milliseconds. The researchers also analyzed the blockchain approach's effectiveness at detecting cheating, finding that blockchain was no worse than the popular anti-cheat implementations out today.

Keywords: kernel, blockchain, smart contract, shim, anti-cheat, cheat detection, player assets, client-server, player-to-player

1 Introduction

Cheating, or the use of things like scripts, exploits, and other programs or software to give a player an unfair advantage, is as old as video games themselves. However, as video games have become mainstream, and the financial incentives (e.g. esports competition prizes and streaming revenue) have risen, keeping cheaters out of games has become a challenge that developers and communities are struggling with. High levels of cheating in a game can ruin a game's reputation and destroy its player base, as players will find other games to play.

Currently, developers use anti-cheats; software that ideally prevents cheating, or at the very least detects cheating so that action can be taken against the offender before honest players are affected [14]. Most developers opt for kernel-level anti-cheats, like Easy Anti-Cheat or Riot Games' Vanguard, which have drawbacks and community criticism. Unfortunately, the battle between game and cheat developers has been one-sided in favor of the cheat developers. While anti-cheats can catch many cheaters, cheat developers are constantly coming up with new ways to spoof anti-cheat mechanisms.

Thankfully, the introduction of new technologies, like blockchain, has opened the door to more robust anti-cheat systems. The idea behind using blockchain technology as an anti-cheat is to leverage the idea of peer consensus, that all connected users must validate a request before it is realized. So as an anti-cheat, peer consensus is used to validate player actions in real-time [11]. The major benefit of using blockchain is that it isn't invasive like kernel-level anti-cheats.

Although the approach using blockchain has the potential to become the new anti-cheat system, it faces drawbacks that may not make it viable today. The major issues of a blockchain anti-cheat system are 1) integrating blockchain anti-cheats would require a complete overhaul of the gaming industry 2) high validation latency or the time it takes the blockchain to assert fair play, and 3) increasing validation latency as the number of players increases [11].

In this paper, we begin by presenting some background about client/server architecture and the state of cheat detection. Then, to understand the benefits that blockchain presents over the popular kernel-level anti-cheat, we will discuss what kernel-level anti-cheats are, and their benefits and drawbacks. We will also explore the basics of blockchain, the approach proposed by Kalra *et al.*, and the results of their study on latency and effectiveness analysis. Finally, we will discuss the feasibility of blockchain anti-cheat systems.

2 Background

2.1 Client-Server Architecture

Most large, online multiplayer video games use the client-server architecture to connect players (see Figure 1). This involves a central server, which manages the world game state, and the client, which only needs to take user input and render the output state [2]. The client-server interaction can be broken down into four generic steps,

1. Player enters an input. For example, pressing a key on the keyboard or a button on a controller.
2. Client turns the input into a request and sends it to the server.

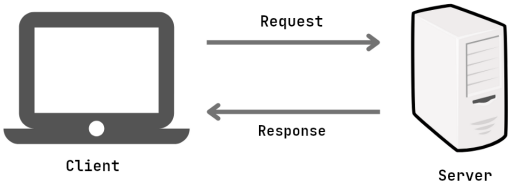

Figure 1. Simplified structure of client-server architecture [5]

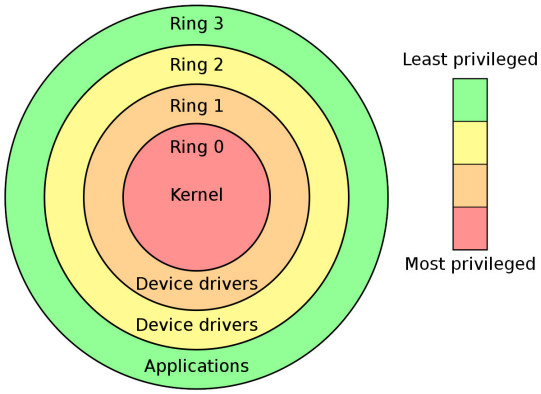

Figure 2. Structure overview of an operating system [6]

3. Server processes the request and sends a response to the client. In the world of games, the server updates the world game state.
4. Client processes the response and renders the game state based on the information it received from the server.

The benefits of the client-server architecture are that it makes directly manipulating the game state extremely difficult and allows many machines with different specifications to play the same game.

However, since the server can't tell if an input was made by a human or a program, faking or creating fake inputs, a.k.a cheating, is easy. And this is what many cheats do.

2.2 Cheating Software

Kalra *et al.* classifies cheating software into four main categories:

1. **Game** cheats are purposefully programmed into a game by the developers themselves. These are things like cheat codes.
2. **Application** cheats modify the game or its data files or read from/write to the game's memory. These cheats provide an unfair advantage in terms of information or inputs. Some examples would be things like aimbots or wallhacks.
3. **Protocol** cheats add, remove, modify, etc. packets, which are small chunks of a larger message [10]. The use of protocol cheats is quite limited and they can be easily detected/prevented with cryptographic protocols.
4. **Infrastructure** cheats tamper with game software like display drivers, or network hardware. These cheats can change how the game is rendered to reveal more information.

We will focus on the two most common cheats that games face, Application and Infrastructure cheats [11].

2.3 Anti-cheat systems

Currently, there are several "anti-cheat" systems that developers use as a way to keep cheaters out of their games. One common approach is player spectating and reporting, which allows players to spectate and/or report suspicious players [13, 15]. Another common approach is third-party software, like kernel-level anti-cheats, that is installed alongside a game.

3 Kernel-Level Anti-Cheats

A computer's operating system (OS) is comprised of several layers built upon one another (see Figure 2). The kernel is the lowest layer of a computer's operating system and manages the computer's hardware resources, memory allocation, and processes [7]. The key idea of the kernel is that it can look up into the layers above it, but the layers above can't look in.

So, with that key idea in mind, kernel-level anti-cheats work by scanning processes in the layers above the kernel to find suspicious programs or drivers. Games that utilize kernel-level anti-cheats require that all players have the anti-cheat installed to play the game. A player that refuses to install the anti-cheat will not be able to launch the game.

3.1 Benefits

The benefits of implementing a kernel-level anti-cheat are that there are plenty of them that are reliable and "good enough", along with player spectating and reporting, at keeping cheaters out of a game. Epic Games' Easy Anti-Cheat and PunkBuster are just two examples deployed in massive game franchises. This cuts down on game development costs as developers and publishers don't need to invest the time and money on designing and implementing their own anti-cheat.

3.2 Drawbacks

While trusted by developers and gamers alike, kernel-level anti-cheats have drawbacks that make them unappealing to

some. Unfortunately, the main selling point of kernel-level anti-cheats is also their biggest problem, they run on the kernel. This means that the developer is given high clearance in the operating system, which has led to privacy and security concerns, due to the kernel having access to all levels of the operating system. Another problem with kernel-level anti-cheats is that since they block programs or drivers that may alter the way the computer operates, the anti-cheat may falsely identify a program as cheating. Riot Games' Vanguard is a recent example showcasing the problems with kernel-level anti-cheats. Users reported that it would flag drivers and applications used for overclocking, fan control, and temperature monitoring as cheats [13]. Kernel-level anti-cheats can also introduce instability into a system, as any instability in the kernel means system-wide instability. Vanguard had a serious problem with instability issues when it first launched as well, causing OS crashes for many users, which led to players uninstalling it and the game. Finally, cheat developers have started to move their programs into the kernel, where kernel-level anti-cheats can't find them.

4 Using Blockchain as an Anti-Cheat

Blockchain is a "shared, immutable ledger that facilitates the process of recording transactions and tracking assets..." [9], a fancy record-keeping system. Every transaction must be verified by all other users on the blockchain. Once verified, the transaction is added to the record as a block. In our case of games, transactions are the inputs from a player, and assets are things that the player might own in the game (Figure 3 shows the basic concepts of blockchain in the context of games).

While commonly associated with cryptocurrencies like Bitcoin and Etherium, blockchain is much broader in scope, with many different kinds of blockchains [12]. Every blockchain has what is called a smart contract, which is a set of "if/when…then…" statements that specify the "rules" that the transactions of that blockchain must follow [8]. A transaction that does not meet the specifications of the smart contract cannot be verified and is not added to the record.

The cheat-detection approach proposed by Kalra *et al.*, uses a specific kind of blockchain called Hyperledger Fabric. Hyperledger Fabric offers four features that are appealing to the structure of games [3, 4]:

1. Open smart contract model would allow developers to set their own rules for their games.
2. Low validation latency.
3. The ability to create channels. The parallel in games would be game lobbies.
4. Versioning of smart contracts gives the developer the ability to alter the rules of the game, akin to a game patch.

4.1 Smart Contract

In the blockchain approach, the smart contract takes the place of the server, and it contains all the logic that goes into managing player assets. It would also specify constraints on player inputs. Kalra *et al.* propose that a smart contract template be provided to developers for them to set the base rules of the game. The smart contract is crucial in cheat detection as it specifies the permitted inputs. Any inputs that do not meet the smart contract specifications cannot be validated and are not processed.

4.2 Shim

The shim is the interface between clients and the Smart Contract and is special to Hyperledger Fabric. Every person on the blockchain, or in our case, the game lobby, has a shim. A special shim called the initiator shim belongs to the person who created the lobby and set the rules of the game via the smart contract. The importance of the initiator shim in cheat detection is that it deploys the smart contract on every peer, meaning that everyone in a session is playing according to the same rules.

4.3 Latency Study

The study done by Kalra *et al.* looks at the latency of their approach in ten multiplayer First-Person Shooter (FPS) games on Steam. They chose to use FPS games in the study because they believed that FPS games represented a worst-case scenario for latency due to their requirement of real-time consensus on every event update [11]. For each of the ten games they measured three things (see Figure 4):

1. Average and maximum number of players per game session,
2. Average latency in milliseconds (ms), and
3. Client tickrate, the number of times the client updates per second.

The average and the maximum number of players per game session were calculated using data from online game trackers. The average latency and the tickrate of the client were taken from the Steam console. These three values are all related to each other. Higher tickrate and player counts lead to increased latency, and vice versa, with lower tickrate and player counts leading to less latency.

It is important to note that in a full implementation of a blockchain anti-cheat, there would be no server. In the study, Kalra *et al.* were forced to connect to servers as the chosen games use the client-server architecture. Due to this point, they also looked at the latency distributions of the servers that were available to them during the study (see Figure 5). It is possible that the high average latency that was observed was due to the high latencies of the servers they were connecting to.

The key finding of the study was that for a successful game session, which they considered as a session with no

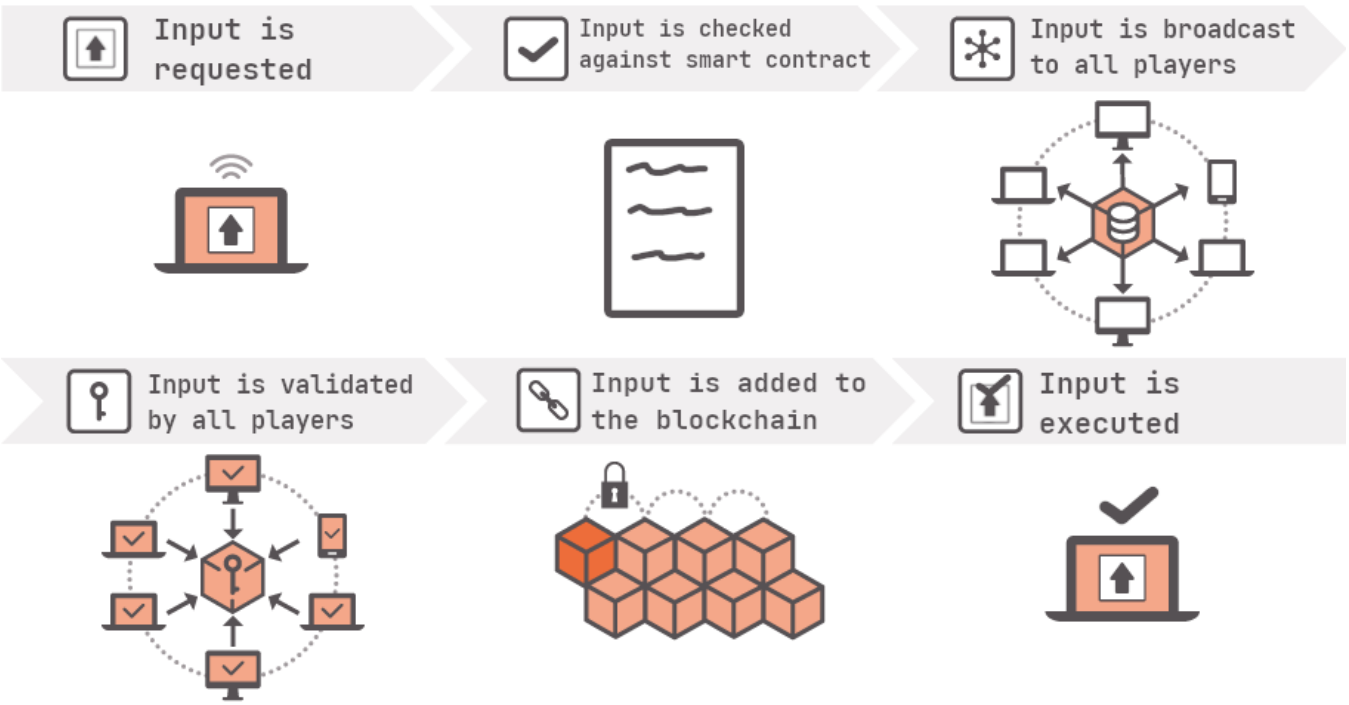

Figure 3. Basic concept of how the blockchain anti-cheat works. Reconstructed from [1]

Game	# Players Avg.	# Players Max	Average Latency (ms)	Client Tick Rate
Counter-Strike 1.6	25.49	32	241	30
Counter-Strike: GO	18.93	63	240	64
Counter Strike: Source	14.84	64	234	66
Day of Defeat	4.59	30	245	30
Double Action: Boogaloo	0.42	17	288	30
Half-Life	1.75	31	258	60
Half-Life 2: Deathmatch	0.99	64	244	30
Left 4 Dead 2	2.38	24	272	30
Team Fortress Classic	0.41	15	253	30
Team Fortress 2	5.63	32	270	30

Figure 4. Latency measurements from latency study [11]

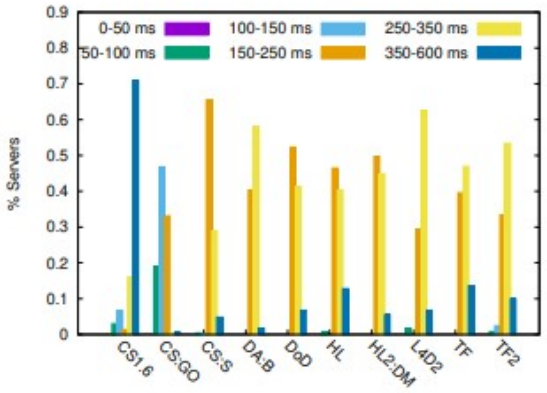

Figure 5. Latency distribution of servers for each game available to Kalra *et al.* [11]

perceived lag or jitter for ten minutes of play, the average latency was approximately 250ms [11].

4.4 Effectiveness

What makes this approach appealing as an anti-cheat is that it validates game states instead of looking for suspicious programs. It does this by comparing the game state reported by the client, and the game state at the server.

To analyze the theoretical effectiveness of their approach, Kalra *et al.* look at data from 25 sessions of Doom, a popular FPS game, provided by the community. The shim was used to generate events based on that data. They then logged and classified all of the events (~350 thousand) into five categories – armor, health, location, shoot, and weapon. Based on this data, Kalra *et al.* found that their approach using blockchain does no worse than popular anti-cheat systems (see Figure 6). They do admit, however, that blockchain would still require a client-side anti-cheat system, like a kernel-level anti-cheat, to detect common application cheats.

4.5 Drawbacks

Due to how different the blockchain approach is from the client-server approach, it would require a shift in the entire gaming industry involving publishers, developers, and players. [11]. There's also a major problem with latency (see Figure 4) as low latency is ideal. An average latency of 250ms

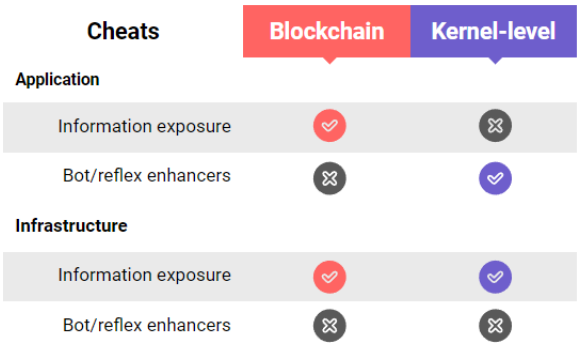

Figure 6. Effectiveness of the blockchain approach vs. other anti-cheat approaches in detecting different kinds of cheats [11]. The leftmost column represents the blockchain approach, the middle column a robust client-server architecture without anti-cheats, and the rightmost column kernel-level anti-cheats.

is high, but that average could be due to the high server latencies that the Kalra *et al.* had to connect to during the research process (see Figure 5).

4.6 Benefits

The smart contract gives the blockchain approach some appealing benefits. Since asset management is done on the smart contract, which is not part of the actual game code, the introduction of new skins or weapons (in the case of FPS games) would only require a change to the smart contract. The ability to create unique experiences would also be possible by modifying the smart contract. Kalra *et al.* modified weapon appearance and weapon functionality, like giving a weapon infinite ammo, by changing the smart contract. They also believe that using the same approach one could introduce "new" enemies into the game, making the game harder than intended [11].

5 Conclusions

The blockchain approach proposed by Kalra *et al.* showed promise as a future anti-cheat system as it was able to be no worse than current anti-cheat systems. It also gives developers and players the ability to alter the game to fit their needs and/or likes through the smart contract. However, it has two problems that don't make it viable today. The first is high latency. Due to the need for peer consensus, which requires input validation on every client, latency increases to borderline unplayable levels. The other is that the blockchain approach would still require a client-side anti-cheat system, like a kernel-level anti-cheat, to detect common application cheats.

It would be interesting to see if the same experiment done on servers with lower latencies would result in better results. The theoretical effectiveness analysis confirms some of the proposed benefits of using blockchain as an anti-cheat, however, it is still theoretical, and we would like to see how the running system stacks up against cheats.

To summarize, the blockchain anti-cheat approach becomes viable if the answer to these two questions is yes:

1. Do players want to move away from the intrusiveness of kernel-level anti-cheats?
2. Can a fully implemented blockchain anti-cheat have low latency?

If players can accept kernel-level anti-cheats, the blockchain approach becomes less viable, as it doesn't provide any effectiveness benefits over kernel-level anti-cheats. However, if players want to move away from kernel-level anti-cheats and a fully implemented blockchain anti-cheat has latency on par with what we see now, the blockchain approach becomes a potential alternative. It removes the intrusiveness of kernel-level anti-cheats and provides similar levels of effectiveness.

Acknowledgments

I would like to thank my advisor Kristin Lamberty and reviewer Chineng "Cookie" Vang for their help and insight throughout the research process.

References

[1] Kilroy Blockchain. [n. d.]. *WHAT IS BLOCKCHAIN? HOW DOES BLOCKCHAIN WORK?* https://kilroyblockchain.com/what-is-blockchain

[2] Stefano Ferretti and Gabriele D'Angelo. 2018. *Client/Server Gaming Architectures.* Springer International Publishing, Cham, 1–2. https://doi.org/10.1007/978-3-319-08234-9_272-1

[3] Linux Foundation. 2018. *An Introduction to Hyperledger.* https://www.hyperledger.org/wp-content/uploads/2018/07/HL_Whitepaper_IntroductiontoHyperledger.pdf

[4] Linux Foundation. 2020. *Hyperledger Fabric.* https://www.hyperledger.org/wp-content/uploads/2020/03/hyperledger_fabric_whitepaper.pdf

[5] Shubham Gautam. 2022. *Client Server Architecture.* https://www.enjoyalgorithms.com/blog/client-server-architecture

[6] Hertzsprung. [n. d.]. *File:Priv rings.svg.* https://commons.wikimedia.org/wiki/File:Priv_rings.svg

[7] Jeanelle Horcasitas. 2021. *What Is a Kernel?* https://www.digitalocean.com/community/tutorials/what-is-a-kernel

[8] IBM. [n. d.]. *What are smart contracts on blockchain?* https://www.ibm.com/topics/smart-contracts

[9] IBM. [n. d.]. *What is blockchain technology?* https://www.ibm.com/topics/what-is-blockchain

[10] Cloudflare Inc. [n. d.]. *What is a packet? | Network packet definition.* https://www.cloudflare.com/learning/network-layer/what-is-a-packet/

[11] Sukrit Kalra, Rishabh Sanghi, and Mohan Dhawan. 2018. Blockchain-Based Real-Time Cheat Prevention and Robustness for Multi-Player Online Games. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies* (Heraklion, Greece) *(CoNEXT '18).* Association for Computing Machinery, New York, NY, USA, 178–190. https://doi.org/10.1145/3281411.3281438

[12] Alan G. Labouseur, Matthew Johnson, and Thomas Magnusson. 2019. Demystifying Blockchain by Teaching It in Computer Science: Adventures in Essence, Accidents, and Data Structures. *J. Comput. Sci. Coll.*

34, 6 (apr 2019), 43–56.

[13] Anton Maario, Vinod Kumar Shukla, A. Ambikapathy, and Purushottam Sharma. 2021. Redefining the Risks of Kernel-Level Anti-Cheat in Online Gaming. In *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. 676–680. https://doi.org/10.1109/SPIN52536.2021.9566108

[14] Ruan Spijkerman and Elizabeth Marie Ehlers. 2020. Cheat Detection in a Multiplayer First-Person Shooter Using Artificial Intelligence Tools. In *2020 The 3rd International Conference on Computational Intelligence and Intelligent Systems* (Tokyo, Japan) *(CIIS 2020)*. Association for Computing Machinery, New York, NY, USA, 87–92. https://doi.org/10.1145/3440840.3440857

[15] Qinghao Zhang. 2021. Improvement of Online Game Anti-Cheat System based on Deep Learning. In *2021 2nd International Conference on Information Science and Education (ICISE-IE)*. 652–655. https://doi.org/10.1109/ICISE-IE53922.2021.00153