Penetration Testing of Enet Protocol Implementation in Online Games

MSc Internship
MSc Cybersecurity

Arjun Vijaypal Singh
Student ID: 21213330

School of Computing
National College of Ireland

Supervisor: Mr. Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Arjun Vijaypal Singh
Student ID:	21213330
Programme:	MSc Cyber Security **Year:** 2022-2023
Module:	Industrial Internship
Supervisor:	Vikas Sahni
Submission Due Date:	18 September 2023
Project Title:	Penetration Testing of Enet Protocol Implementation in Online Games
Word Count:	5485 **Page Count** 22 Page

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Arjun Vijaypal Singh

Date: 16/09/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Penetration Testing Approach on Enet Protocol Implementation in Online Games

Arjun Vijaypal Singh
21213330

Abstract

With the increasing popularity of multiplayer online games, efficient and effective networking methods are now required to make sure that group games run smoothly. The ENet protocol's reliability and performance advantages have led to its widespread adoption as the preferred network communication method in online gaming. The use of the ENet protocol in gaming systems can introduce security vulnerabilities that threaten user data, game integrity, and network security. This research investigates in depth the security vulnerabilities of the ENet protocol implementation in online games. It evaluates the security posture of games using the ENet protocol and identify potential vulnerabilities by performing the extensive penetration testing techniques that could be exploited by malicious actors. Data transmission, network protocols, encryption mechanisms, input validation, and server-side security measures have been analysed as part of the study.

Keywords: Game Penetration Testing, Enet Protocol Security, Risk Assessment.

1 Introduction

Millions of people over the globe participate in multiplayer online gaming, which has developed into an essential component of the contemporary landscape of digital entertainment. Multiplayer games depend significantly on networking protocols that are both fast and reliable to offer a flawless and immersive gaming experience for all players. Because of its lightweight design, low-latency qualities, and built-in dependability features. An ENet protocol has become a popular option for network communication in online games. This popularity may be attributed to its low-latency characteristics. ENet provides a solid method for delivering game data via UDP (User Datagram Protocol), which strikes a balance between reliability and performance. This solution is offered to developers by ENet.

Nevertheless, considering the growing number of cyberattacks and the critical nature of protecting player data, it is necessary to conduct an exhaustive analysis of the security risks connected with the use of the ENet protocol in online games. Although ENet provides dependability and efficiency, the complex structure of online gaming environments adds possible attack vectors that might jeopardize the privacy of players, the integrity of games, and the security of the network.

The primary objective is to evaluate the security posture of games that utilize the ENet protocol and to provide game developers, network administrators, and security professionals with actionable insights to improve the security of their gaming infrastructures. The purpose

of penetration testing is to identify pervasive security flaws, potential attack paths, and exploitable vulnerabilities through a comprehensive analysis.[1]

Moreover, the objective of this research endeavour is to provide practical pen-testing techniques for the ENet protocol. By incorporating secure coding practices, network monitoring techniques, and encryption mechanisms.

Research Question: What are the vulnerabilities within the online game built on ENET protocol, and how can they be exploited?

2 Related Work

In the world of massive multiplayer online games (MMOGs), the balance between gameplay enjoyment and security is a major concern. Exploiting vulnerabilities, such as client manipulation and teleportation bugs, can give unethical participants an unwarranted advantage. These difficulties parallel those encountered in online financial security, highlighting the need to protect virtual assets. Multiple studies have described the stages of manipulation, culminating in a nuanced risk assessment diagram that facilitates understanding and mitigation of security threats. Efforts to fortify MMORPGs include various techniques such as encryption, randomized card distribution, and countering purposeful game disruptions, highlighting the need for comprehensive security measures. Additionally, the combination of real-world transactions with in-game economies increases the importance of advanced security strategies, demanding detailed evaluations and targeted defences to ensure the stability and integrity of these virtual worlds.

(Hu & Zambetta, 2008) explained the means of attack and its countermeasure for Massive multiplayer online games cheating, abusing, and collusion in the games' virtual assets. The author also showed that the attacker can manipulate the client infrastructure to again an unfair advantage that a normal player can't get.

(McGraw & Hoglund, 2007a) have described the vulnerability which also exists in the normal banking application. Race condition occurrences are rare however, it allows an attacker to race against time to execute the action beyond the time and state. The paper describes a specific example of state manipulation known as tele hacking. In this attack, the player character's coordinates in the game (e.g., World of Warcraft) are manipulated to teleport the character across vast regions of the game map. By directly altering the location parameters in the memory, cheaters can bypass normal movement and instantly appear in different areas. The manipulation can even be made subtle by keeping the teleportation within proximity, resembling normal movement rather than an obvious exploit.

Similar cheats and frauds are also major security issues in multiplayer online games the author have discussed by (Y. C. Chen, Hwang, et al., 2005). they introduced Knut Hakon's categorization of cheating into five stages: unknown cheats, known cheats being exploited by

[1] http://enet.bespin.org/Features.html

few, known cheats with no patches known within a large group, known cheats with patches, and known cheats with patches applied. These stages indicate the progression of cheating methods from unknown and unpatched to widely known and potentially patched by the game vendor. The authors then present an improved risk diagram based on Knut Hakon's stages, indicating the varying levels of risk associated with each stage. The diagram illustrates that even when cheats are known and patches are applied, there can still be a minor risk to other users due to the possibility of cheating before the patches are implemented. Next, the classification of cheating means is discussed.

(J. J. Yan & Choi, 2002) summarized eleven categories of fraud, and extend it to seventeen categories based on real criminal incidents. The seventeen categories of cheating include fraud by collusion, fraud by alliance, fraud by abusing policy, cheating related to virtual property, fraud by compromising passwords, fraud by denying service to peer players, fraud due to lack of authentication, fraud-related internal misuse, fraud by social engineering, fraud by modifying game software or data, fraud by exploiting bugs or design flaws.

Some examples of online security failures described by (J. Yan, 2003) which are as follows: Firstly, it emphasises the dangers of transmitting card information without encryption, highlighting the significance of using encryption to prevent surveillance. In addition, the assessment examines potential vulnerabilities in the client programme that would allow fraudsters to view their companions' and opponents' cards. Drawing parallels to the game SIGUO, a proposed solution involves storing private card data on the server. Concerning the impartiality of online Bridge card dealing, the review emphasises the importance of a robust pseudorandom number generator to prevent exploitative practises such as card sharing and expected deals, which could compromise the integrity of online games. In addition, the issue of premeditated player abandonment to prevent losing rank points is analysed, and it is suggested that this form of deception be mitigated by monitoring and displaying statistics on incomplete games.

(Bono et al., 2009) explained about how online games, especially MMORPGs (Massively Multiplayer Online Role-Playing Games), can be dangerous to your security. The focus is on how hard these games are to play and how easy they are to attack. There are also two case studies: Anarchy Online/Age of Conan and Second Life. The author talks about the effects that cheats and bugs could have on online games, such as the financial effects on players who buy and sell in-game things with real money. The author stresses how important it is to deal with cheating to keep players happy and keep game companies from getting hurt. It is talked about how complicated MMORPGs are, with a huge number of features and things that players can do. This makes it harder for writers to keep the game world under control because there are more ways for things to go wrong. MMORPGs have a large attack surface because there are many ways for clients and servers to talk to each other, as well as the use of third-party plug-ins and add-ons, support for many file types, and the inclusion of external apps and P2P communication. Each of these traits opens up a possible hole that an attacker could use. In the case studies of Second Life and Anarchy Online/Age of Conan, the author gives detailed examples of flaws and threats. The author talks about the dangers of user-made material, video files, in-game dialogue, and tools that run on their own. Particular risks like directory access flaws and buffer overflows are called out. To reduce these risks, the author suggests using

security-aware development techniques, such as validating input, putting plug-ins and add-ons in sandboxes, testing plug-ins and add-ons thoroughly for flaws, and teaching players how to act safely in virtual worlds.

Packet tampering which is mainly focused in this research has been discussed by (Laurens et al., 2007)When packets leave the computer, and their contents are changed if better play could have happened. The game code and surroundings on the client side don't change at all. Most of the time, an implementation is a completely different machine that acts as a proxy server for the game-client system. The proxy server will run a complex program that looks at bits coming in and going out to find out what's going on in the game world. This is how the proxy gets basic information, like where people are in the world and which way the cheater is facing. If it sees an outgoing command packet where the player tried to shoot an enemy player but missed, the proxy programme could change the packet to show a different way the player was facing, which would have made the shot hit. The game server has no reason to doubt this packet, which makes packet-tampering methods very hard to spot and very hard to do.

On the other hand (Ban & Zhao, 2012) has discussed about the transaction security issue One thing that makes an MMORPG virtual world stand out is its ability to handle transactions. In the same way that transactions in the real world affect the social economy, transactions in the game world affect the economy in the game world. This is both an important way for players to connect with each other and an important result of their interactions.

Existing research addresses online gaming security concerns such as fraud, manipulation, and strategies, however, lacks comprehensive defence assessments and penetration testing strategies. This study aims to address the problem by proposing a comprehensive testing approach for ENet-based games, identifying vulnerabilities. This research contributes to the advancement of secure online gaming environments by providing a novel method for identifying security defects using specialized tools amid the ever-changing gaming world and associated risks.

Table 1. Researcher Contribution

Authors	Key Points
Hu & Zambetta, 2008	Provided Counter measure various cheating method
McGraw & Hoglund, 2007a	Discussed Race condition vulnerability
J. J. Yan & Choi, 2002	Problems in online games
Ban, Y., & Zhao, Y	Game logic and transaction security
Chang, H. B., Kwon, H. J., & Kang, J. G.	Case study of vulnerabilities in online games
Chang, H., Park, J. H., & Kang, H	Technical Counter measure for game hacks
Cho, C.-S., Sohn, K.-M., Park, C.-J., & Kang, J.-H	Scenario-based Control of Massive Virtual Users
Davis, S. B., & Price, W. J	Security issues with third party vendor

Duh, H. B. L., & Chen, V. H. H	frameworks of cheating
Kasim, S., Valliani, N., Ki Wong, N. K., Samadi, S., Watkins, L., & Rubin, A	Cyber Security Strategy in games
McGraw, G., & Chow, M. (2009)	Security in software

3 Research Methodology

This research methodology uses the approach of performing penetration testing on Enet Protocol-based desktop online games. This is basically an upper layer of the UDP protocol, and engines such as Photon have built-in support for it. The research project analyses a Photon Pun-based game and introduces a Mega proxy utility that supports UDP, TCP, and ENET protocols. This research uses a sequentially organized methodology discussed by (Goutam & Tiwari, 2019)

The Tools used multiple dependencies and several supporting modules to work properly. A quantum module supports various packages and messages to parse the data via the Enet channel. Then the channel communicates with the server. An approach for the penetration testing tool is to intercept the game request with the Mega-Proxy script and make necessary changes in the tampering script to intercept the messages and tamper a specific function accordingly using Python's if-else statements. Once the message tampers ongoing request changes app[2]lies as being an unauthorized player, just like modifying the request in the burp suite and replying to the request. The methodology hasn't been discussed in the literature review paper yet. however, the security issues and cheating methods were discussed that led to hacking the game. The PUN (Photon Unity Networking) facilitates the ability of developers to establish and manage personalized events within the networking framework of Unity games. The utilization of .NET callbacks enables game developers to achieve synchronization of game state modifications and activation of targeted actions among interconnected clients. The utilization of these callback functionalities has the potential to greatly augment the quality of real-time multiplayer gaming experiences through the facilitation of seamless communication and synchronization among players. This, in turn, contributes to the creation of a more immersive and pleasurable gameplay environment.

[2] https://doc.photonengine.com/pun/current/getting-started/dotnet-callbacks

3.1 Working Mechanism of Megaproxy

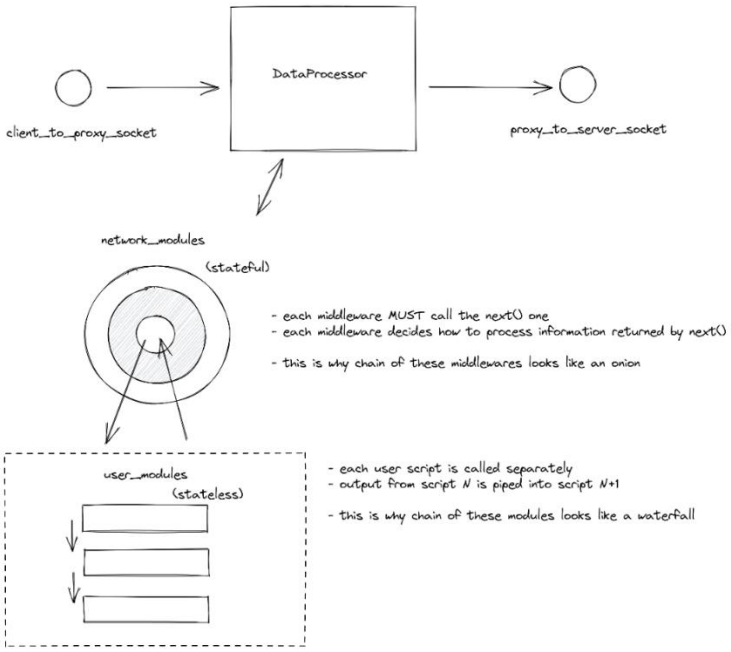

Figure 1 Working Mechanism of Megaproxy

[3]

Mega Proxy is a highly adjustable and extensible proxy architecture that allows clients and servers to communicate securely using TLS and DTLS encrypted protocols. Mega Proxy, the core class, is at the heart of the proxy and must be initialised using a Config object. The Config object contains crucial information such as the project name and a list of listeners, each expressed as a "5-tuple" (source IP, source port, destination IP, destination port, protocol). The configuration items are required to connect and proxy to the game server. The config files are specifically made in YML.

[3] https://confluence.magicmedia.studio/display/SEC/How+MegaProxy+works

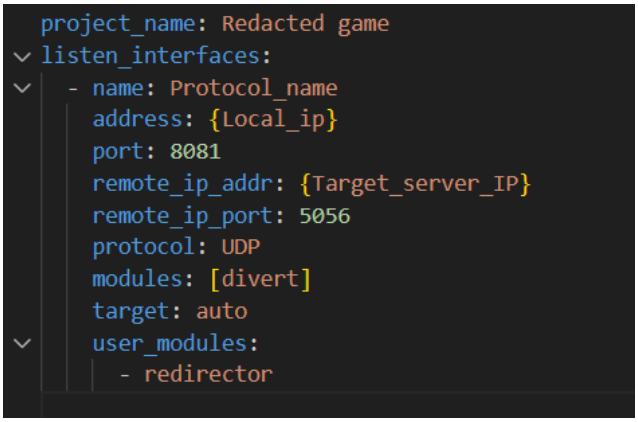

Figure 2 Config file

Users must supply the project name and specify one or more listeners using the Config data class to set up a proxy. Each listener may be associated with either TCP or UDP protocols, or any other protocol that supports TLS encryption and is supported by the game. The configuration files are written in an easy-to-understand style, and examples are provided for reference.

The main.py script constructs the essential Config object and initialises Mega Proxy after getting the configuration file via the command-line utility. Mega Proxy instance examines the listener configurations and launches distinct listener threads for each protocol. Depending on the protocol, TCP or UDP, the processes execute tcp_proxy() or udp_proxy(). The tcp_proxy() function creates two sockets for TCP connections: one to connect to the client and another to connect to the server. If TLS encryption is required, a TLS handshake is performed at this juncture. Similarly, the udp_proxy() function sets up two sockets and initiates a TLS handshake if necessary for UDP connections.

Once client and server TLS sessions are established, MegaProxy initiates a data processing cycle. Each recipient has its own instance of DataProcessor, which applies user-defined scripts (user_modules) and middleware's (modules) to incoming data. user_modules are custom scripts stored in the listeners' directory, whereas the middleware's are located in the modules' directory. The client_handler method runs the data processing loop, which sends and screens data between the client and the server. Data Processor object linked with the listener handles the data that comes from the client. The data that has been analysed is then sent to the computer, and vice versa.

Mega Proxy can be made to do more by supporting different middlewares and scripts made by the user. Users can add new middleware's to the./modules directory and make custom scripts for specific listeners in the./listeners/project_name>/listener_name> directory.The DataProcessor gets these modules and scripts as Python modules on the fly so they can be used when handling data.

4 Design Specification

The Design of the tool is complex as it supports various protocol built in python language also it requires several dependency modules which can install by using the command called pip requirement.txt.

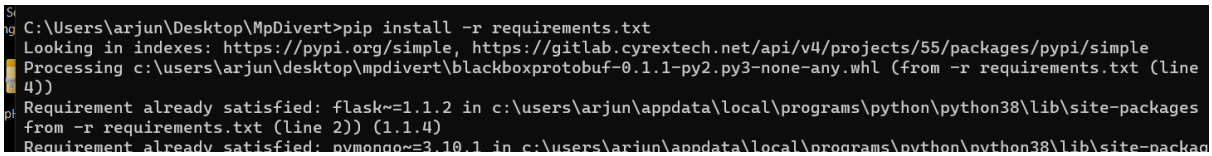

Figure 3 Requirements for Mega proxy

The structure of the tool is as follows where config are based on YAML template and it contains the configuration such as IP address of local and target devices. The Listener contains the actual project name and its tampering script were modifying the script according to tamper and alter the data for game. MP contains two kinds of modules: user programmes and network modules and the Protocol directory will contain each of the protocol which is implemented in MP such as TCP, UDP, and Enet and each protocol has its own way of working and tampering.

In contrast, "Modules" are middlewares that can be connected to the traffic processing engine. The initial concept behind these "building blocks" was to combine network protocol layers to construct something more complex.[4]

Directory	Purpose
configs	Configuration files in JSON format
docs	Misc files for Confluence documentation
listeners	Directories in format project_name/listener_name
modules	Modules that you can use in config/modules. Contains: logger (saves all traffic to Mongo), mp_dynamic (defines new listeners in runtime), http2 (decodes http2 traffic)
protocols	Implemented protocol libraries
sessions	Traffic sessions (either .pcap or .mpsession)
src	The sources of MegaProxy. You will rarely need to look at them

Figure 4 Structure of Mega proxy

Tool support various protocol for interception and tampering date of the game depending upon the game flow and protocol. Here, the flowchart has been developed specifically for the Enet Protocol on top of that the Photon module developed by exit games which usually runs on Enet protocol which is given as follows:

[4] https://confluence.magicmedia.studio/display/SEC/MegaProxy+directory+structure

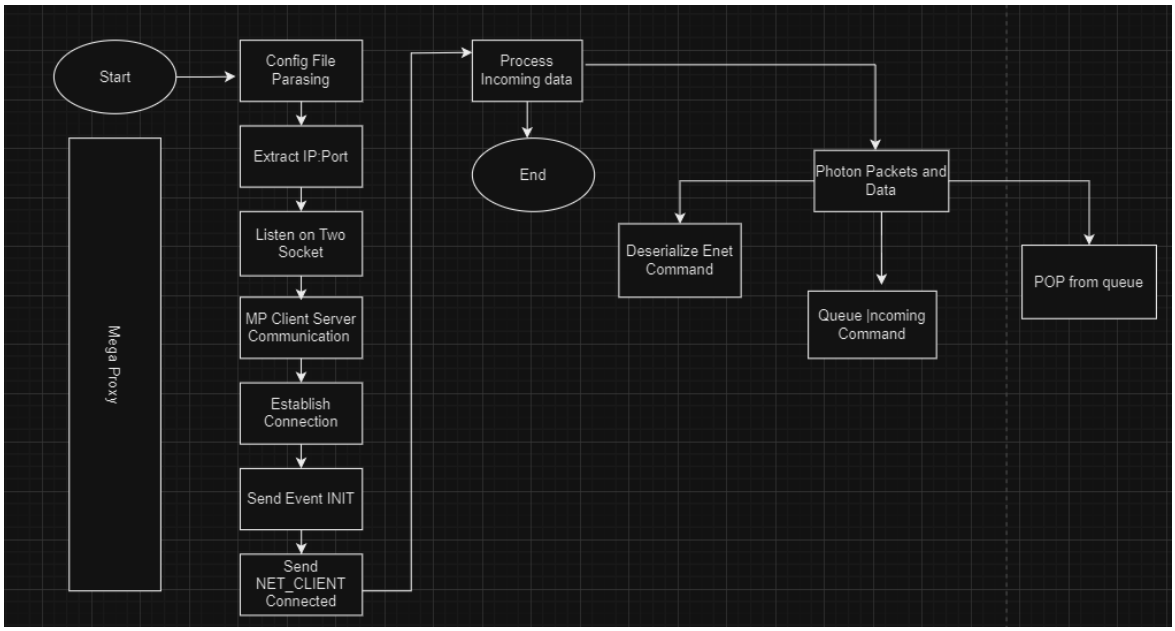

Figure 5 Workflow of Connection with Photon peer

The presented flowchart illustrates the procedural steps involved in the working mechanism of a real-time multiplayer gaming networking system, which serves as the focal point of investigation within a thesis research endeavour. The objective is to develop a network infrastructure that enables seamless communication between the multiplayer component ("mp"), the game server, and the connected clients. The flowchart commences by initiating the parsing process of a configuration file that encompasses network-related parameters, including server IP addresses and port numbers. The first step in the process ensures that the networking settings can be readily modified to align with the specific demands of the game. After parsing the configuration, the IP:PORT pairs that have been extracted are utilised to establish connections between the multiplayer component and both the server and clients .

To facilitate the management of these connections, two distinct sockets are established, with each socket specifically designated for the purpose of facilitating data transmission between "mp" and the server, as well as between "mp" and the clients. The bidirectional communication channels, denoted by arrows labelled "mp <-> Server" and "mp <-> Client," symbolise the ongoing exchange of data between the multiplayer component and the server and clients, respectively. At this point, an important moment is presented, wherein the server enters a state of the event for incoming client connections. If a client establishes a successful connection, the flow of operations proceeds uninterrupted. Conversely, if the client fails to connect, the server will persist in a state of waiting. Upon establishing a connection with a client, the process of event handling is executed by transmitting an "INIT" event to all modules that have been registered. This occurrence functions as a means of informing other components of the application that a fresh client has established a connection and is prepared for engagement.

In addition, subsequent to the initialization process, a "NET_CLIENT_CONNECTED" event is dispatched to designated recipients. This event has the potential to contain supplementary information pertaining to the associated client or any pertinent data necessary for the game. The central aspect of
the networking functionality is centred on the management of incoming data, which is denoted by the process known as "Photon Packets and Data." The utilisation of the ENet library, a dependable User Datagram Protocol (UDP) networking library that is frequently employed in the context of real-time multiplayer gaming, enables the efficient deserialization of incoming data commands. In

order to establish efficient data processing and mitigate congestion, a queuing system is implemented. To ensure orderly execution and minimise the potential loss of data in situations with high traffic, incoming commands are systematically queued and processed sequentially.

The flowchart is considered complete when all the necessary steps have been executed, indicating the successful integration of the real-time multiplayer gaming networking system as an essential element of the thesis investigation.

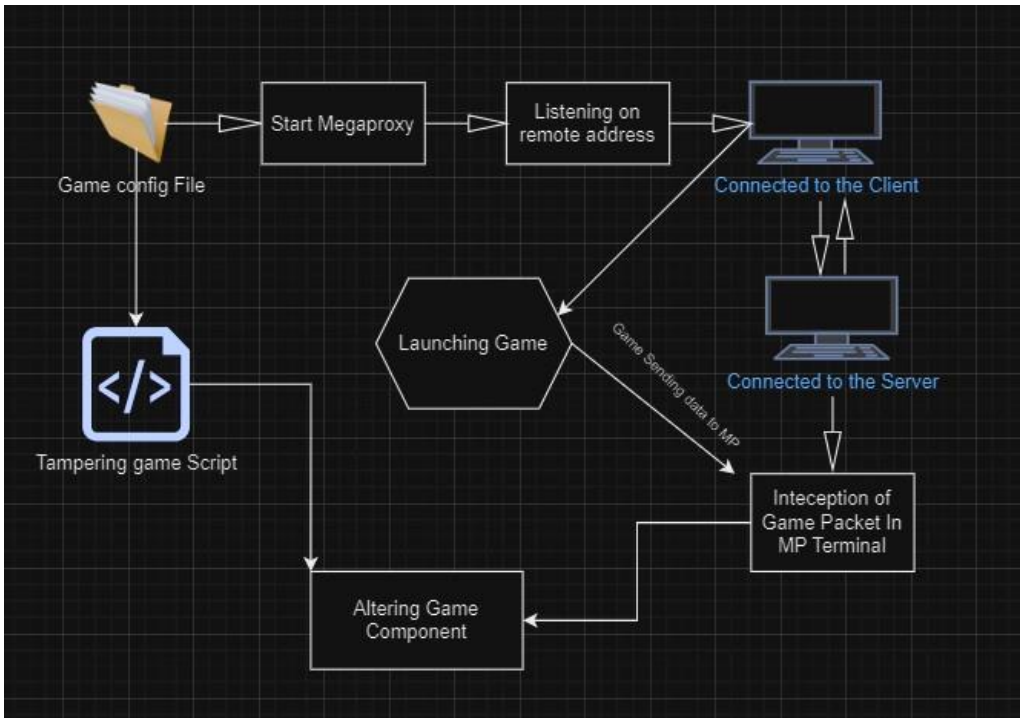

Figure 6 Game Interaction with Megaproxy

The following diagram illustrates the How mega proxy will interact with the game client and respond back with the packets. Once the configuration file has been setup and Mega Proxy is started with CLI it listens on the target game IP and port. Once the game has been started. It immediately sends data to the Proxy server that is Mega Proxy. Tampering script is the essential to alter the game data to start the initial state of penetration testing. It requires the knowledge of Python and the game protocol knowledge to be able to work on the testing approach.

5 Implementation

For the research purpose, It was observed that it was required to use the specific game which is built on Enet protocol. As the Mega proxy already supports the Enet protocol and its functionality. However, Mega Proxy has multiple branches developed accordingly depending upon the game support. Because of its existing compatibility with the Protocol in the Mega proxy, the MPDivert Branch was chosen. Wireshark was used to capture IP and port data during the game exploration phase, following the conclusion of all necessary research features and game development. This information is crucial for input file configuration.

5.1 Project Planning and Language Selection

The main goal of the project is to make a Tampering Tool that can work with data and change it based on a script. The tool is made with Python 3 and works with the Photon Quantum package from exit games. Mega proxy can handle the Photon package because it is built on Enet by default. Mega proxy is built on Python, so it was necessary to run a version of Python higher than 2.7 to make the change into a script. The Enet Protocol is used to make the game that has been considered for testing.

5.2 Development

The execution of an entire operation is deeply connected with the protocol section, particularly the Photon Quantum's message function. Within this feature, a crucial decision-making process happens in which the development of tamper scripts customised to specific message components takes place. These components include the presentation of participant data, the implementation of the "Join" operation, and the transmission of disconnect messages. These components are seamlessly incorporated into the counterfeit script using the capabilities of the message feature. This integration allows the script to intercept and manipulate communications, providing a means to alter the transmitted data. By judiciously utilizing the message feature and integrating it into the compromised script, the system obtains the capacity to intercept, examine, and manipulate messages. This manipulation grants the ability to modify data, thereby altering the trajectory of communication and interaction in a networked environment.

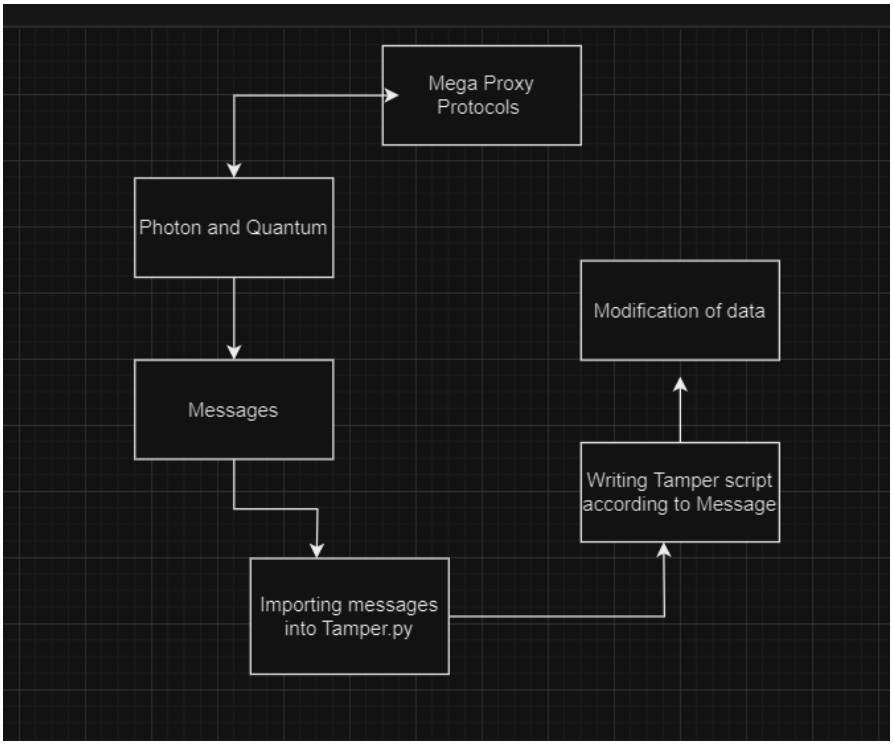

Figure 7 Script Development of tamper.py

5.2.1 Tampering.py

This module encompasses a set of functions designed to manipulate various categories of Quantum Photon messages and deterministic tick inputs. The script starts by loading essential modules and classes from the Quantum Photon protocol library. The use of the loguru logger for the purpose of logging messages seems to be evident. Tampering.py and Redirector.py are interconnected. One input is connected to another output via a pipeline script. Tampering.py cannot change the out of put redirector.py. Typically, all the tampering and extending occurs in tamper script and redirector produce the output.

Definition of Function: tamperMessage(message: Message): The function accepts a Message object as a parameter and seems to alter the message's content according to its type. The system manages many sorts of messages, including Join, SessionConfig, SetPlayerData, and Command, via the manipulation of specified characteristics or the instantiation of new message instances. Multiple conditional checks and adjustments are being implemented on these messages.

The objective of this code is to do testing and experimentation involving the manipulation of Quantum Photon protocol messages and deterministic tick inputs. This research thesis encompasses the use of Quantum Photon technology to simulate or modify network traffic inside game-related contexts.

5.2.2 Redirector.py

The script Redirector.py was created with the purpose of establishing a network redirection proxy that enables the manipulation and observation of network traffic occurring between a client and server. The real-time communication is facilitated by using the Quantum Photon protocol library. The script starts by importing a range of classes and modules from the Quantum Photon protocol library, among additional custom modules.

The script uses global variables, namely to_server_socket, to_client_socket, tampering, DHEStarted, DHEFinished, encryptionQueue, and tampered, to effectively handle the status of the proxy and the tampering procedure.
A custom class called MyTickInput has been implemented, which seems to modify the functionality of tick input data. The dataset contains custom message data. The code defines a dictionary called CustomMessageData, which associates certain message types with their respective handler classes. This functionality seems to be used for the purpose of deserializing and altering message data.

In the context of photon peers and network setup, it is common practise to create two instances of Photon Peer. One instance is designated for the server-side, while the other is intended for the client-side. These instances are then linked together to establish a connection. A DeterministicNetwork instance is instantiated to manage the processing of Quantum Photon messages and their serialization/deserialization. The processing function is a fundamental component of a system that performs various operations on input data to produce desired output. It is responsible for the process function is clearly specified, indicating that it serves as the central logic of the proxy. The input for this process consists of network data, the identify of the target server, and a local port. The method uses PhotonPeer objects to handle incoming data and implements different logic depending on the type and content of the message.

The system undertakes encryption-related functions, manages several categories of communications such as operation requests, answers, and events, and seems to engage in tampering activities using the tampering.py module. The system monitors the state of tampering and records instances of tampering. The processed data is then sent to the designated the recipient. The script incorporates an event handling function called "on_event" that is activated in response to occurrences of events. The system manages various events pertaining to client connection, reloading of the tampering module, and other associated functionalities.

5.3 Testing

After the development of the tamper script, testing has been initiated with the mega proxy for two-way communication.

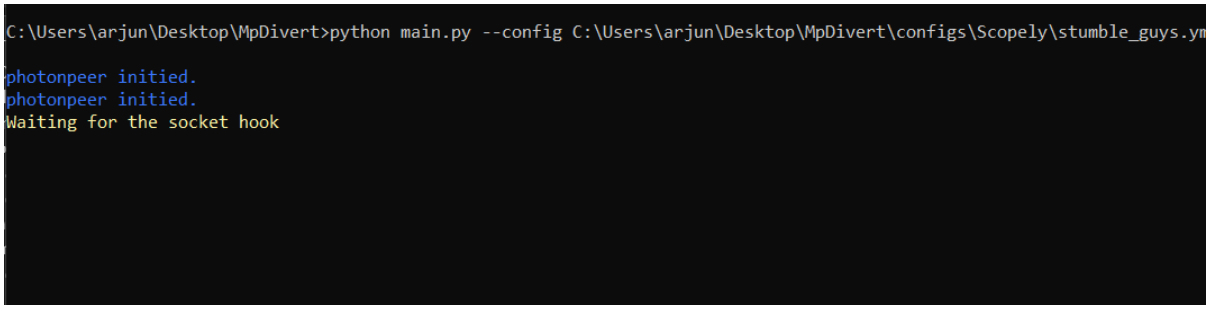

Figure 8 Testing the Mega-proxy.

6 Evaluation

This section involves findings of the study and the implementation of research in practical. Mega proxy and the game built on Enet protocol is being used for the evaluation of the thesis. After the development of the tamper script. The task was to initiate the two pair connection between server – Client and Client to Server. It was necessary to run the Command prompt as administrator.

6.1 Experiment with Mega proxy Interception of Game data

The Following test case involve the testing of the actual workflow of the mega proxy where once the game and mega proxy is started the game continuously send the data to mega proxy instead of the actual server.

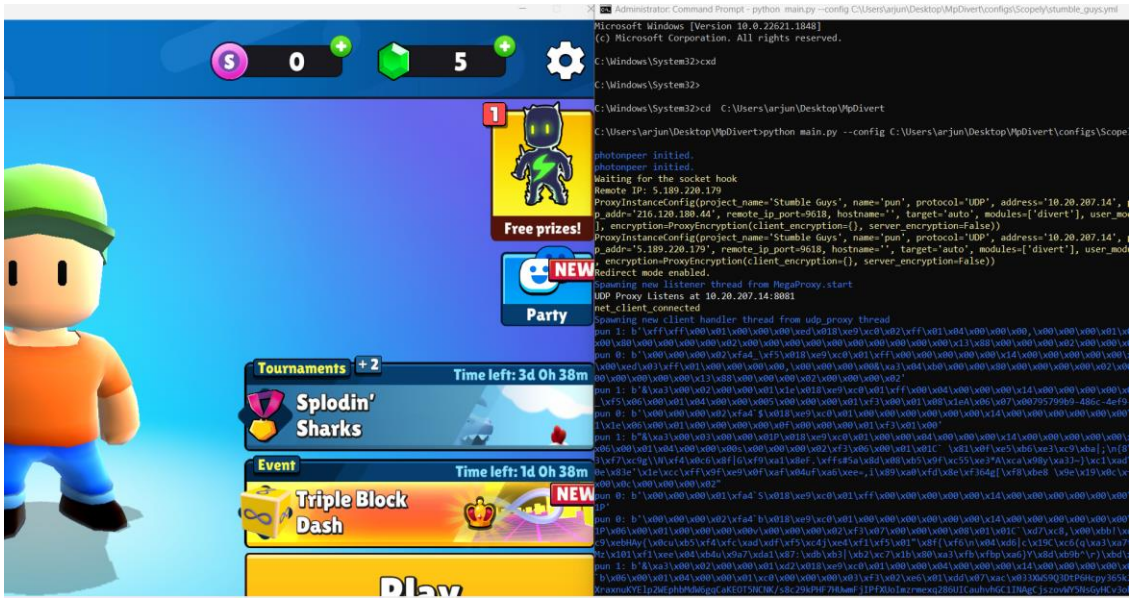

Figure 9 Interception of game data into megaproxy

The below image shows that it displays all the parameter of the players data such as ID and other encrypted values with the authentication value **MCWKESH**

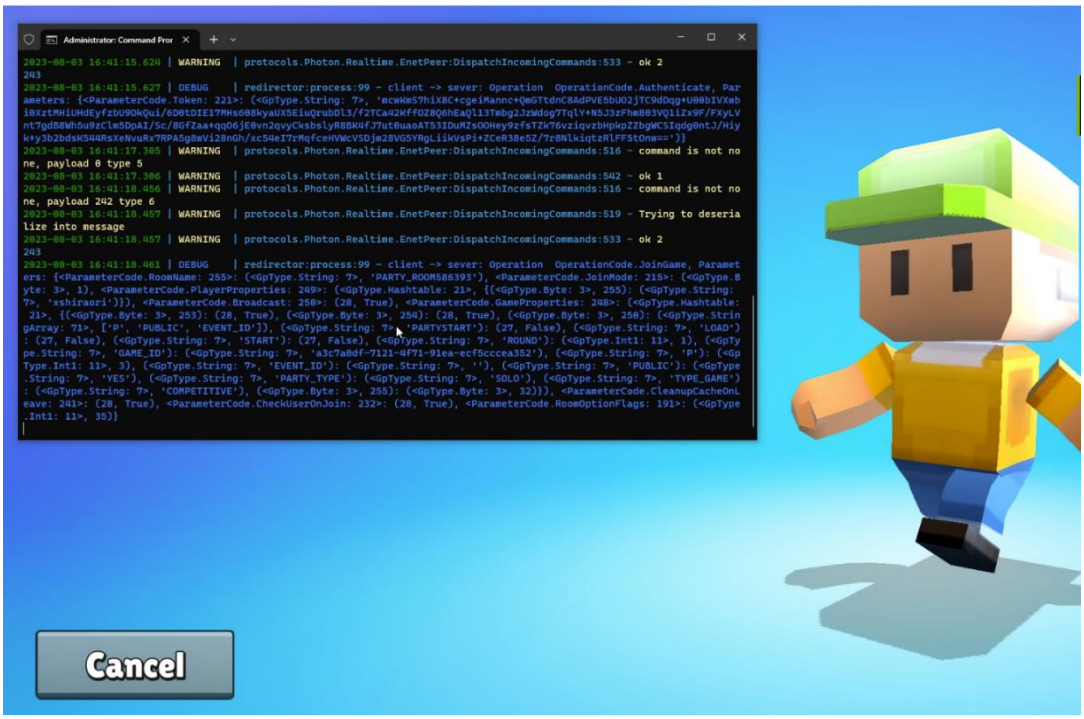

Figure 10 Displaying the game parameter

6.2 Experiment with Mega-proxy to Work on game tampering.

The function tamperTickInput looks to alter network data that is associated with a player's input. purpose IS to include modifying input data pertaining to direction and actions. The purpose of the function is presumably to perform operations on input data corresponding to certain player indices.

```python
def tamperTickInput(ti: DeterministicTickInput):
    if ti.PlayerIndex == 0:
        if ti.DataArray != bytearray(b'\x00\x00\x00'):
            logger.success(ti)
            if ti.DataArray != None:
                stream = BitStream(ti.DataArray)
                stream.ReadBool()
                dir = stream.ReadByte()
                new_stream = BitStream(size=3)
                new_stream.WriteBool(True)
                new_stream.WriteByte(dir)
                new_stream.WriteByte(250)

                ti.DataArray = new_stream.Data
                logger.debug(ti)
    if ti.PlayerIndex != 0:
        logger.success(ti)
        ti.DataArray = None
        ti.DataLength = 0
        logger.debug(ti)
    return ti
```

Figure 11 Tamper script snippet

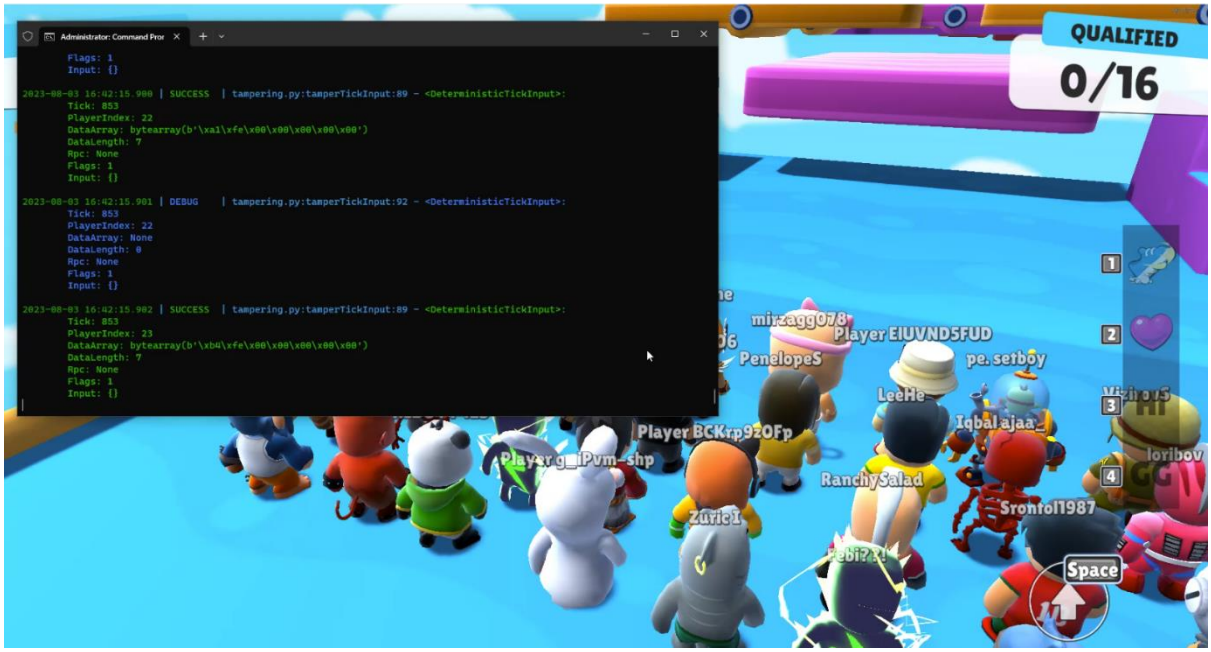

Figure 12 Player Position and index

6.3 Experiment with Mega Proxy to change name without soft currency

The following case study case involves of testing the actual gameplay with the penetration testing approach. The game doesn't allow the name change which is visible to other players as it require the soft currency i.e 100 gems in order to change the name for 2nd time as shown in the below screenshot.

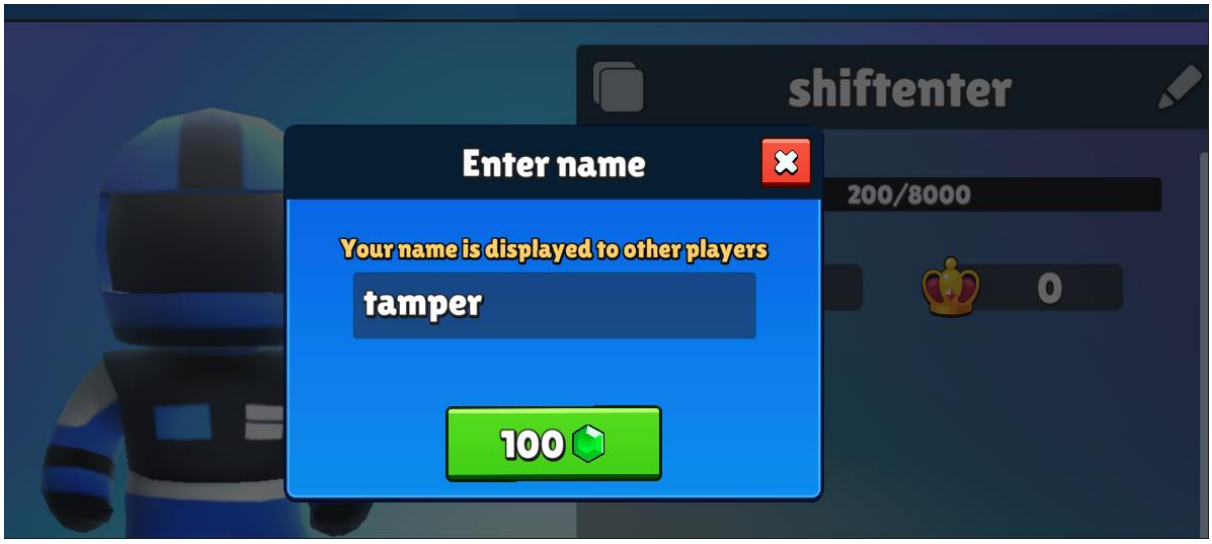

Figure 13 100 Gems require to change name

The below code snippet is responsible for tampering with the game data and make unauthorize changes into the game.

```
        continue
    elif isinstance(result[0], SetPlayerData):
        result[0].Data.Nickname = "test123"
        tampered = True
        logger.warning(result[0])

    else:
```

Figure 14 Tamper script for changing name

Here, the game data has been tampering with the script and the player name has been changed with the give name in the script and other player can view those name.

Figure 15 Name change without gems.

6.4 Discussion

The discussion delves into the implications of the research findings, highlighting the significance of the study in the context of online gaming security and the utilization of the ENet protocol. This section also explores the practical implications of the tamper script development, its potential impact on gaming environments, and the broader implications for the online gaming community. After developing the script, it was tested for a two-pair connection. During the second case, the game was launched and all packets were intercepted by the mega proxy, confirming that the script is functioning correctly and is ready to be tampered with. Whereas in the third case study represent the tampering with the game data such as displaying player position and unauthorized name changes. The penetration testing performed on the ENet protocol-based game disclosed the intricate relationship between performance optimisation and security flaws. While the ENet protocol offers commendable benefits in terms of low-latency data transmission, the study uncovered potential attack vectors that could compromise the gaming experience's integrity. This highlights the critical significance of undertaking thorough security assessments on networking protocols in order to identify and address potential vulnerabilities prior to their exploitation by malignant actors.

7 Conclusion and Future Work

The purpose of this study was to investigate and evaluate the efficacy of penetration testing techniques for ENet-based multiplayer online games. The research query revolves around finding the comprehensive method for locating the game's vulnerabilities. The objectives were to create and customize a tamper script within the context of a Mega Proxy tool, demonstrate unauthorized changes to the gameplay, and analyse the results critically. The study has effectively shown the capacity for manipulating game data via the use of the tamper script and Mega Proxy that were developed. The performed tests have provided insights into the potential vulnerabilities that may occur because of unauthorized tampering, namely in terms of manipulating player positions and altering names without the necessary authorization or money. The tamper script demonstrated its efficacy in intercepting and altering certain elements of the game, therefore bringing attention to possible security vulnerabilities. The results mentioned above are in accordance with the stated study aims and provide a valuable

contribution to the understanding of security concerns inside multiplayer online gaming settings. The limitations of the tamper script in effectively managing complex game events and a limited number of tests may impede a comprehensive evaluation of the impacts of tampering. Furthermore, the study focuses its attention on a particular protocol, namely ENet, and a certain genre of games, thereby limiting its ability to comprehensively address the wide range of security issues present in many gaming contexts.

There are several potential areas of investigation that might be pursued in order to contribute to the development of knowledge and understanding. One possible route for exploration is enhancing the tamper script to accommodate a wider range of game interactions, including complex player behaviors and interactions. The adoption of real-time tampering detection methods in collaboration with game creators has the potential to enhance overall game security. Furthermore, an in-depth investigation of the use of machine learning algorithms for the purpose of detecting and mitigating unauthorised tampering has the potential to provide novel approaches in addressing security apprehensions.

In addition, broadening the scope of the study to include a wide range of games using different protocols would provide a thorough evaluation of security obstacles in numerous gaming contexts.

References

Ban, Y., & Zhao, Y. (2012). The security research of massively multiplayer online role-playing games. *Proceedings - 2012 International Conference on Computer Science and Service System, CSSS 2012*, 1900–1903. https://doi.org/10.1109/CSSS.2012.473

Bono, S., Caselden, D., Landau, G., & Miller, C. (2009). Reducing the attack surface in massively multiplayer online role-playing games. *IEEE Security and Privacy*, *7*(3), 13–19. https://doi.org/10.1109/MSP.2009.75

Chang, H. B., Kwon, H. J., & Kang, J. G. (2010). The design and implementation of tamper resistance for mobile game services. *Mobile Information Systems*, *6*, 85–105. https://doi.org/10.3233/MIS-2010-0094

Chang, H., Park, J. H., & Kang, H. (2008). The security system design in online game for u entertainment. *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, 1529–1533. https://doi.org/10.1109/WAINA.2008.206

Chen, W., & Chen, M. (2002). *Internet Game Security*.

Chen, Y. C., Chen, P. S., Hwang, J. J., Korba, L., Song, R., & Yee, G. (2005). An analysis of online gaming crime characteristics. *Internet Research*, *15*(3), 246–261. https://doi.org/10.1108/10662240510602672/FULL/XML

Chen, Y. C., Hwang, J. J., Song, R., Yee, G., & Korba, L. (2005). Online gaming cheating and security issue. *International Conference on Information Technology: Coding and Computing, ITCC*, *1*, 518–523. https://doi.org/10.1109/ITCC.2005.215

Cho, C.-S., Sohn, K.-M., Park, C.-J., & Kang, J.-H. (2010). Online game testing using scenario-based control of massive virtual users. *2010 The 12th International Conference on Advanced Communication Technology (ICACT)*, *2*, 1676–1680.

Davis, S. B., & Price, W. J. (2008). Security issues for third party games: Technical, business and legal perspectives. *Computer Law & Security Review*, *24*(2), 163–168. https://doi.org/10.1016/J.CLSR.2008.01.004

Duh, H. B. L., & Chen, V. H. H. (2009). Cheating behaviors in online gaming. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5621 LNCS*, 567–573. https://doi.org/10.1007/978-3-642-02774-1_61

Efe, A., & Önal, E. (2020). ONLINE Game Security: A Case Study of an MMO Strategy Game. *GU J Sci, Part A*, *7*(2), 43–57. http://dergipark.gov.tr/gujsa

Goutam, A., & Tiwari, V. (2019). Vulnerability Assessment and Penetration Testing to Enhance the Security of Web Application. *2019 4th International Conference on Information Systems and Computer Networks, ISCON 2019*, 601–605. https://doi.org/10.1109/ISCON47742.2019.9036175

Hu, J., & Zambetta, F. (2008). Security issues in massive online games. *Security and Communication Networks*, *1*(1), 83–92. https://doi.org/10.1002/SEC.5

Kasim, S., Valliani, N., Ki Wong, N. K., Samadi, S., Watkins, L., & Rubin, A. (2022). Cybersecurity as a Tic-Tac-Toe Game Using Autonomous Forwards (Attacking) And Backwards (Defending) Penetration Testing in a Cyber Adversarial Artificial Intelligence System. *ICOSNIKOM 2022 - 2022 IEEE International Conference of Computer Science and Information Technology: Boundary Free: Preparing Indonesia for Metaverse Society*. https://doi.org/10.1109/ICOSNIKOM56551.2022.10034922

Ki, J., Cheon, J. H., Kang, J. U., & Kim, D. (2004). Taxonomy of online game security. *Electronic Library*, *22*(1), 65–73. https://doi.org/10.1108/02640470410520122/FULL/XML

Laurens, P., Paige, R. F., Brooke, P. J., & Chivers, H. (2007). A novel approach to the detection of cheating in multiplayer online games. *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, 97–106. https://doi.org/10.1109/ICECCS.2007.11

McGraw, G., & Chow, M. (2009). Guest editors' introduction: Securing online games: Safeguarding the future of software security. *IEEE Security and Privacy*, *7*(3), 11–12. https://doi.org/10.1109/MSP.2009.65

McGraw, G., & Hoglund, G. (2007a). Online games and security. *IEEE Security and Privacy*, *5*(5), 76–79. https://doi.org/10.1109/MSP.2007.116

McGraw, G., & Hoglund, G. (2007b). Online games and security. *IEEE Security and Privacy*, *5*(5), 76–79. https://doi.org/10.1109/MSP.2007.116

Mohr, S. (2011). IT SECURITY ISSUES. *International Journal of Computer Science & Information Technology (IJCSIT)*, *3*(5). https://doi.org/10.5121/ijcsit.2011.3501

Molloy, M. K. (1989). Comments on "A Note on the Performance of ENET II." *IEEE Journal on Selected Areas in Communications*, *7*(3), 427–430. https://doi.org/10.1109/49.16876

Treadway, B. (2018). *Online Gaming and Spam: Security Issues That Are Often Overlooked*.

Woo, J., & Kang Kim, H. (2012). *Survey and Research Direction on Online Game Security*.

Yan, J. (2003). Security design in online games. *Proceedings - Annual Computer Security Applications Conference, ACSAC*, *2003-January*, 286–295. https://doi.org/10.1109/CSAC.2003.1254333

Yan, J. J., & Choi, H. J. (2002). Security issues in online games. *Electronic Library*, *20*(2), 125–133. https://doi.org/10.1108/02640470210424455/FULL/PDF

Yan, J., & Randell, B. (2005). Security in Computer Games: from Pong to Online Poker. *Https://Eprints.Ncl.Ac.Uk*.

Yan, J., & Randell, B. (2009). An investigation of cheating in online games. *IEEE Security and Privacy*, *7*(3), 37–44. https://doi.org/10.1109/MSP.2009.60