

Uso de cheats en los videojuegos

Robert Mikhailovskiy

22 de septiembre de 2022

Índice

1. Introducción	4
2. Metodología	6
3. Clasificación	12
3.1. Bugs	12
3.2. Manipulación de HDD y memoria	12
3.3. Manipulación de paquetes	13
3.4. Macros y bots	13
3.5. Uso de cuentas secundarias	14
4. Cheat Engine	15
4.1. Introducción	15
4.2. Interfaz	15
4.3. Funcionamiento	16
4.4. Más usos	18
5. Accesorios gaming	19
6. Comandos de consola	25
6.1. Introducción	25
6.2. Comandos en el The Elder Scrolls V: Skyrim	25
7. Cheats en los juegos de un solo jugador	28
7.1. Introducción	28
7.2. Trainers	28
7.3. Editores de partidas guardadas	30
7.4. Edición de ficheros	32
8. Cheats en los juegos multijugador	36
8.1. Introducción	36
8.2. Tipos de cheats en los shooters	36
8.2.1. Aimbot	36
8.2.2. ESP	37
8.2.3. Triggerbot	42
8.2.4. BunnyHop	44
8.2.5. Control de retroceso	45
8.2.6. Efectos visuales	49
8.3. Tipos de cheats en los MOBA	50
8.3.1. Introducción	50
8.3.2. Tipos de cheats en los MOBA	50
8.3.3. Mejoras visuales	51
8.3.4. Uso automático de objetos	55
8.3.5. Auto-disable	55
8.3.6. Auto-save	56

8.3.7. Burro mensajero	56
8.3.8. Macros para héroes concretos	56
8.3.9. Overwolf	57
9. Resumen de cheats	59
10. Anticheats	61
10.1. Anticheats en el servidor	61
10.2. Anticheats en el cliente	62
11. Medidas adoptadas por las empresas	64
11.1. Introducción	64
11.2. Medidas restrictivas	64
11.3. Estado legal del uso de cheats	65
11.4. Estado legal del desarrollo y la venta de cheats	66

1. Introducción

Los videojuegos han aparecido en el siglo XX como consecuencia del desarrollo de las tecnologías digitales y en las tres últimas décadas se han convertido en un elemento muy común de entretenimiento, generalmente para personas jóvenes. Los videojuegos, igual que los juegos tradicionales, tienen unas reglas que tienen que cumplir los jugadores. También, igual que en muchos juegos tradicionales, los jugadores encuentran formas de esquivar esas reglas para obtener algún beneficio. Estos beneficios dependen del juego. Por ejemplo, en los juegos de un jugador es común eludir reglas para pasar niveles difíciles con mayor facilidad, mientras que en los juegos de muchos jugadores se usan programas para jugar con ventaja respecto a los rivales, por ejemplo poder ver información que no es visible para otros.

Originalmente, un cheat era una herramienta creada por los programadores para simplificar el proceso de desarrollo y el proceso de testeo de un juego. Los cheats permitían testear los elementos de juegos de una forma mucho más rápida que la tradicional. Por ejemplo, ejecutar un nivel determinado o añadir un objeto al personaje para testear su uso.

Bastantes veces los desarrolladores no eliminan la posibilidad de usar cheats (comandos de consola) en sus videojuegos. En la mayoría de los casos, cuando un juego no es multijugador y el uso de los comandos de consola no presenta un problema serio, se deja la opción de habilitar la consola de comandos a los jugadores para que estos puedan usar los comandos con sus finalidades. Por ejemplo, los jugadores podrían usar un cheat para disminuir la dificultad de un nivel si tienen problemas en pasarlo. Por otro lado, muchos videojuegos permiten construir modificaciones propias de los usuarios, ya sea cambiar algunos ajustes del juego, añadir nuevo contenido o corregir problemas que se han dejado los desarrolladores. En este caso, la consola de comandos es muy útil para que las personas que creen modificaciones también puedan hacer pruebas de su código eficazmente.

En algunos casos los juegos han salido a la venta con los comandos de consola no deshabilitados porque los desarrolladores se han olvidado de quitarlos antes de publicar la versión final. Uno de los ejemplos antiguos sería el juego Manic Miner, que salió en el año 1983 con un comando que permitía pasar cualquier nivel sin tener que pasar los anteriores [1]. Este caso hizo que los comandos de consola habilitados para el jugador se convirtieron en un elemento bastante común en los videojuegos.

Por otro lado, han surgido otros tipos de cheats. Estos tienen como objetivo principal dar ventajas no legales al jugador en un juego. A diferencia de los comandos de consola, son producto de personas o empresas de software externas a los desarrolladores de los juegos. Uno de los primeros cheats salió en el año 1981 para el juego Castle Wolfenstein.[2]. Se llamaba "The Great Escape Utility" y permitía cambiar modelos del juego, añadir objetos al personaje principal o cambiar de nivel. Con el desarrollo de los juegos multijugador, ha incrementa-

do muchísimo el uso de estos cheats. Se han hecho parte de la cultura de los videojuegos y casi se ha creado una industria orientada a la producción de los cheats.

Estos cheats se pueden distribuir de varias formas, según el deseo de su creador. La mayoría de ellos son pagados. Algunos pocos cheats son públicos (gratuitos). Estos, en la mayoría de los casos, contienen muy pocas funcionalidades comparadas con los pagados y suelen contener una versión de pago en la cual se encuentra el resto de las funcionalidades. Otra causa por la cual un cheat puede ser gratuito es que haya sido robado o simplemente se ha dejado de mantener. El tipo de cheat menos común son los cheats privados. Estos, además de ser pagados y caros, solamente se distribuyen en comunidades reducidas, con dos finalidades: por un lado, se limita la posibilidad de que los desarrolladores de un juego puedan detectar el cheat y aplicar las medidas correspondientes y, por otro lado, simplemente no se quiere compartir con gente desconocida para no dejarles tener las mismas ventajas que tienen dentro de su comunidad.

Aparte de los cheats como programas, hay otras formas de conseguir ventajas, como por ejemplo usar una cuenta con rango bajo u obtener beneficio a partir de errores en el código del juego.

Con el aumento del uso de las trampas, las empresas de desarrollo de videojuegos han sido obligadas a prestar mucha atención a esta situación y han desarrollado varias formas de controlarla. Muchas empresas prohíben el uso de trampas en sus juegos multijugador y aplican distintas formas de limitar el acceso al juego a personas si detectan que han usado trampas.

Este manual es una recopilación de las diferentes técnicas aplicadas para jugar con ventajas en los videojuegos. Se pretende clasificar los cheats según su modo de interacción con el juego, evaluar varios casos de ejemplo, explicar las técnicas anticheats existentes y mostrar las medidas que adoptan las empresas de videojuegos para poder limitar el uso de los cheats.

2. Metodología

En esta sección se explica la estructura del manual y se enseña el proceso de la búsqueda de información y las fuentes usadas. La mayor parte de las búsquedas se han hecho en el idioma inglés, porque la mayoría del contenido relacionado con el manual se encuentra en este idioma.

- **Clasificación.** Esta sección contiene información acerca de las distintas formas de cheats, explicando la interacción entre el cheat y el juego (o entre el cheater y el juego, en caso de que el cheat no sea un programa externo). En total se explican 5 tipos distintos y se enseñan varios ejemplos. La fuente principal de este capítulo ha sido el trabajo de fin de máster de Samuli Heltonen llamado "Comparative Study of Anti-cheat Methods in Video Games".[3] Este trabajo tiene como objetivo principal explicar los métodos anticheat que hay presentes en los juegos, pero también tiene un capítulo referente a los tipos de cheats, que se ha tomado como base, pero se ha cambiado un poco la clasificación. No se ha usado el capítulo de "Spoofing", debido a que no se ha considerado un cheat, se ha cambiado bastante el contenido del capítulo "Exchanging real money for in-game goods and services" generando así el capítulo de "Uso de cuentas secundarias", y se ha añadido toda la parte del uso de los ficheros del disco duro de los juegos que no está presente en el trabajo de fin de máster. También se ha añadido más información y se han añadido ejemplos que no se encuentran en el trabajo original.

Para explicar los bugs, se ha buscado información sobre un bug conocido de duplicación de objetos en Dying Light usando las palabras clave "dying light dupe bug" y se ha buscado si es legal usar bugs en el PUBG: BATTLEGROUNDS con las palabras "is bugs usage in pubg allowed".[4][5] El contenido del capítulo de cuentas secundarias en los juegos se basa en la situación de los juegos como el Dota 2 o el Counter-Strike: Global Offensive, donde es común usar más de una cuenta.

- **Cheat Engine.** Esta sección consiste en explicar el uso del programa Cheat Engine y su relación con el método de manipulación de la memoria del juego.[6] Primero de todo se enseña la interfaz del programa y se explican sus elementos. Luego, se explica paso a paso cómo encontrar una posición de memoria y cómo cambiar su valor. En la parte final, se enseñan más funcionalidades que tiene el programa, como la creación de un trainer o la posibilidad de compartir las direcciones de memoria encontradas. Se enseña un repositorio que contiene direcciones de memoria para un juego que se usa en uno de los capítulos de este manual.[7] El trainer (tabla de Cheat Engine) se ha buscado en internet usando las palabras clave "cheat engine game tables".
- **Accesorios gaming.** En esta sección se explica qué son los accesorios gaming y cuáles son sus diferencias comparadas con los accesorios de oficina. En este capítulo también se enseña el programa Logitech Gaming

Software, un software para los accesorios gaming de la empresa Logitech. Se explican algunas funcionalidades más importantes de este programa, resumiendo las capacidades adicionales que les proponen las empresas a sus accesorios de la gama gaming. Algunos ejemplos de estas funcionalidades son el cambio de la sensibilidad y la programación de botones.

- **Comandos de consola.** En esta sección se explica qué son los comandos de consola. Como se explica en la introducción, aunque no sean un método prohibido, son el inicio de los cheats y, por lo tanto, también presentan interés. Primero de todo se explica cuál es el objetivo de los comandos de consola y más adelante se explica como funcionan los comandos de consola en el juego de The Elder Scrolls V: SKyrim. El uso de este juego viene justificado por la gran cantidad y capacidad de los comandos de consola que permiten controlar todas las entidades del juego. Para explicar todos estos capítulos se ha usado una página relacionada con los comandos de consola de la Wiki de la serie The Elder Scrolls (The Elder Scrolls Wiki).[8]
- **Cheats en los juegos de un solo jugador.** Esta sección contiene varios ejemplos de métodos de cheats que se usan en los juegos de un solo jugador. Estos juegos mayoritariamente son juegos offline, ya que no necesitan conectarse al servidor y guardan todo su contenido en el ordenador del jugador.

El primer cheat descrito se llama trainer. Se enseñan sus funcionalidades y se explica cómo funciona internamente (usando la memoria del juego). Esta información se ha encontrado en la página "Trainer (games)" de Wikipedia. Este capítulo contiene un ejemplo de trainer para el juego de Fallout 4 que enseña un ejemplo concreto de funcionalidades para un juego.[9] Para encontrarlo se ha hecho una búsqueda en Google usando las palabras clave "fallout 4 trainer".

El segundo cheat descrito es el editor de partidas guardadas. Aquí se explica como se guarda una partida de un juego en el ordenador y se explica el funcionamiento de un programa que permite editar el contenido de esta partida. Se usa como ejemplo la serie de juegos Mass Effect, porque es un ejemplo muy clásico y extenso que no solamente permite cambiar datos y elecciones del jugador en el juego actual, sino también en los juegos predecesores de la serie.[10] Para encontrar un programa editor de partidas guardadas, se ha buscado en Google con las palabras clave "mass effect save editor". Se enseñan las funcionalidades del editor encontrado y se explica cómo cambiar los valores de una partida.

El último cheat descrito en este capítulo consiste en buscar y editar información en los ficheros del juego, que se encuentran en la carpeta del juego, en el disco duro. El juego que se usa como base se llama S.T.A.L.K.E.R. Call of Pripyat. Primero de todo, se explica la estructura del directorio de este juego, enseñando donde se guardan los datos y cómo descomprimirlos con un programa, que se ha encontrado en Google con las palabras clave

”stalker data unpacker”. El siguiente paso es enseñar las carpetas donde se guardan los distintos tipos de datos, como las texturas o las características. Se muestra un ejemplo de un fichero de características, que contiene las variables del personaje principal y otro ejemplo que contiene las características de un arma. Después, se muestra un ejemplo de un fichero que contiene código fuente del juego.

- **Cheats en los juegos multijugador.** Esta sección esta dedicada a los juegos de multijugador. Es el capítulo más extenso, ya que contiene más ejemplos que los otros dos capítulos y estos ejemplos aparecen más desarrollados. Contiene dos partes principales: en la primera parte se explican los ejemplos de cheats en los juegos de disparo y en la segunda parte se explican los cheats en los juegos de género MOBA. La causa de tener dos partes es que estos dos géneros de juegos son los géneros que más cheats tienen.

El capítulo de shooters tiene en total 5 tipos de cheats. El primero de todos es aimbot, el más clásico. La mayoría de la información sobre este cheat, como sus ajustes y tipos, se ha encontrado en una publicación de la página de Habr, un foro en el idioma ruso orientado a la tecnología.[11] Se ha usado también la página de Hackerbot (un portal sobre cheats) para añadir más información.[12] Luego hay una descripción de cómo un aimbot calcula las coordenadas para apuntar a un enemigo.[13] Esta información técnica se ha encontrado en una publicación del foro Lolz, un foro dedicado a los videojuegos y a la ingeniería inversa. La publicación, además de explicar cómo se calculan las coordenadas, tiene un código fuente escrito en C#, pero no está incluido en este manual.

El siguiente tipo de cheat (ESP) se ha explicado usando la misma publicación de Habr.[11] Se basa en el juego de PUBG: BATTLEGROUNDS para explicar qué es un radar (un tipo de ESP). Más adelante se explica qué es un wall-hack (versión avanzada de ESP) y se muestran varias capturas de pantalla de los juegos PUBG: BATTLEGROUNDS y Counter-Strike: Global Offensive con el cheat activado.[14] Después de mostrar estas capturas, se enseña el código de un wall-hack simple para el Counter-Strike: Global Offensive.[15] Este código fuente se ha encontrado en un repositorio público de Github mediante la búsqueda con las palabras clave ”csgo wallhack source code”. Su autor se llama Snaacky. Más abajo se muestra una captura de Counter-Strike: Global Offensive con el código ejecutado.

El tercer tipo de cheat explicado se llama triggerbot. Su explicación y sus posibles ajustes también se basan en la misma publicación de Habr que en los dos casos anteriores.[11] Más adelante se enseña un código fuente encontrado en otro repositorio de Github del mismo autor que en el caso del ESP.[16] Para hacerlo se han mirado los distintos repositorios de Snaacky y se ha encontrado un código fuente de triggerbot para el mismo juego (Counter-Strike: Global Offensive).

Después, se explica el tipo de cheat relacionado con la velocidad del movimiento del jugador (bunnyhop). Una parte de la información sobre este cheat se ha encontrado en la publicación de Habr y otra parte se ha encontrado en Wikipedia buscando la palabra "bunnyhop" en el buscador de Google.[11][17] Más abajo se enseñan dos códigos fuentes de bunnyhop, pero funcionan de modo diferente. El primero es un script en AHK encontrado en el foro de YouGame, otro foro con la misma temática que Lolz.[18] El segundo ejemplo es un código en un repositorio de Snaacky. Se explica como funciona cada uno y la diferencia entre ambos.[19]

El cuarto tipo de cheat en los juegos de disparo permite controlar el retroceso de un arma. En primer lugar, se explican dos ejemplos de cómo funciona el retroceso en los shooters. El primer ejemplo es el juego de PUBG: BATTLEGROUNDS, y para explicar los patrones de retroceso, se usa la captura de un video explicativo de WackyJacky101, un blogger que publica vídeos sobre este juego.[20] El segundo ejemplo es el caso del juego Counter-Strike: Global Offensive, donde se enseña un ejemplo de retroceso. Para encontrarlo se ha hecho una búsqueda con las palabras clave "csgo ak47 recoil pattern". Después de los ejemplos, se enseña un ejemplo de script en LUA para controlar el retroceso para los ratones gaming de la compañía de Logitech, un tipo de cheat externo al juego. Luego, se enseña otro script externo, esta vez en AHK, para imitar el patrón de retroceso en el juego de Counter-Strike: Global Offensive.[21] Este ejemplo ha sido encontrado en internet buscando por palabras clave "csgo no recoil ahk script".

El último tipo de cheat en los shooters permite desactivar ciertos efectos visuales. Se explica qué efectos visuales se puede desactivar y luego se muestra y se explica un código, otra vez de Snaacky, que permite desactivar el efecto de ceguera en el juego de Counter-Strike: Global Offensive.[22]

Ahora viene el capítulo de cheats en los juegos MOBA. Antes de nada, se explica qué es un MOBA, ya que este género de juegos no es tan conocido como los juegos de disparo. Después, se explican los tipos de cheats que hay para los juegos de este tipo, basándose en el juego Dota 2 y el cheat llamado Octarine, que contiene una gran variedad de funcionalidades. En total hay tres categorías (mejoras visuales, macros y Overwolf).

En la parte de mejoras visuales, antes de nada, se enseña cómo es la interfaz del juego sin cheats. Después, se enseña cada mejora que puede proporcionar un cheat. Se usan varias capturas de pantalla para enseñar cómo son las mejoras visuales en el juego.

En la parte de macros, se explican las situaciones más comunes de uso automático de objetos o habilidades. Finalmente, se enseñan dos casos con personajes concretos, donde los macros toman un control elevado del personaje.

Finalmente, se enseña una implementación de Overwolf, un cheat especial

para el Dota 2 que permite ver estadísticas de los jugadores que han en la partida y se explica qué estadísticas se puede ver.

- **Resumen de cheats.** La parte de la clasificación y los ejemplos de cheats se acaba con esta sección, donde se encuentra un mapa conceptual que permite ver de forma fácil a qué tipo de cheat pertenecen varios ejemplos explicados en los capítulos anteriores.
- **Anticheats.** Esta sección contiene la información relacionada con los métodos que permiten detectar el uso de cheats en un juego. Este capítulo es un resumen del trabajo de fin de máster de Samuli Lehtonen.[3] Se divide en dos partes: anticheats en el servidor y anticheats en el ordenador del cliente y viene acompañado por algunos ejemplos, como es el caso del sistema Overwatch en el Dota 2.[23]
- **Medidas adoptadas por las empresas.** En esta sección se explican las formas con las que las empresas creadoras de videojuegos combaten a los cheats y a los cheaters. Para redactar este capítulo se ha buscado mucha información en internet. En primer lugar, se ha buscado la definición del Acuerdo de Licencia del Usuario Final para explicar su objetivo y su relación con los juegos.[24] Luego se explica una serie de medidas restrictivas que imponen las empresas de videojuegos en sus productos. Para hacerlo se ha usado la página "Cheating in video games" de Wikipedia y luego se han buscado casos particulares de algunos juegos, como el PUBG: BATTLEGROUNDS o los juegos de Valve para explicar las peculiaridades de sus medidas restrictivas.[2][25][26] Se explican métodos como el bloqueo temporal y permanente, la anulación del progreso del personaje (una medida del juego Grand Theft Auto V), el bloqueo por el identificador del hardware, etc.

Después de las medidas restrictivas, se explica el estado legal de uso de los cheats y el estado legal del desarrollo y la distribución de los mismos. Para explicar ambas partes, se han realizado varias búsquedas por internet. Por ejemplo, se han buscado frases como "legality of cheating in videogames", o "legality of developing cheats for videogames". Se han resumido varias fuentes y se han buscado varios casos de violación del derecho del autor y casos de detención policial de cheaters y desarrolladores de cheats, como es el caso de un jugador de Corea del Sur multado por haber usado cheats en un juego o el caso de 141 cheaters detenidos en China.[27][28] Esta información se ha encontrado en varios periódicos digitales y páginas relacionadas con los videojuegos, como por ejemplo Kotaku.[29]

Como resumen, las principales fuentes de recursos han sido:

- "Comparative Study of Anti-cheat Methods in Video Games". Trabajo de fin de máster relacionado con los anticheats.
- Habr. Foro relacionado con la tecnología.
- YouGame y Lolz. Foros relacionados con los cheats y la ingeniería inversa.

- Wikipedia. Información general.
- Hackerbot. Página relacionada con los cheats.
- Github. Repositorio de código.
- Páginas de noticias relacionadas con los videojuegos.

3. Clasificación

3.1. Bugs

Un bug es simplemente un error de programación que puede provocar algún comportamiento no deseado en el juego. El proceso de depurar aplicaciones tan complicadas como los videojuegos es muy costoso y debido a esto es muy probable que al publicar un juego haya bastantes bugs o fallos de programación en este. Por lo tanto, el proceso de detectar y arreglar bugs no para al sacar un juego a la venta, sino que sigue hasta que los desarrolladores del juego dejen de soportar el juego.

Los jugadores puede detectar algunos bugs y usarlos para obtener algún beneficio. Por ejemplo, en el juego Dying Light se puede duplicar un objeto manipulando el menú del inventario del jugador.[4] Los desarrolladores suelen corregir los bugs detectados por los jugadores, mientras que estos buscan más y más vulnerabilidades en el juego.

Normalmente, las empresas de videojuegos no clasifican los bugs como "cheats" y, por lo tanto, no castigan a los jugadores. Sin embargo, en algunos juegos multijugador sí que aplican restricciones a los jugadores que usan bugs, como es el caso de PUBG: BATTLEGROUNDS. [5]

3.2. Manipulación de HDD y memoria

Los juegos guardan sus datos y configuraciones en el ordenador del jugador. En el caso de los juegos offline, allí se guardan todos los datos, mientras que en los juegos online solamente se guarda una porción de datos en el ordenador del jugador, y la otra porción se guarda en el servidor. La información de la sesión actual se guarda en la memoria RAM, mientras que los datos generales se guardan en el disco duro.

No toda la información que se encuentra en el ordenador tiene que ser vista por el jugador. Por ejemplo, el jugador no tendría que saber la posición exacta de sus enemigos sin poder verlos físicamente, ni tampoco tendría que saber todas las características de su arma, solo necesita unas cuantas. Por ejemplo, no tendría que saber la fórmula para calcular el retroceso de un arma. Para poder leer esta información, existen varios programas.

En el caso del disco duro, los datos suelen estar comprimidos en varios archivos. Estos archivos pueden contener algoritmos de escenas, características de objetos, modelos, texturas, audios y más ficheros. Un jugador podría descomprimir estos archivos y editarlos, por ejemplo cambiar la apariencia de una textura para que sea más simple o cambiar el modelo de un enemigo para que sea más grande y sea más fácil apuntar. Esto se usa mucho para crear modificaciones de usuario para los juegos que lo permiten y también para crear editores de partidas guardadas.

En el caso de la memoria RAM, la información no aparece estructurada ni comprimida. Esta información contiene la información de la partida actual, como

por ejemplo las posiciones de los jugadores, su equipaje, sus características, etc. Existen programas que permiten leer la memoria de un proceso y mostrarla al usuario en formato hexadecimal. Aparte de visualizar la memoria, estos programas tienen más herramientas que permiten detectar valores deseados en la memoria y cambiarlos. Estos programas sirven de apoyo para buscar valores en la memoria que luego se usan en la mayoría de los cheats, por ejemplo los trainers. El ejemplo más clásico de un programa para leer la memoria de un juego es Cheat Engine.[6].

3.3. Manipulación de paquetes

Este tipo de cheats, por razones obvias, solamente existe en los juegos online. Un juego online presenta la estructura de una aplicación distribuida, donde en una partida hay un servidor y unos clientes (jugadores). Durante el juego se hace un constante intercambio de paquetes entre el servidor y los clientes, para que el juego pueda ser interactivo en tiempo real.

No toda la información del juego se encuentra en el ordenador del cliente y no toda la información que le manda el servidor tiene que ser fácilmente accesible al jugador. El servidor puede mandarle al jugador más información de la necesaria para aumentar la cantidad de operaciones que se hace en el cliente, optimizando de este modo el rendimiento del servidor.

En primer lugar, se puede interceptar los paquetes que le vienen al cliente y estudiarlos, para sacar información oculta. Para esto se puede usar el programa WireShark, un analizador de protocolos que permite ver la información de todos los paquetes que envía y recibe un ordenador. Al mostrar la información en formato hexadecimal, se trata de hacer una búsqueda de valores que podrían presentar interés para el jugador.

En segundo lugar, se puede estudiar qué información es enviada al servidor por parte del cliente, otra vez usando WireShark. Después de estudiar esta información, se puede editar la información del paquete antes de enviarlo al servidor. Por ejemplo, en el caso, cuando el jugador ataca a un enemigo y le provoca 40 puntos de daño, se puede encontrar y cambiar este número en el paquete para que el servidor piense que el jugador ha hecho más daño al enemigo.

3.4. Macros y bots

Un tipo común de cheats en los juegos son acciones o combinaciones de acciones hechas por un programa externo en vez del jugador. El objetivo principal es simplificar el juego para liberar al jugador de algunas necesidades delegándolas a un programa.

Un macro es una serie de acciones que ejecuta un programa en un caso determinado, por ejemplo si el jugador presiona un botón. Estos macros pueden funcionar de dos formas: los más simples solamente simulan pulsaciones de teclado o ratón, sin interactuar directamente con el juego; otros pueden editar la

memoria del juego o editar los paquetes para simular acciones. Normalmente, los macros sirven para ejecutar varias acciones rápidamente en juegos donde hay muchos controles.

Un bot es un programa que puede jugar solo a un juego. Consiste en un agente inteligente que percibe la información del entorno del juego y puede decidir que acción hacer basándose en su base de conocimiento. Los bots suelen usarse en juegos donde se trata de hacer acciones repetidas durante mucho tiempo para alcanzar algún objetivo. Por ejemplo, existen muchos bots para los juegos de la serie World Of Warcraft, donde el objetivo de los bots es matar enemigos en lugares determinados repetidamente para subir de nivel y ganar dinero. Esto presenta una desventaja para los jugadores reales debido a que un bot puede estar haciendo lo mismo durante todo el día, mientras que un jugador real tiene mucho menos tiempo para jugar y así nunca podrá superar al bot.

3.5. Uso de cuentas secundarias

El uso de cuentas secundarias no siempre tiene como objetivo ganar algún beneficio. En algunos juegos un jugador puede crear varias cuentas o personajes para así tener una experiencia de juego distinta en cada caso. Sin embargo, en los juegos competitivos las cuentas secundarias sí que son un problema. El sistema de rangos en los juegos competitivos existe para que un jugador pueda jugar con otros jugadores de su nivel, equilibrando de esta forma la partida. Si un jugador usa una cuenta con un rango que no le corresponde, provocará problemas en la partida debido a que sus compañeros no tendrán la misma experiencia.

Otro problema relacionado es la compra-venta de cuentas en los juegos competitivos. Existen mercados donde se venden cuentas con juegos competitivos de cualquier rango, con lo cual el sistema de rangos deja de funcionar tal como estaba planeado, ya que cualquier persona puede comprar una cuenta de cualquier rango sin restricciones y generar caos en las partidas.

En muchos juegos competitivos el uso de cuentas secundarias está prohibido y existen modos de detectar si una persona juega desde otra cuenta.

4. Cheat Engine

4.1. Introducción

Cheat Engine es un programa libre que permite escanear y editar la memoria de un proceso. Su objetivo principal es poder encontrar regiones en la memoria de un juego que pueden tener algún valor de interés y poder cambiarlos. Por ejemplo, de esta forma se puede editar la cantidad de recursos del jugador en un juego de estrategia. También tiene la posibilidad de guardar estas direcciones y programar botones para poder asignarles valores concretos, con lo cual se puede hacer un trainer simple. No se puede usar Cheat Engine en los juegos online, debido a que los datos se verifican con el servidor y no es posible cambiarlos.

4.2. Interfaz

En la Figura 1 se puede observar la interfaz de la pantalla principal de Cheat Engine. En la parte superior hay un botón para escanear la memoria de un proceso. En la parte central izquierda hay una pantalla que enseña todas las apariencias de un valor en la memoria del proceso seleccionado. En la parte central derecha se encuentra el botón para buscar un valor en la memoria y varios ajustes, como podría ser pausar el juego durante el escaneo. En la parte inferior se encuentran las posiciones de memoria guardadas del proceso seleccionado.

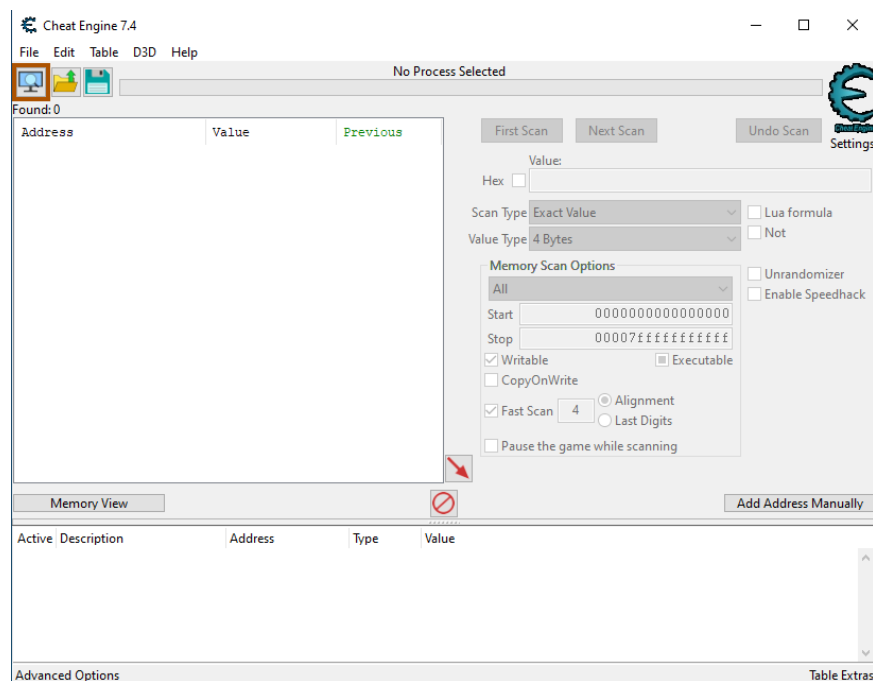


Figura 1: Pantalla principal de Cheat Engine

4.3. Funcionamiento

Para poder escanear la memoria del juego, hay que tenerlo ejecutado. Después, hay que hacer clic en el botón izquierdo de la barra superior para seleccionar el proceso del juego. Para hacerlo más fácil se puede elegir solo aplicaciones (primera pestaña).

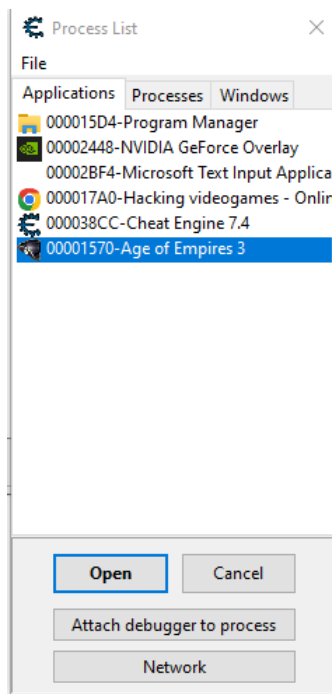


Figura 2: Selección del proceso

Ahora se puede ver que está habilitada la parte de la interfaz derecha y se puede empezar a buscar valores que se podría cambiar. Por ejemplo, en el juego hay un recurso cuya cantidad ahora mismo es 7. Para encontrar este valor hay que escribir este valor en el campo "Value" y presionar el botón New Scan. En la parte de la izquierda se enseñará una lista de todas las posiciones de memoria del proceso que contienen este valor.

No todas las posiciones encontradas corresponden con el valor que se pretende encontrar y por esto hay que volver a hacer una búsqueda con otro valor. Para hacer esto, primero hay que cambiar este valor en el juego de alguna forma (por ejemplo, gastando el recurso). Después, hay que introducir el nuevo valor en el mismo campo de antes y presionar el botón "Next Scan", para que la búsqueda solamente se haga dentro de las posiciones de memoria previamente guardadas. De esta forma, se descartan las posiciones que no tienen que ver con la variable de interés. Se puede repetir este proceso hasta que solo quede una posición o hasta que todas las posiciones encontradas cambien juntas de valor.

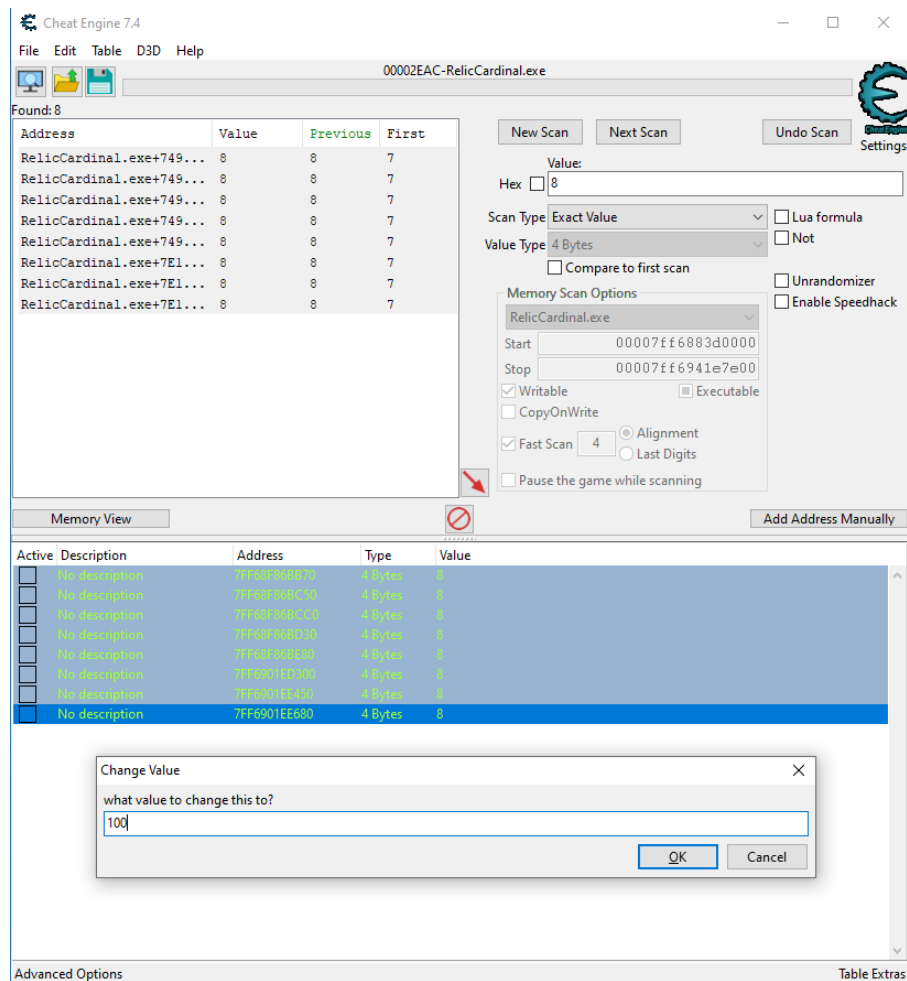


Figura 3: Cambio de valor

Al averiguar cuál es la posición correcta, se puede editarla. Para hacerlo hay que seleccionar la posición (o seleccionarlas todas si hay varias) y presionar el botón de la flecha roja para que la parte seleccionada aparezca en la lista inferior. En esta lista hay que volver a seleccionar todas las posiciones a cambiar y hacer doble clic en la columna "Value" (Figura 3). Sale una pequeña ventana en la cual hay que introducir el nuevo valor y presionar el botón Ok.

Ahora se puede abrir el juego y ver si el valor se ha cambiado. En algunos casos, para notar el cambio hay que hacer alguna manipulación con el valor antiguo (volver a gastar el recurso, por ejemplo). Si no se ha producido un cambio, significa que las posiciones de memoria cambiadas no son responsables del valor y hay que volver a hacer una búsqueda.

4.4. Más usos

Los juegos pueden tener la asignación de memoria estática o dinámica. Si la asignación es variable, hay que hacer la búsqueda de direcciones cada vez que se inicia el juego. Si la asignación es estática, entonces basta con haber encontrado una vez las direcciones, porque no van a cambiar.

En los juegos de asignación estática de memoria, la asignación es la misma para cualquier ordenador. Esto significa que una persona puede compartir sus direcciones con otras personas. Un ejemplo muy conocido es el caso de HazeDumper, un repositorio que contiene todas las direcciones de memoria de posible interés para el juego de Counter-Strike: Global Offensive.[7] Las direcciones se encuentran en un fichero JSON. La información de este fichero puede ser descargada usando Javascript o cualquier otro lenguaje que permite trabajar con JSON, como por ejemplo Python. A partir de estas direcciones de memoria se puede hacer un programa que pueda leer el proceso del juego y manipular las direcciones, ya sea para sacar información o cambiar valores. En el caso de Python, la librería "pymem" permite trabajar con los procesos.

Cheat Engine permite asignar nombres propios a las direcciones de memoria encontradas, hacer algoritmos propios para editar estos valores y guardar todo en un fichero. De esta forma se puede crear un cheat completo y compartirlo con otros usuarios. Para poder usar una tabla descargada de internet, hay que abrir esta tabla con Cheat Engine y ejecutar el juego correspondiente. Luego, se podrá activar o desactivar cualquiera de las opciones de la tabla de direcciones.

<input checked="" type="checkbox"/>	Compact UI (Numpad Ctrl + Numpad 1)			<script>
<input checked="" type="checkbox"/>	Infinite Deployable Equipment			<script>
<input checked="" type="checkbox"/>	Player Health			
<input checked="" type="checkbox"/>	Health	P-> 1A1A048C070	4 Bytes	100
<input type="checkbox"/>	Max Max Health	P-> 1A1A048C074	4 Bytes	150000
<input checked="" type="checkbox"/>	Invincibility Mode	P-> 1A1A048C099	Byte	On
<input checked="" type="checkbox"/>	Player Weapon Wielder (Shoot to populate)			<script>
<input checked="" type="checkbox"/>	Current Ammo	P-> 1A1A099ABE8	4 Bytes	200
<input type="checkbox"/>	Max Ammo	P-> 1A1A099ABEC	4 Bytes	200
<input checked="" type="checkbox"/>	Ammo Miracle	P-> 1A1A099ABF0	Byte	On
<input checked="" type="checkbox"/>	Save Data (Collect coins to populate)			<script>
<input type="checkbox"/>	Lives	P-> 1A0A59FB910	4 Bytes	1
<input type="checkbox"/>	Coin Count	P-> 1A0A59FB914	4 Bytes	10022
<input type="checkbox"/>	Blessing Count	P-> 1A0A59FB918	4 Bytes	102
<input type="checkbox"/>	Experience Level	P-> 1A0A59FB91C	4 Bytes	4
<input checked="" type="checkbox"/>	Player Data			
<input type="checkbox"/>	Health	P-> 1A1C198C318	4 Bytes	100
<input type="checkbox"/>	Equipped Left Hand Weapon	P-> 1A1C198C31C	4 Bytes	4
<input type="checkbox"/>	Equipped Right Hand Weapon	P-> 1A1C198C320	4 Bytes	200
<input type="checkbox"/>	Ammo Count	P-> 1A1C198C324	4 Bytes	200
<input type="checkbox"/>	Xp Count	P-> 1A1C198C328	4 Bytes	6
<input type="checkbox"/>	Orison Level	P-> 1A1C198C32C	4 Bytes	100
<input type="checkbox"/>	Extra Hp	P-> 1A1C198C330	4 Bytes	0

Figura 4: Tabla de Cheat Engine para Saints Row

5. Accesorios gaming

En la última decena de años, con el aumento masivo de los videojuegos en la vida cotidiana, han surgido muchos accesorios de ordenador orientados a los jugadores. La diferencia que tienen con los accesorios de oficina es la cantidad de opciones que ofrecen y la posibilidad de programar acciones.



Figura 5: Ratones gaming

Como se puede ver en la Figura 5, los ratones gaming tienen un diseño que los diferencia mucho de los ratones de oficina. Su forma es distinta, tienen más botones y algunos tienen retroiluminación. Pero estas diferencias no son las únicas: estos ratones tienen la sensibilidad (DPI) muy alta y ajustable, un tiempo de espera menor que los ratones de oficina y un número de clics garantizado mucho mayor.

Algunas empresas que venden accesorios para juegos, desarrollan software específico para sus productos, con el cual aumentan los ajustes del accesorio para adaptarse mejor a las necesidades del jugador. Un ejemplo es el software Logitech Gaming Software de la empresa Logitech.



Figura 6: Logitech Gaming Software

En la Figura 6 se ve la ventana principal del programa. En la parte de abajo hay 5 opciones:

- Inicio
- Ajuste de los botones
- Ajuste de la sensibilidad (DPI)
- Ajuste de iluminación
- Análisis del uso

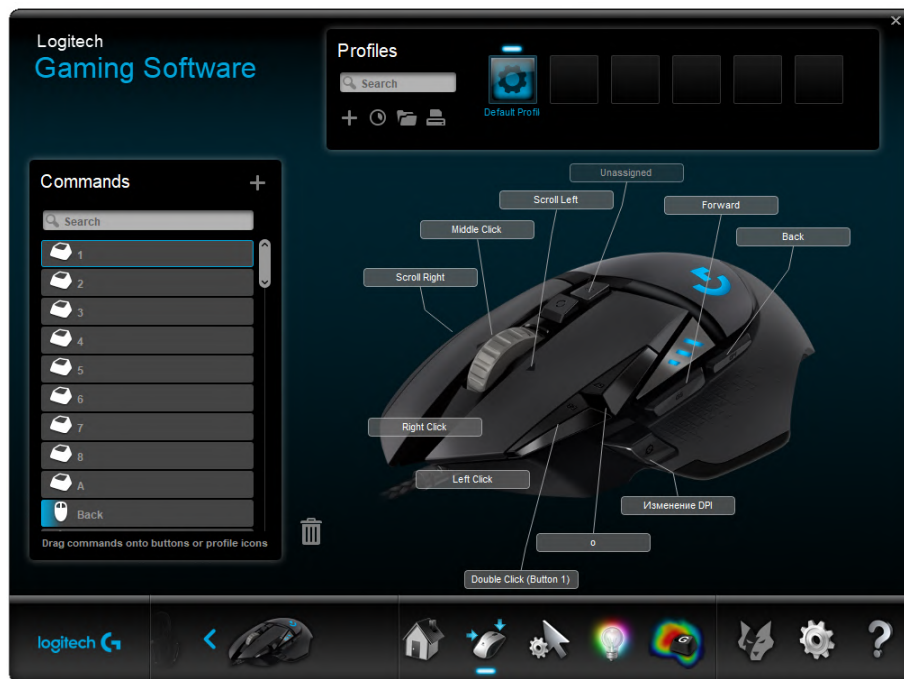


Figura 7: Ajustes de los botones

En la Figura 7 se ve la pantalla de los ajustes de los botones. En la parte izquierda están todos los comandos posibles y en la parte derecha se enseña la distribución actual de los comandos. En la parte de arriba se encuentran los distintos perfiles, en los cuales se puede guardar las distintas distribuciones, por ejemplo, una distribución para cada tipo de juego. Para programar un botón, se le da doble clic y se abre un menú con distintas opciones.

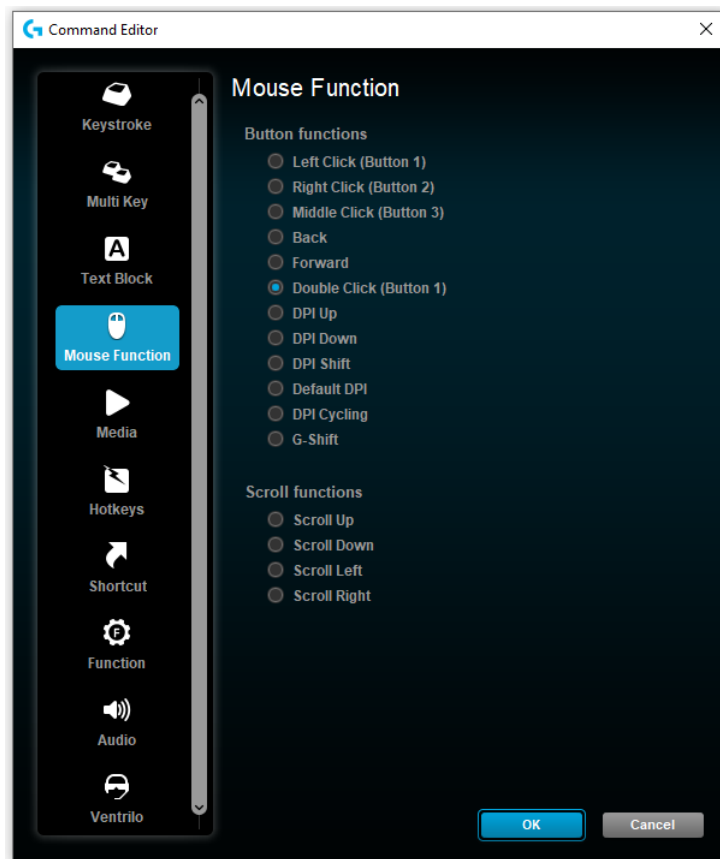
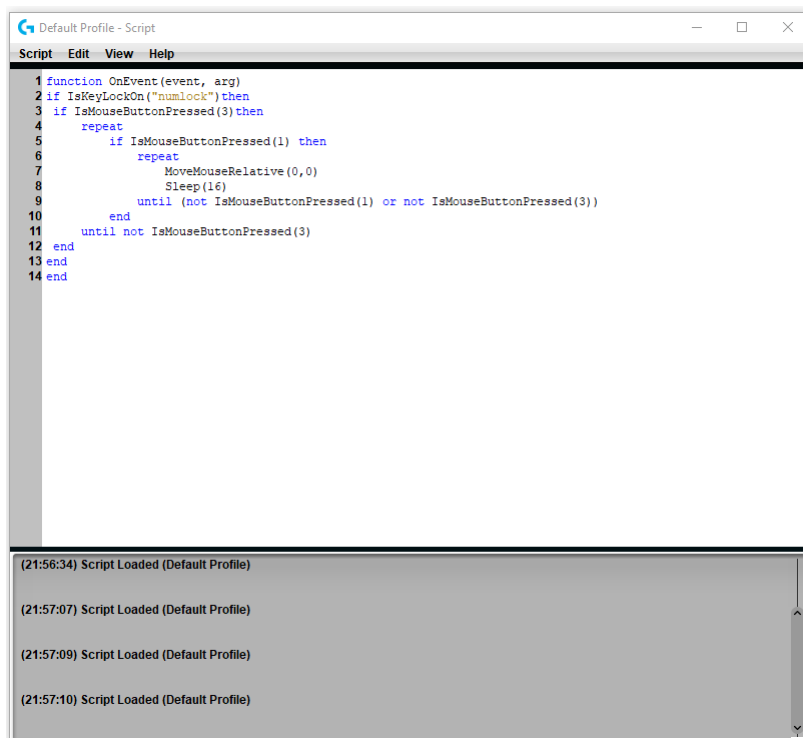


Figura 8: Editor de comandos

Entre las distintas opciones que se le pueden asignar al botón están:

- Una tecla
- Combinación de teclas (un macro)
- Función de ratón (clic izquierdo, clic derecho, clic izquierdo doble, etc.)
- Acciones multimedia
- Hotkeys de Windows (Alt + F4, Ctrl + Tab, etc.)
- Más acciones poco usadas

Si se selecciona un perfil y se hace clic en la opción "Scripting", sale un editor de texto como se ve en la Figura 9. En este campo de texto se puede escribir funciones en el lenguaje LUA que se van a ejecutar mientras el ratón esté conectado y el programa esté iniciado. En la figura de ejemplo se enseña un script que permite controlar el retroceso en los juegos de disparo moviendo el puntero del ratón hacia abajo.



```
1 function OnEvent(event, arg)
2 if IsKeyLockOn("numlock") then
3 if IsMouseButtonPressed(3) then
4     repeat
5         if IsMouseButtonPressed(1) then
6             repeat
7                 MoveMouseRelative(0,0)
8                 Sleep(16)
9             until (not IsMouseButtonPressed(1) or not IsMouseButtonPressed(3))
10        end
11    until not IsMouseButtonPressed(3)
12 end
13 end
14 end
```

(21:56:34) Script Loaded (Default Profile)

(21:57:07) Script Loaded (Default Profile)

(21:57:09) Script Loaded (Default Profile)

(21:57:10) Script Loaded (Default Profile)

Figura 9: Editor de scripts en LUA



Figura 10: Ajuste de sensibilidad

En la Figura 10 se enseñan los ajustes de sensibilidad. Hay una opción para configurar la DPI del ratón, en el caso del ratón de la figura va desde 100 hasta 25600. Se puede crear hasta 5 niveles diferentes, y se puede programar un botón para cambiar de nivel. Otra cosa que se puede configurar es la cantidad de señales que envía el ratón por segundo. Cuánto mayor es este número, más fluido se mueve el puntero, pero consume más recursos. Es recomendable tener este valor al máximo, porque el consumo de recursos es despreciable, pero la fluidez del puntero es muy importante.

6. Comandos de consola

6.1. Introducción

El proyecto de desarrollo de cualquier producto de programación tiene que contener una parte de testeo de sus funcionalidades. Estos tests pueden ser unitarios (solamente comprueban una función o porción pequeña de código) o de integración (comprueba el funcionamiento de un conjunto lógico de elementos). Un test de integración podría consistir en comprobar si un personaje puede añadir un objeto a su inventario, y un test de integración podría comprobar si un personaje puede comprar un objeto, añadirlo a su inventario, usarlo y posteriormente venderlo o dejarlo en algún sitio.

El proceso de testeo es muy complicado, lleva mucho tiempo cubrir todas las funciones del código con tests. Aún más difícil es poder testear juegos, ya que en un juego el desarrollador no puede ir directamente a una situación concreta. Para resolver este problema, en muchos juegos existen comandos de consola que permiten interactuar con el juego, ponerse en la situación deseada para comprobar su funcionamiento. La consola suele estar desactivada por defecto y para activarla tiene que haber una opción en el menú del juego.

6.2. Comandos en el The Elder Scrolls V: Skyrim

The Elder Scrolls: Skyrim es un juego de acción y rol perteneciente a la saga de juegos The Elder Scrolls. Es un juego de mundo abierto y se encuentra en un mundo de fantasía, donde existen entidades como elfos, orcos, vampiros, dragones, etc. Hay una historia principal y varias historias secundarias que el jugador puede pasar cuándo quiera. El personaje principal tiene muchas características, tales como vida, aguante, magia, nivel, etc. Tiene también varias habilidades mágicas y puede usar o tener varios objetos, ya sean espadas, armaduras o pociones mágicas.

Para poder testear el correcto funcionamiento de todos los elementos (objetos o misiones) del juego, los desarrolladores han implementado muchísimos comandos de consola, con los cuales se puede imitar cualquier situación.[8] La consola se abre con la tecla del símbolo ordinal (⁰) y cuando se presiona sale un rectángulo translúcido en el cual se puede escribir texto.



Figura 11: Consola de The Elder Scrolls V: Skyrim

En primer lugar están los comandos que activan o desactivan algunas funcionalidades generales:

- Inmortalidad del personaje principal
- Inteligencia Artificial de los personajes no jugables
- Colisiones del personaje seleccionado
- Modo de cámara libre
- Activación de árboles e hierba
- Activación de la interfaz gráfica

En segundo lugar, están los comandos que van dirigidos a una entidad. Para seleccionar esta entidad, se tiene que hacer un clic izquierdo encima de la misma. Entre los posibles comandos se encuentran:

- Activar o desactivar al personaje
- Añadir un objeto al personaje
- Cambiar los puntos de vida del personaje
- Matar al personaje
- Mover al personaje
- Cambiar el nivel del personaje
- Mostrar atributos del personaje, como su nivel o posición

Luego están los comandos orientados a las misiones. Hay que tener en cuenta que una misión suele tener varias etapas:

- Completar todas las misiones
- Completar una misión
- Elegir la etapa de una misión
- Reiniciar una misión

Después se encuentran los comandos relacionados con el personaje principal:

- Añadir o eliminar un objeto
- Añadir o eliminar una habilidad
- Añadir o eliminar un grito
- Cambiar el nivel
- Cambiar el nivel de una habilidad
- Mostrar la interfaz de cambio de raza
- Moverse a un posición del mapa
- Moverse a una entidad

Con estos comandos el proceso de testear el juego se acelera muchísimo, ya que se puede interactuar con cualquier objeto del juego y se puede ir instantáneamente a cualquier misión y su etapa. Los comandos también le presentan utilidad al jugador porque este puede usarlos con varias finalidades, ya sea experimentar o divertirse.

7. Cheats en los juegos de un solo jugador

7.1. Introducción

En los juegos de un solo jugador no hay competición, por lo tanto, los cheats están principalmente orientados a ofrecerle comodidad y diversión al jugador. En la mayoría de los casos, los cheats se usan para pasar niveles muy complicados, como podría ser tener que enfrentarse a un personaje enemigo que tiene mucha vida y no tener que estar mucho tiempo con él porque, el jugador podría aburrirse si no lo consigue en poco tiempo.

7.2. Trainers

Un trainer es un tipo de cheat que permite cambiar valores de las variables del juego en tiempo real. Estas variables podrían ser, por ejemplo, la vida del personaje principal o su dinero, sus objetos, personajes enemigos y sus atributos, etc. Las funcionalidades más clásicas de un trainer son la inmortalidad, el número infinito de balas y el dinero infinito.

Este cheat es un programa externo al juego. Lee la memoria del proceso del juego y allí interactúa con las porciones de memoria que contienen los valores de interés. Estas porciones de memoria se pueden detectar previamente con un programa externo, como podría ser Cheat Engine. Pero si la asignación de la memoria del juego es dinámica, entonces esta búsqueda de valores se tiene que hacer cada vez de nuevo y, por lo tanto, la tiene que hacer el mismo trainer.

En la Figura 12 se ve ejemplo de trainer para el juego Fallout 4.[9] Es un juego de acción y de disparo, dónde el personaje controlado por el jugador se encuentra en un mundo abierto junto a otros personajes humanos y animales. El personaje puede tener muchos objetos, como armas, chalecos, dinero, etc.

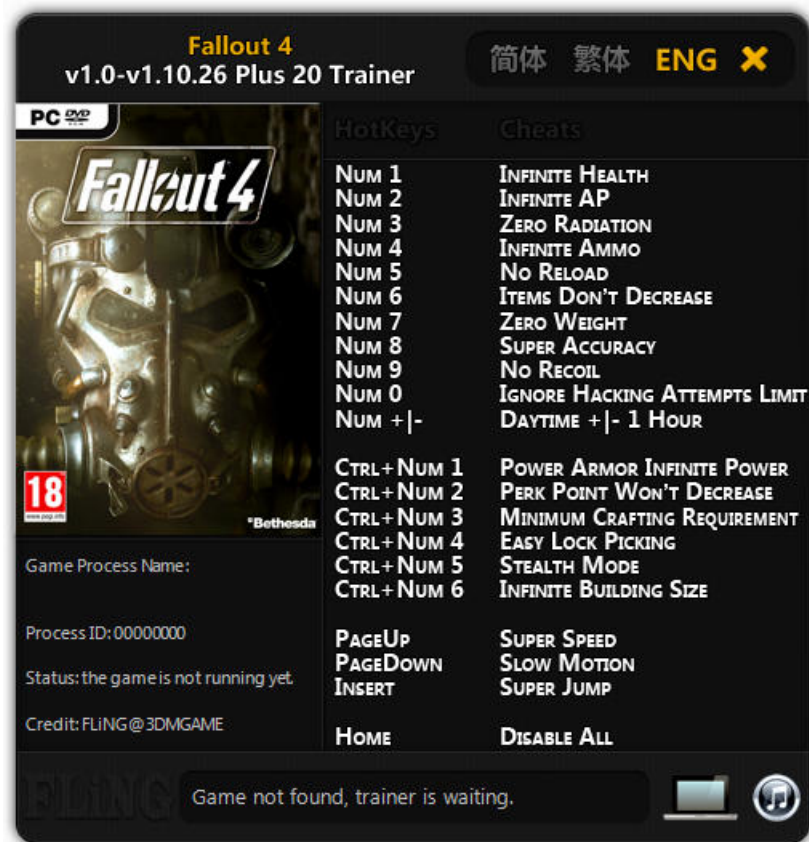


Figura 12: Trainer para Fallout 4

Entre las distintas funcionalidades se encuentran:

- Vida infinita. El personaje no puede morir.
- Aguante infinito. El personaje nunca se cansa al correr o saltar.
- Radiación desactivada. La vida del personaje no se decrementa por la radiación.
- Las armas del personaje no activan la animación de recarga.
- Los objetos del personaje no tienen peso y por lo tanto no afectan a la movilidad del mismo.
- Incremento de velocidad del personaje.
- Incremento del salto del personaje.

Todas estas opciones el jugador puede usar para disminuir la dificultad del juego.

7.3. Editores de partidas guardadas

Para no perder todo el progreso en un juego al salir, normalmente existe la opción de guardarlo para poder seguir con el juego en el momento de salida. Las partidas se guardan como una serie de ficheros en el ordenador y para algunos juegos existen programas que permiten descifrar la información guardada en estos ficheros y editarla usando una interfaz gráfica, para que el usuario pueda saber qué información hay guardada y qué información se puede editar.

Una serie de juegos en la que se usa a menudo el editor de partidas guardadas es Mass Effect. Es una serie de juegos de acción y de rol en mundo abierto, donde el jugador puede escoger misiones con un cierto grado de libertad y en estas misiones puede tomar varias decisiones que influyen en cómo se desarrolla la historia. Contiene 3 juegos, y algunos elementos de un juego dependen de las elecciones tomadas por el jugador en los juegos anteriores.

Si una persona quiere jugar al Mass Effect 3, pero no ha jugado a los juegos anteriores, en el juego se cargan las elecciones y opciones por defecto y, dependiendo del jugador, le pueden satisfacer o no. Si no le gusta alguna elección por defecto, puede guardas su partida, abrirla con un editor de partidas guardadas y empezar a cambiar parámetros a su gusto.

En la Figura 13 se ve el editor más usado para la serie Mass Effect con una partida abierta.[10]

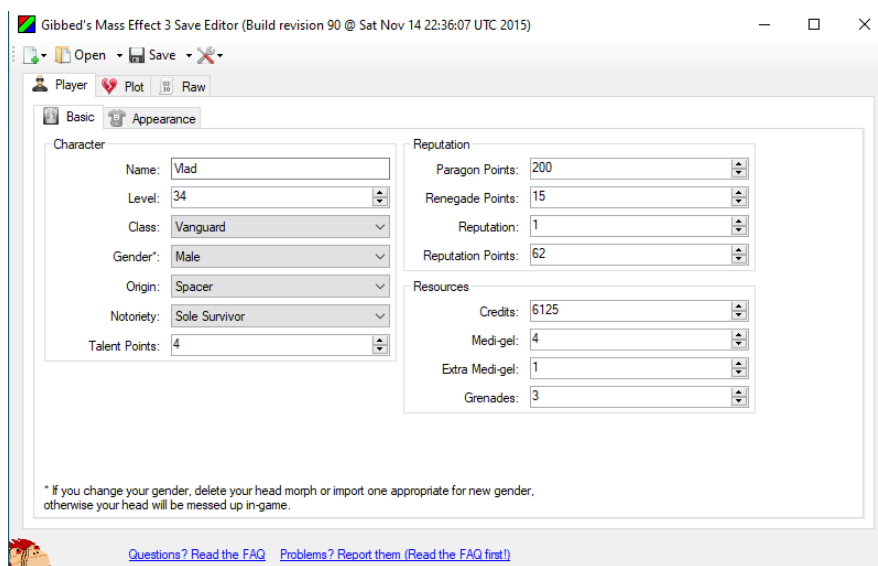


Figura 13: Mass Effect 3 save editor: Player

El programa tiene 3 pestañas: Player, Plot y Raw.

En la primera pestaña se encuentra toda la información del personaje principal. Se puede ver y cambiar el nombre del personaje, su nivel, su sexo. Se puede

escoger la clase del personaje, dependiendo de esta tendrá unas habilidades y atributos u otros. El origen y la notoriedad no influyen mucho en el juego, solamente cambian algunos diálogos. Se puede cambiar su número de talentos. Este número define cuántas habilidades y mejoras puede tener el personaje en el momento actual.

En la parte derecha de esta pestaña están los puntos de reputación. Estos puntos son un elemento del juego de rol, y se ganan haciendo acciones determinadas en casos determinados. Se puede cambiar los puntos del lado positivo del personaje y también los del lado negativo. Más abajo, se puede cambiar los puntos de reputación del personaje, que le permiten al jugador activar frases especiales en algunos diálogos. En la parte de abajo, se puede cambiar la cantidad de créditos (dinero) del jugador, su número de botiquines y granadas.

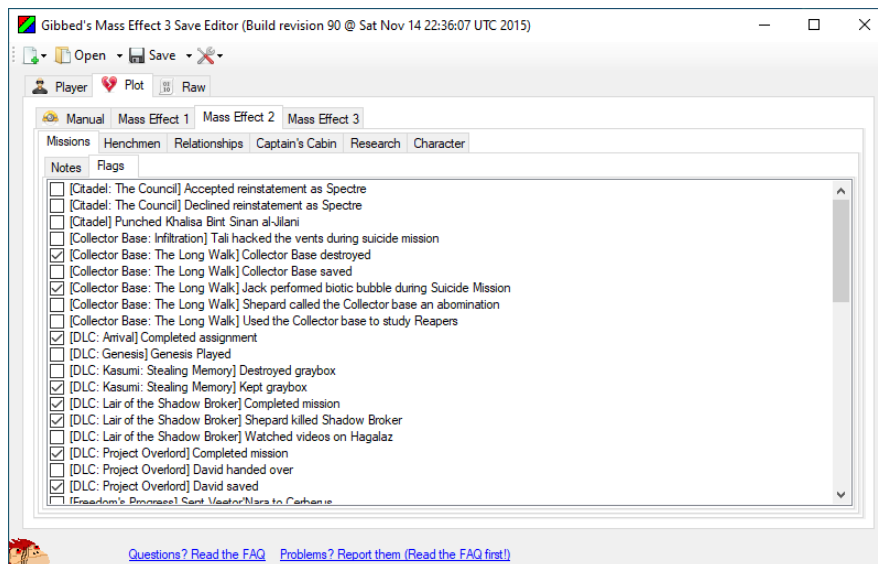


Figura 14: Mass Effect 3 save editor: Plot

En la Figura 14 se ve la segunda pestaña del programa. Aquí hay 4 pestañas interiores: Manual, Mass Effect 1, Mass Effect 2, Mass Effect 3. Mientras que la primera pestaña casi no se usa, en las otras se encuentra toda la información relevante de los tres juegos. Si se le da clic a la pestaña de Mass Effect 2, aparecen más pestañas. Aquí se pueden seleccionar qué misiones se han hecho en este juego, qué acciones se han tomado en estas, qué personajes se han reclutado en el equipo. Después se puede elegir si el personaje principal ha estado en relación con alguno de los personajes previamente. Finalmente, se puede seleccionar qué productos se han adquirido para el camarote del personaje principal, y qué mejoras se han encontrado o comprado para las armas o para la nave espacial del equipo.

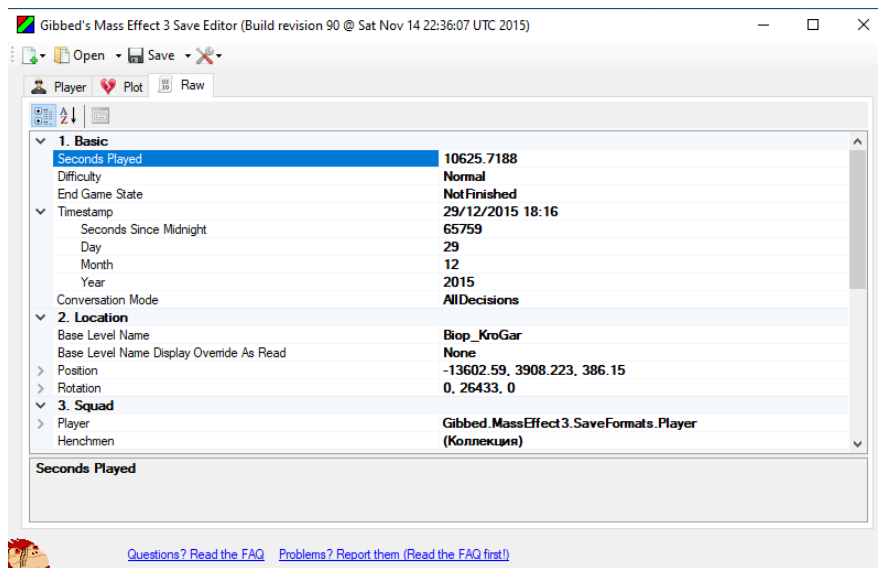


Figura 15: Mass Effect 3 save editor: Raw

En la Figura 15 se ve la pestaña Raw. Aquí se encuentran algunos valores que no presentan mucha importancia, pero igualmente han sido detectados en los ficheros de partidas guardadas y se pueden ver y cambiar. Por ejemplo, se puede ver cuántos segundos han pasado desde el inicio de la historia, qué dificultad se ha escogido, en qué posición exacta se encuentra el jugador en el mapa actual (nave espacial, planeta, etc..).

Hay que tener mucho cuidado al cambiar los valores de las partidas guardadas, ya que se puede escribir un valor incorrecto y así el juego no podrá cargar la partida. También, el editor ofrece todas las posibilidades de cambio sin restricción y, por lo tanto, el jugador tiene que tener cuidado a la hora de escoger elecciones que no son compatibles, por ejemplo, estar en relación con dos personajes a la vez o activar una acción que aún no ha pasado porque se encuentra en misiones posteriores.

7.4. Edición de ficheros

Los juegos de un solo jugador normalmente guardan todos los datos en la carpeta donde están instalados. Estos datos suelen estar distribuidos en varios ficheros de tamaño grande y estos ficheros suelen estar comprimidos. En algunos casos existen herramientas para descomprimir esta información, como en juegos de la serie S.T.A.L.K.E.R. La estructura del directorio de S.T.A.L.K.E.R. Call of Pripyat, el tercer juego de la serie, se puede ver en la Figura 16.

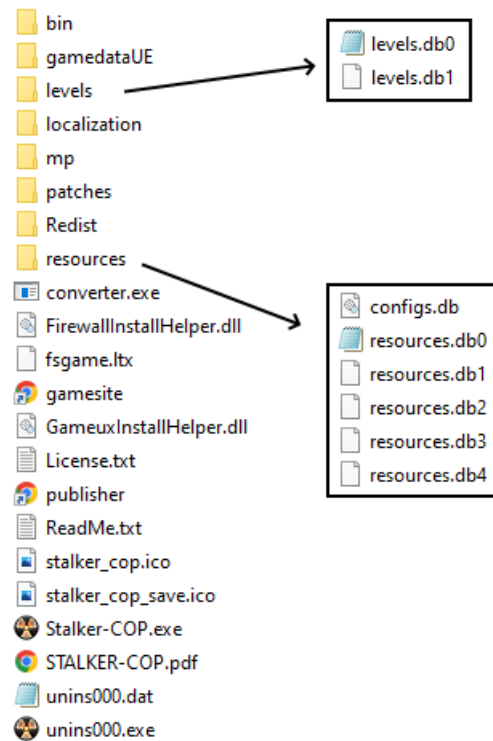


Figura 16: Estructura del directorio de S.T.A.L.K.E.R. Call of Pripyat

Las carpetas de interés son "levels" y "resources". En estas dos carpetas se encuentra toda la información necesaria pero, al abrir estos ficheros con cualquier editor de texto, es imposible entender la información, debido a que los ficheros están comprimidos. Para descomprimirlos, se puede usar el programa "S.T.A.L.K.E.R. Unpacker", que se puede encontrar en internet. Cuando se descomprime la información, se crea una carpeta con el nombre "gamedataUE" en el directorio del juego, y se necesita cambiar su nombre a "gamedata". El motor de juego busca esta carpeta y, si la encuentra, carga los datos de esta carpeta sobrescribiendo los datos cargados de los ficheros comprimidos (que contienen los datos no modificados).

En el directorio "gamedata" hay otros directorios, dentro de los cuales el más importante es "configs", ya que en este directorio se guarda la mayor parte de las variables del juego. Todos los ficheros de este directorio tienen el formato .ltx y se pueden abrir con un editor de texto (por ejemplo, Notepad++). Por ejemplo, en la carpeta "creatures" se puede encontrar el fichero "actor.ltx", que contiene todas las configuraciones del personaje principal. Al abrirlo, se puede ver que el fichero contiene variables y sus valores.

```

205 ;actor_condition only
206
207 jump_power          = 0.01
208 jump_weight_power   = 0.05
209 overweight_jump_k    = 5
210
211 stand_power          = -0.001.1
212 walk_power           = 0.00002
213 walk_weight_power    = 0.0002
214 overweight_walk_k    = 5
215 accel_k             = 3
216 sprint_k             = 100 ;75
217

```

Figura 17: Muestra de actor.ltx

En la parte del fichero mostrada en la Figura 17 se encuentran los coeficientes relacionados con las acciones del personaje como correr, saltar, ir agachado y su efecto en el aguante del mismo. Se puede modificar cualquiera de estos valores. Si se desea que el personaje no se canse tanto al hacer un salto, hay que decrementar el valor de la variable "jump_power". Hay que tener cuidado con los valores, debido a que si se le asigna un valor incorrecto, el juego posiblemente tendrá problemas y se cerrará.

Para cambiar los datos de un arma, se puede buscar el fichero correspondiente en la carpeta "weapons". En la Figura 14 se muestra una parte del fichero que le corresponde al arma AK-74.

```

26 ;-""'-----
27 hit_power          = 0.31, 0.31, 0.31, 0.31
28 hit_impulse        = 100      ; size of physio
29 hit_type           = fire_wound
30 fire_distance      = 200 ;1000
31 bullet_speed       = 550
32 rpm                = 570      ; max round ]
33 rpm_empty_click    = 200
34

```

Figura 18: Muestra de w_ak74.ltx

Algunos de los atributos a cambiar son:

- El daño que hace el arma
- La distancia máxima de fuego
- Velocidad de la bala

- Cantidad de disparos por minuto
- Todos los valores del retroceso

Observando las carpetas, también se pueden cambiar parámetros de los NPC (personajes no jugables), los monstruos, los textos, etc. Fuera de la carpeta "configs", hay otras carpetas de posible interés: "scripts", "sounds", "textures". En la segunda carpeta se encuentran los sonidos del juego en formato .ogg, así que no será un problema cambiar algún sonido si el jugador lo quiere, y en la tercera carpeta se encuentran las texturas del juego en formato .dds, que se pueden abrir y modificar con el programa Adobe Photoshop. En la primera carpeta se encuentran todos los scripts del juego, que básicamente controlan todas las entidades y sus interacciones en el juego. Cambiando las funciones que se encuentran dentro de estos scripts se puede alterar el comportamiento del juego y se usa principalmente para crear modificaciones para el juego.

Código 1: Muestra de game_relations.script

```
function set_npcs_relation(npc1, npc2, new_relation)
    local goodwill = 0
    if(new_relation=="enemy") then
        goodwill = -1000
    elseif(new_relation=="friend") then
        goodwill = 1000
    end
    if npc1 and npc2 then
        npc1:force_set_goodwill(goodwill, npc2)
    else
        abort("Npc not set in goodwill function!!!")
    end
end
```

En el Código 1, se enseña una función del fichero game_relations.script que establece una relación entre dos personajes no jugables. Se puede observar que la sintaxis del lenguaje se parece mucho a Python, por lo cual no va a ser difícil entender y cambiar el código para una persona que tiene conocimientos de este lenguaje.

8. Cheats en los juegos multijugador

8.1. Introducción

En los juegos multijugador en una partida participan varias personas y de aquí sale el deseo de competir de los jugadores. Igual que en los juegos clásicos, en los videojuegos también existen personas que prefieren aplicar métodos no legales para ganar o beneficiarse en una partida. Esta es la razón por la cual los cheats en los juegos multijugador se han convertido en un elemento muy común, presente en muchos juegos y odiado por los jugadores que prefieren no infringir las normas.

Hay distintos tipos de juegos online: de disparo, de acción, de carrera, de rol, etc. Los cheats son muy populares en los juegos de disparo y en los MOBA (videojuegos multijugador de arena de batalla en línea). Un juego de disparo es un juego en el que el jugador controla a un personaje que tiene varias armas y su objetivo es eliminar a los personajes enemigos. Un MOBA es un juego similar a un juego de estrategia en tiempo real, con la diferencia de que un jugador normalmente solo controla un personaje, que tiene unas habilidades, unos objetos y unas características que lo hacen único.

Se explicarán los cheats multijugador basándose en los siguientes tres juegos:

- Counter Strike: Global Offensive
- Dota 2
- PUBG: BATTLEGROUNDS

8.2. Tipos de cheats en los shooters

Los shooters son el tipo de juego favorito para los cheaters por dos razones: por un lado, ofrecen muchas posibilidades para usar trampas y, por otro lado, la comunidad de jugadores de este género es enorme. Debido a esto, casi cada shooter está sujeto a la aparición de cheats de todo tipo.

8.2.1. Aimbot

Aimbot es el cheat más clásico en todos los juegos de disparo. Es un programa que apunta y dispara por sí solo a los enemigos, con lo cual se elimina por completo la dificultad de apuntar bien para el jugador.[12]

Las versiones del aimbot pueden variar: en algunos casos solamente corrige la mira para que apunte a un jugador enemigo y así solamente funciona de apoyo para el jugador, y en otros casos controla la mira de tal forma que un jugador enemigo puede estar detrás del cheater y este le mata sin tener que mover la cámara.[11]

Algunas versiones del cheat permiten elegir en qué parte del cuerpo enemigo se tiene que producir el disparo. Esto se hace porque, por un lado, el daño varía

según la parte de cuerpo y, por otro lado, permite no dañar el casco o el chaleco del jugador enemigo, con la finalidad de poder recogerlo de su cuerpo después de matarlo.

También se puede poner una velocidad de movimiento de la mira para que no parezca que haya sido un programa, sino el jugador.

¿Cómo funciona un aimbot a nivel técnico? Principalmente, funciona analizando las coordenadas de los jugadores en una partida. Un personaje tiene 3 coordenadas (X, Y, Z), que definen su posición en el mapa. También tiene 2 coordenadas de vista: XY y XZ. Estas dos definen a donde está mirando el personaje.[13]

Sabiendo las coordenadas del jugador y las coordenadas de un posible enemigo, se puede hacer un cálculo para definir las coordenadas de vista usando la artotangente de dos parámetros, donde "p" es el jugador y "e" es el enemigo.

$$XY = Arctg(X_p - X_e, Y_p - Y_e)$$

$$XZ = Arctg(Z_p - Z_e, X_p - X_e)$$

Ahora solamente falta escribir estos valores en las posiciones de memoria correspondientes para que el personaje apunte en la dirección deseada.

Con este cheat se consigue una ventaja enorme respecto a los enemigos y, por lo tanto, si un cheater no quiere ser detectado, tiene que usar el aimbot muy cuidadosamente y la menor cantidad de veces posible.

8.2.2. ESP

ESP significa extrasensory perception (percepción extrasensorial) y consiste en mostrarle al jugador toda la información oculta que puede presentar alguna utilidad. En los shooters, el objetivo principal de los cheats ESP es poder ver a los enemigos a través de las paredes.[14] Existen varios grados de ESP. El más básico de estos solamente pinta un radar en forma de rectángulo o círculo, en el cual se ve el jugador en el centro y los posibles enemigos se ven en sus posiciones correspondientes, si se encuentran dentro del límite del radar.



Figura 19: Radar PUBG: BATTLEGROUNDS



Figura 20: Radar PUBG: BATTLEGROUNDS

En las Figuras 19 y 20 se ve la implementación de un radar para el juego PUBG: BATTLEGROUNDS. Consiste en un rectángulo que ocupa una parte bastante grande de la pantalla (lo cual no es óptimo) y se pueden ver varios iconos de distintos colores, que representan a los jugadores, objetos y vehículos. En la parte de arriba de la derecha se puede activar a desactivar cualquiera de los tres tipos de iconos.

Se ve que el jugador principal se representa con un círculo verde, los jugadores enemigos se representan con un círculo amarillo, los objetos se representan con una imagen pequeña y tienen un subrayado verde, y los vehículos se representan con su foto. No es la mejor versión de un radar, ya que los elementos son muy poco visibles y hay que prestar mucha atención a los detalles del radar, perdiendo

así la atención en el propio juego.

Otra implementación más elaborada consiste en subrayar los modelos de los jugadores directamente en la pantalla. De esta forma se ve en todo momento donde se encuentran los enemigos y se les puede apuntar sin tener que ver el mapa del radar. Este tipo de ESP se llama wall-hack (porque permite ver a través de las paredes) y es el cheat más típico de los shooters junto con el aimbot. Existen varias formas de subrayar: se puede mostrar un rectángulo dentro del cual se encuentra el enemigo, se puede subrayar el contorno del enemigo o se puede subrayar las diferentes partes del cuerpo enemigo con varios colores con la finalidad de saber mejor dónde disparar.



Figura 21: Wall-hack en CS:GO

En la Figura 21 se ve una versión muy clásica del wall-hack para el CS:GO. Los jugadores se visualizan mostrando su "esqueleto" en un rectángulo. En la parte de arriba de cada jugador se ve la distancia a la cual se encuentra el jugador respecto al cheater. Los jugadores del equipo enemigo se ven con un color rojo, y los del equipo del cheater se visualizan con el verde.



Figura 22: Wall-hack en PUBG: BATTLEGROUNDS

En la Figura 22 se ve una versión un poco distinta de wall-hack para el PUBG: BATTLEGROUNDS. Se visualiza el modelo completo de los jugadores (en este juego todos los jugadores son enemigos) que se encuentran detrás de un obstáculo para ser visibles para el jugador.

En el Código 2 se encuentra una implementación básica en Python de un wall-hack para el Counter-Strike: Global Offensive.[15]

Este programa funciona leyendo el proceso del juego y accediendo a sus porciones de memoria donde se guarda la información necesaria. Primero de todo se definen los offsets de los elementos de una partida. Se lee el proceso del juego y se entra en un bucle. Adentro se recorren las 32 entidades que corresponden con los posibles jugadores, en caso de que la posición sea válida (el jugador existe) se lee el identificador de su equipo y la posición de la configuración de su brillo. Luego, después de calcular los offsets, se define el color de brillo de estas entidades y se escribe el valor "1" en la casilla de memoria responsable de activar el brillo. En la Figura 23 se puede ver cómo funciona este cheat en el juego.

Código 2: Diamond

```
import pymem
import pymem.process

dwEntityList = 0x4DA31EC
dwGlowObjectManager = 0x52EB678
m_iGlowIndex = 0xA438
m_iTeamNum = 0xF4

def main():
    print("Diamond has launched.")
    pm = pymem.Pymem("csgo.exe")
```



```

client = pymem.process.module_from_name(pm.process_handle, "
client.dll").lpBaseOfDll

while True:
    glow_manager = pm.read_int(client + dwGlowObjectManager)

    for i in range(1, 32): # Entities 1-32 are reserved for
        players.
            entity = pm.read_int(client + dwEntityList + i * 0x10
    )

    if entity:
        entity_team_id = pm.read_int(entity + m_iTeamNum)
        entity_glow = pm.read_int(entity + m_iGlowIndex)

        if entity_team_id == 2: # Terrorist
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x8, float(1)) # R
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0xC, float(0)) # G
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x10, float(0)) # B
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x14, float(1)) # Alpha
            pm.write_int(glow_manager + entity_glow * 0
x38 + 0x28, 1) # Enable glow

        elif entity_team_id == 3: # Counter-terrorist
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x8, float(0)) # R
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0xC, float(0)) # G
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x10, float(1)) # B
            pm.write_float(glow_manager + entity_glow * 0
x38 + 0x14, float(1)) # Alpha
            pm.write_int(glow_manager + entity_glow * 0
x38 + 0x28, 1) # Enable glow

if __name__ == '__main__':
    main()

```

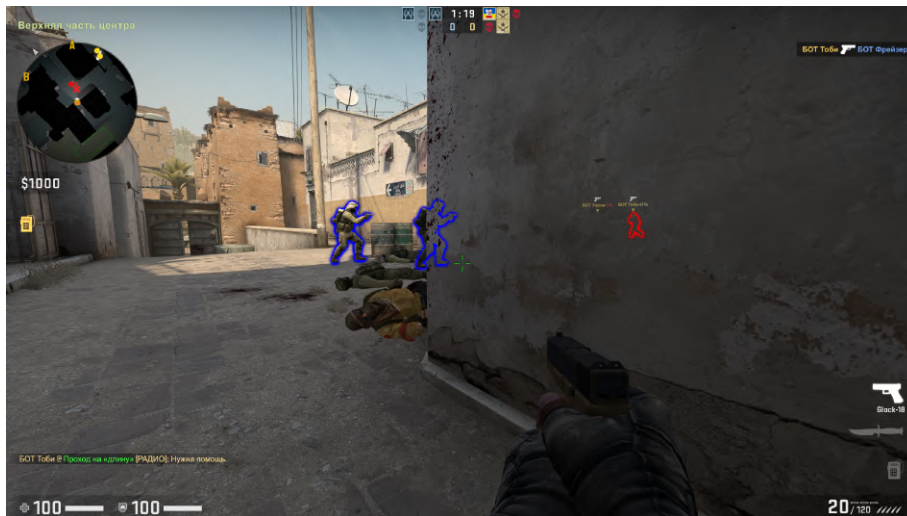


Figura 23: Diamond

8.2.3. Triggerbot

Triggerbot es un programa que hace lo contrario al aimbot. Hace automáticamente un disparo cuando el jugador está apuntando a un enemigo.[12] Es una técnica que generalmente se usa cuando el jugador no se está moviendo y está mirando a un punto concreto, donde hay una alta probabilidad de que aparezca un jugador enemigo. Por ejemplo, puede haber un pasillo con una puerta y los jugadores enemigos pueden entrar desde el lado izquierdo. En este caso el jugador posiciona su mira casi al extremo del borde izquierdo de la puerta, para no perder tiempo en mover el ratón.

Puede tener varios ajustes:

- Puede tener o no tener una tecla para activar o desactivar el script para no generar sospechas.
- Puede permitir escoger ciertas zonas de cuerpo para disparar. Por ejemplo, se puede seleccionar la opción de que solamente dispare si el jugador está apuntando a la cabeza del enemigo.
- Puede permitir seleccionar un retraso antes de disparar, con la finalidad de disminuir las posibles sospechas.
- Puede seleccionar la opción de no disparar cuando el jugador enemigo no se encuentra directamente visible por alguna razón, por ejemplo se encuentra dentro del área de impacto de una granada de humo. Si un jugador hace una baja en estas condiciones, es probable que está usando algún cheat.

En el Código 3 se ve una implementación de un triggerbot para el juego de Counter-Strike: Global Offensive.[16] En esta implementación se ve al principio

de todo unos offsets (desplazamientos en la memoria) que contienen algunas acciones y datos del juego. Por ejemplo, *dwForceAttack* es responsable de hacer el disparo con el botón izquierdo del ratón. Más abajo hay un bucle infinito que lee las posiciones de los jugadores relativas al punto de visión del jugador principal y, si se detecta un jugador enemigo en la mira, se graba el valor 6 en la posición del ataque del jugador principal, lo cual produce un disparo.

Código 3: Sapphire

```
import keyboard
import pymem
import pymem.process
import time
from win32gui import GetWindowText, GetForegroundWindow

dwEntityList = 81605708
dwForceAttack = 52443380
dwLocalPlayer = 14382556
m_fFlags = 260
m_iCrosshairId = 71736
m_iTeamNum = 244

def main():
    print("Sapphire has launched.")
    pm = pymem.Pymem("csgo.exe")
    client = pymem.process.module_from_name(pm.process_handle, "
client.dll").lpBaseOfDll

    while True:

        player = pm.read_int(client + dwLocalPlayer)
        entity_id = pm.read_int(player + m_iCrosshairId)
        entity = pm.read_int(client + dwEntityList + (entity_id -
1) * 0x10)

        entity_team = pm.read_int(entity + m_iTeamNum)
        player_team = pm.read_int(player + m_iTeamNum)

        if entity_id > 0 and entity_id <= 64 and player_team !=
entity_team:
            pm.write_int(client + dwForceAttack, 6)

        time.sleep(0.006)

if __name__ == '__main__':
    main()
```

Este tipo de cheat es fácilmente detectable en partidas de bajo nivel, debido a que los jugadores con poca experiencia no tienen una reacción muy rápida ni tampoco controlan muy bien la mira.

8.2.4. BunnyHop

BunnyHop es una técnica mediante la cual un jugador puede desplazarse más rápido por el mapa saltando repetidamente y es un tipo especial y común de bug. Apareció por primera vez en el juego Quake.[17] Cuando un jugador salta, su velocidad horizontal se calcula como una combinación lineal de la velocidad actual y la dirección deseada y no está limitada. De esta forma, si el jugador empieza a desplazarse haciendo un salto hacia delante y ligeramente hacia un lado con un ángulo determinado, su velocidad incrementa. Si el jugador vuelve a saltar otra vez justo al llegar a la tierra, esta velocidad no se pierde y sigue incrementando. De esta forma, se puede llegar a moverse muy rápido por el mapa si se consigue hacer el salto en el momento adecuado.

Esta técnica no está prohibida y se usa en muchos juegos tanto en el ámbito público como en los torneos oficiales. Principalmente, se usa en los juegos de las antologías Counter-Strike, Quake y en algunos otros juegos de la empresa Valve. Ha llegado a ser un elemento muy importante de estos juegos, y muchos jugadores han creado mapas propios para poder entrenar el bunnyhop.

Como es bastante complicado presionar la tecla de salto en el momento correcto, se han creado muchos scripts que lo hacen automáticamente. En los Códigos 4 y 5 hay dos implementaciones de este cheat para el juego Counter-Strike: Global Offensive.

La primera implementación está creada en el lenguaje AHK, un lenguaje propio del programa AutoHotkey.[18] Consiste en un bucle que va presionando la tecla de espacio cada 50 milisegundos en el caso de estar presionada por el jugador. Si el jugador la deja de presionar, no se hace nada. No se mete en la memoria del juego.

Código 4: Bunnyhop en AHK

```
F9::suspend, toggle

*space::
Loop
{
    GetKeyState, state, space, P
    If state = U
        break

    Send, {space}
    Sleep, 20
}
return
```

La segunda implementación es más elaborada y está hecha con Python.[19] Lo primero que se puede observar es que este código interactúa con el proceso del juego. Al principio de todo están la dirección de memoria responsable de la acción de saltar y la dirección que contiene la información del jugador. Al

ejecutarse, el script empieza a leer el proceso del juego y entra en un bucle donde se ejecuta la parte del código responsable de los saltos. Se mira primero de todo si el jugador he presionado el espacio. Si es así, se lee la información del jugador para saber si está en el suelo. Si es así, se escribe en la memoria la acción del salto, provocando el salto del jugador en el juego. De esta forma, el jugador al mantener presionado el espacio va a saltar justo después de haber llegado al suelo, lo cual aumentará su velocidad.

Código 5: Ruby

```
import keyboard
import pymem
import pymem.process
import time
from win32gui import GetWindowText, GetForegroundWindow

dwForceJump = (0x51F4D88)
dwLocalPlayer = (0xD36B94)
m_fFlags = (0x104)

def main():
    print("Ruby has launched.")
    pm = pymem.Pymem("csgo.exe")
    client = pymem.process.module_from_name(pm.process_handle, "
client.dll").lpBaseOfDll

    while True:
        if not GetWindowText(GetForegroundWindow()) == "Counter-
Strike: Global Offensive":
            continue

        if keyboard.is_pressed("space"):
            force_jump = client + dwForceJump
            player = pm.read_int(client + dwLocalPlayer)
            if player:
                on_ground = pm.read_int(player + m_fFlags)
                if on_ground and on_ground == 257:
                    pm.write_int(force_jump, 5)
                    time.sleep(0.08)
                    pm.write_int(force_jump, 4)

            time.sleep(0.002)

if __name__ == '__main__':
    main()
```

8.2.5. Control de retroceso

En los juegos de disparo, el retroceso es un elemento común que tiene dos objetivos: en primer lugar, dar una sensación más realista de disparo para que el jugador pueda sentir mejor el arma y, por otro lado, aumentar la dificultad del juego haciendo el proceso de apuntar y disparar bastante más complejo.

El retroceso puede estar implementado de varios modos. A continuación se explican dos ejemplos:

- **PUBG: BATTLEGROUNDS.** En este juego el retroceso es un movimiento de la mira del jugador que tiene tres direcciones (arriba, izquierda, derecha) que se ejecuta cada vez que el jugador hace un disparo con un arma. El retroceso depende mucho del arma y también de los accesorios puestos en el arma, como podría ser un silenciador, una empuñadora o un compensador. Para las armas de distancia larga, como los rifles de francotirador, la bala siempre va al punto central de la mira. Para las armas de distancia corta, la dirección de la bala puede variar un poco. En general, controlar el retroceso es bastante complicado para un jugador con poca experiencia. En la Figura 24 se ven 6 ráfagas de tiros del arma QBZ-95. Como se puede ver, son bastante parecidas, pero no iguales debido a un factor de aleatoriedad.[20]

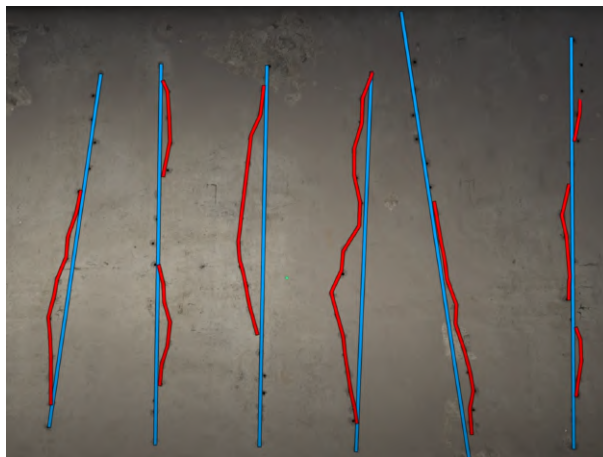


Figura 24: Ejemplo de retroceso en el PUBG: BATTLEGROUNDS

- **Counter-Strike: Global Offensive.** En este juego el retroceso se diferencia mucho del PUBG: BATTLEGROUNDS y muchos otros juegos. Para cada arma hay definido un patrón de movimientos para cada bala. Para aprender a controlar este tipo de retroceso, simplemente hay que memorizar los patrones. Por otro lado, la bala tampoco va al centro de la mira, sino que también tiene un patrón definido. Debido a esto, para poder hacer una ráfaga correcta, a veces no se le tiene que apuntar al enemigo, sino a un sitio que se encuentra en una posición determinada cercana al enemigo. Para una persona que no haga jugado al Counter-Strike, puede resultar complicado, ya que es un retroceso totalmente distinto a los otros. En la Figura 25 se ve el patrón de retroceso del arma AK-47. Como se ve, para hacer que todas las balas lleguen al mismo sitio, al empezar la ráfaga se tiene que bajar bastante la mira con un pequeño movimiento hacia la derecha. Después se tiene que mover rápidamente hacia la izquierda,

otra vez a la derecha y una vez más a la izquierda, sin hacer movimiento vertical.

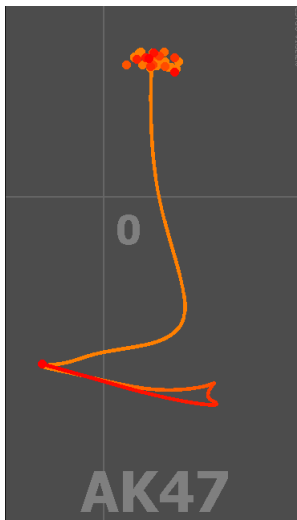


Figura 25: Ejemplo de retroceso en el Counter-Strike: Global Offensive

Para controlar el retroceso se puede usar scripts externos (que no se meten en el proceso del juego) y el algoritmo va a ser distinto para cada juego. Bastantes compañías de accesorios para los ordenadores que venden ratones crean posibilidades de crear u usar scripts propios que se almacenan dentro del propio ratón, entre los cuales están Razer, Logitech y A4Tech.

En el Código 6 hay un script bastante básico para controlar el retroceso a un nivel bastante universal para los ratones de gama gaming de Logitech y está escrito en el lenguaje LUA. Este script funciona solamente si está activada la tecla de NumLock. En este caso, si está presionado el botón derecho (cuya acción es apuntar), ejecuta un bucle dentro del cual se mira si el botón izquierdo está presionado (cuya acción es disparar). Si es así, entonces con la acción *MoveMouseRelative(0,6)* se mueve el ratón hacia abajo hasta que no se deje de disparar o apuntar.

Código 6: Script universal para el control del retroceso

```
function OnEvent(event, arg)
if IsKeyLockOn("numlock") then
if IsMouseButtonPressed(3) then
repeat
if IsMouseButtonPressed(1) then
repeat
MoveMouseRelative(0,6)
Sleep(16)
until (not IsMouseButtonPressed(1) or not
IsMouseButtonPressed(3))
end
until not IsMouseButtonPressed(3)
```

```

end
end
end

```

Con este script se permite disminuir la necesidad de mover el ratón hacia abajo. Sin embargo, el retroceso horizontal no se controla, debido a que es aleatorio y no hay modo de averiguar en que lado se mueve la mira después de un disparo. Otra desventaja de este algoritmo es que no es dinámico, en el sentido de que no se puede cambiar la velocidad de desplazamiento del ratón sin tener que cambiar el parámetro correspondiente. Si un jugador tiene un arma que tiene mucho retroceso y otra arma que tiene muy poco, tendría que ajustar el parámetro al arma que tiene poco, ya que si lo hace al revés, al disparar con el arma que tiene poco retroceso se hará un movimiento demasiado rápido hacia abajo y el jugador tendrá que mover el ratón hacia arriba, que es mucho más incómodo y antinatural.

Este script no ayuda a controlar el retroceso en el juego Counter-Strike: Global Offensive, ya que este tiene un sistema diferente. En los Códigos 7 y 8 se encuentra una parte de un script escrito en AHK especialmente para este juego.[21] Con el primer trozo de código se permite escoger un script u otro dependiendo del arma actual del jugador. Hay una tecla para pausar la ejecución del programa y también una tecla para salir. El segundo trozo de código es un ejemplo de control de retroceso para un arma. Primero de todo, se mira si el jugador ha escogido el arma AK-47. Después se entra en un bucle, dentro del cual hay 30 bloques de código parecidos con parámetros distintos. Cada bloque de código es responsable de hacer el movimiento después del disparo correspondiente (en total hay 30 disparos para esta arma). Antes de ejecutar un movimiento, cada vez se revisa si el jugador aún tiene pulsado el botón izquierdo, para que solamente se mueva el ratón cuando el jugador esté disparando.

Código 7: Posibilidad de elegir arma

```

;Key binds
key_AK:="F4"
key_M4A1:="F6"
key_M4A4:="F7"
key_Famas:="F8"
key_Galil:="F9"
key_UMP:="F12"
key_AUG:="Home"
key_SG:="End"
key_RCoFF:="F11"
key_Terminate:="F10"
key_shoot:="LButton"
key_zoom:="LAlt"

```

Código 8: Control de retroceso para el arma AK-47

```

;AK-47
if ak
{

```



```

loop
{
DllCall("mouse_event", uint, 2, int, 0, int, 0, uint, 0, int, 0)
sleep 50
if !GetKeyState(key_shoot)
{
DllCall("mouse_event", uint, 4, int, 0, int, 0, uint, 0, int, 0)
break
}
DllCall("mouse_event", "UInt", 0x01, "UInt", -4*modifier, "UInt",
7*modifier)
sleep 99
if !GetKeyState(key_shoot)
{
DllCall("mouse_event", uint, 4, int, 0, int, 0, uint, 0, int, 0)
break
}
DllCall("mouse_event", "UInt", 0x01, "UInt", 4*modifier, "UInt",
19*modifier)
sleep 99

...

if !GetKeyState(key_shoot)
{
DllCall("mouse_event", uint, 4, int, 0, int, 0, uint, 0, int, 0)
break
}
}
}

```

8.2.6. Efectos visuales

En los juegos de acción existen situaciones en la que el campo de visión del jugador se reduce debido a alguna acción de otros jugadores. Este es el caso de las granadas de humo y de las granadas cegadoras. Existen cheats que permiten deshabilitar estos efectos visuales y hacer que estas granadas no afecten al jugador. Se suelen llamar NoSmoke o NoFlash, dependiendo de su funcionalidad.

Código 9: Emerald

```

import pymem
import pymem.process
import time

dwLocalPlayer = (0xD36B94)
m_flFlashMaxAlpha = (0xA40C)

def main():
    print("Emerald has launched.")
    pm = pymem.Pymem("csgo.exe")
    client = pymem.process.module_from_name(pm.process_handle, "
client.dll").lpBaseOfDll

```

```

while True:
    player = pm.read_int(client + dwLocalPlayer)
    if player:
        flash_value = player + m_flFlashMaxAlpha
        if flash_value:
            pm.write_float(flash_value, float(0))
        time.sleep(1)

if __name__ == '__main__':
    main()

```

En el Código 9 hay una implementación de un cheat que elimina el efecto visual de ceguera.[22] Como se ve, dentro de un bucle se está leyendo el offset del jugador y se le suma el offset del estado de la ceguera para saber si el jugador está cegado. En este caso simplemente se reescribe el campo con el valor 0 haciendo que el efecto visual desaparezca.

8.3. Tipos de cheats en los MOBA

8.3.1. Introducción

Un MOBA (videojuego multijugador de arena de batalla en línea) es un subgénero de estrategia en tiempo real en línea donde un jugador normalmente solo controla a un personaje.[30] Cada personaje tiene sus habilidades únicas y puede adquirir objetos durante la partida. Hay dos equipos y en cada equipo hay de 3 a 5 jugadores. Cada equipo tiene unas construcciones con distintas funcionalidades y el objetivo es destruir la construcción principal del equipo contrario. El origen de los MOBA fue una modificación del juego WarCraft 3 y actualmente los juegos más famosos de este género son League of Legends y Dota 2.

8.3.2. Tipos de cheats en los MOBA

Un MOBA, al ser parecido a un juego de estrategia, es un juego en proyección isométrica muy complicado en el sentido de que hay muchas acciones y muchos elementos a controlar. Un personaje suele tener 4 habilidades y 6 objetos, que se resumen en 10 botones que hay que saber presionar rápidamente, sin contar que hay que controlar el movimiento y las acciones del personaje y también la cámara, que también son bastantes botones del teclado y del ratón. Por lo tanto, una de las funcionalidades de los cheats para los MOBA es el control automático de ciertos elementos, por ejemplo algunos objetos del personaje.

Por otro lado, una de las mecánicas de las estrategias que hacen que el juego sea más complicado es el campo de visión y la niebla de guerra. Cada personaje y construcción tienen un campo de visión limitado, y todo lo que no está dentro del campo de visión, no aparece visible para el equipo. Así que otra funcionalidad muy importante de los cheats los MOBA es poder mostrar la información que no es visible para un jugador.

Los ejemplos de cheats para MOBA se basan en el juego Dota 2.

8.3.3. Mejoras visuales

Como los juegos MOBA están muy llenos de información y efectos visuales, hay una gran cantidad de elementos ocultos que se le podría mostrar al jugador para que este obtenga más información durante la partida.

En la Figura 26 se puede ver qué información tiene un jugador por defecto. En la parte de arriba, están los personajes de nuestro equipo y los del equipo contrario y la cantidad de bajas que ha hecho cada equipo. En la parte de abajo, en la izquierda, se ve el mapa de la partida y en la parte central se encuentra la información del personaje controlado por el jugador. En este caso el personaje tiene 5 habilidades y 6 objetos en su inventario. En la parte central se ven el personaje controlado por el jugador y un personaje enemigo. Arriba de ambos personajes hay una barra que enseña cuánta vida tiene el personaje y su nivel.



Figura 26: Interfaz de Dota 2

En cada parte de estas, Octarine ofrece posibilidades de aumentar la información. En primer lugar, se puede activar la posibilidad de ver la cantidad exacta de los puntos de vida y maná de un jugador enemigo. También, se puede ver las recargas (cooldowns) de sus habilidades y objetos.



Figura 27: Cooldowns de un personaje

La niebla de guerra no permite ver lo que está pasando fuera del campo de visión del personaje. Pero usando cheats se puede captar acciones enemigas fuera del campo de visión y crear algún aviso, como por ejemplo en el caso de la habilidad "Sunstrike" del personaje Invoker. Esta habilidad le permite atacar con un rayo una zona circular que el equipo enemigo no puede ver. Sin embargo, usando el cheat se puede ver a dónde está apuntando con la habilidad y también en cuántos milisegundos se producirá el daño causado por la misma.



Figura 28: Sunstrike en región escondida

Una mecánica del juego muy importante es el pergamino de teletransporte. Es un objeto especial que le permite a un jugador teletransportarse desde un sitio cualquiera a cualquier edificio de su equipo. Si un jugador enemigo está el pergamino de teletransporte dentro de la niebla de la guerra, no se podría ver si no hubiera esta opción en el cheat. Se puede ver el punto inicial, el punto final

y también el tiempo dentro del cual el jugador aparecerá en el punto final si no se interrumpe el uso del objeto.



Figura 29: Pergamino de teletransporte

Algunos héroes tienen habilidades para crear ilusiones. Una ilusión es una entidad que tiene la forma de un personaje y es controlada por el jugador. Aparece de forma temporal (para unos segundos) y normalmente hace menos daño y tiene menos vida que el personaje original. Al morir, una ilusión simplemente desaparece. Para el equipo enemigo es imposible saber qué entidad es el personaje principal y qué entidad es una ilusión, ya que tienen la misma apariencia. Con el cheat se puede detectar ilusiones cambiando su apariencia.



Figura 30: Ilusiones

En el juego es muy importante saber cuándo los enemigos pueden ver al perso-

naje y cuándo no. Para obtener esta información también hay una funcionalidad del cheat. Si al menos un jugador enemigo puede ver al personaje, debajo de este aparece un círculo indicando que está visible.



Figura 31: Visibilidad

Otra mecánica muy parecida a la anterior es la detección de la invisibilidad. Cuando un personaje entra en el estado de invisibilidad, aparece visible para su equipo, pero no para el contrario. Por otro lado, existen formas de detectar personajes invisibles, por ejemplo con un objeto llamado Sentry Ward. Es un objeto estático que un jugador puede poner en algún sitio del mapa donde es probable que se encuentre un personaje invisible. El cheat permite saber si un personaje siendo invisible se encuentra en una zona de detección de invisibilidad del equipo enemigo. En la Figura 32 aparece un icono encima del personaje indicando que este personaje se encuentra bajo la zona de visión de un Sentry Ward (objeto a la derecha del personaje). Si el personaje activa la invisibilidad, el equipo contrario igualmente podrá verlo.



Figura 32: Detección de invisibilidad

Una función muy importante de los cheats es poder aumentar la distancia de la cámara. Así, el jugador no tiene que mover tanto la cámara para poder ver qué es lo que está pasando en sus alrededores y le permite ver más información y poder reaccionar más rápido en casos de ataque.

8.3.4. Uso automático de objetos

Una opción del cheat es usar algunos objetos de forma totalmente automática en casos especiales. Un caso podría ser, por ejemplo, usar un objeto para curarse cuándo el personaje recibe mucho daño y su vida baja hasta el 20 %. Otro caso de uso automático serían objetos que activan un efecto durante un tiempo que no cuesta maná, como es el caso del objeto Phase Boots, que le permite al personaje incrementar su velocidad durante 4 segundos cada 8 segundos. Se puede delegar el control de este objeto al cheat para no tener que pulsar la tecla manualmente.

8.3.5. Auto-disable

Un tipo de cheat muy potente se llama "Auto-disable" y sirve para poder neutralizar a un personaje enemigo en el caso de un ataque inesperado o si este personaje quiere usar alguna habilidad importante. El cheat usa una habilidad del personaje o un objeto escogido para poder neutralizar al enemigo, ya sea inmovilizarlo, atarlo o aplicarle el efecto de silencio que impide usar habilidades. El caso más común es el efecto llamado "Hex", que pueden lanzar algunos personajes. Consiste en que el personaje afectado no puede usar habilidades ni objetos y su velocidad decrece casi al 100 % durante 3 segundos. El uso de este

cheat es bastante peligroso porque es fácilmente detectable y es muy odiado por los jugadores.

8.3.6. Auto-save

Este tipo de cheat es parecido al anterior, pero su objetivo principal no es neutralizar al personaje enemigo, sino esquivar sus habilidades y lo puede hacer de distintas formas. En caso de que la habilidad del enemigo es dirigida a un punto, el cheat simplemente mueve al personaje para esquivar la habilidad. En otros casos, cuando la habilidad es dirigida al personaje, se mira si hay alguna forma de esquivar con una habilidad o un objeto. Si es así, entonces se usa esta habilidad u objeto en el momento correcto para que la habilidad enemiga no afecte al personaje. Igual que el cheat anterior, también se detecta fácilmente.

8.3.7. Burro mensajero

Cuando un personaje se compra un objeto, el objeto aparece en la base del equipo del personaje. Para recoger el objeto, el jugador puede ir a la base o puede llamar a una entidad llamada burro mensajero para que este se lo traiga. Esta entidad es también controlada por el jugador y tiene dos habilidades temporales: el sprint y la invulnerabilidad. Ya que el jugador no puede siempre estar pendiente del burro, puede delegar al cheat el uso de estas dos habilidades, concentrándose más en su personaje principal.

8.3.8. Macros para héroes concretos

En este apartado se explican dos casos de scripts para héroes concretos: en primer lugar, uso automático de los objetos y habilidades de un personaje al presionar un botón y, en segundo lugar, control completo de un personaje y sus ilusiones.

Skywrath Mage es un héroe de daño mágico cuya funcionalidad principal es hacer mucho daño mágico a un héroe durante pocos segundos. El personaje tiene 4 habilidades activas que producen daño mágico y los objetos que se suelen comprar para este héroe incrementan su potencial mágico y en total para hacer una combinación completa se puede llegar a tener que presionar diez botones. El script apunta automáticamente a un jugador enemigo y usa todos los objetos y habilidades simultáneamente produciendo el máximo de daño posible.

Arc Warden es un héroe de daño principalmente físico cuya habilidad principal consiste en crear una ilusión especial que puede usar las habilidades y los objetos del héroe. El script controla el movimiento de la ilusión, sus habilidades y objetos como si lo hubiera hecho el jugador. De esta forma el jugador solamente tiene que controlar al personaje principal y dejar que el cheat se ocupe de la ilusión. Como el programa ejecuta las acciones mucho más rápido que el jugador, produce un impacto enorme en la parte final de la partida, cuándo el personaje se hace poderoso porque tiene muchos objetos. En el script se puede elegir qué habilidades y objetos puede usar la ilusión.

8.3.9. Overwolf

Overwolf es una herramienta que permite saber varias estadísticas de los jugadores que hay en una partida sin tener que mirar sus cuentas. Como muchas estadísticas de los jugadores, como sus personajes favoritos o sus partidas, son públicas y accesibles mediante páginas como Dotabuff, Overwolf busca los perfiles de los jugadores por su id y simplemente visualiza la información que encuentra.

The screenshot shows the Overwolf Dota Plus interface with a dark blue theme. It displays statistics for a team of players, organized into three main sections: 'PLAYER - GENERAL', 'PLAYER - THIS MONTH', and 'HERO - THIS MONTH'. Each player's row includes their profile picture, name, rank, general match statistics (Matches, With Me, Win/Loss), role, lane, most played heroes, and hero-specific statistics for the current month (Hero, Matches, Win/Loss, Lane). A notification at the top right states 'DotaPlus has been updated! Click here to find out what's new.'

PLAYER - GENERAL				PLAYER - THIS MONTH				HERO - THIS MONTH			
Player	Rank	Matches	With Me	Matches	Role	Lane	Most Played	Hero	Matches	Role	Lane
@cmkcmkml	185	4455	53%	58	82%	70%	6 1 5 3	1	100%	100%	
Antiboboman	39	1301	52%	11	90%	63%	3 1 2 2	No Stats			
Knock knock, let the...	39	4305	52%	112	100%	73%	17 16 11	No Stats			
xd	238	1959	54%	95	56%	44%	17 8 8	1	100%		
;-)	203	1711	56%	11	100%	81%	2 1 1	1	0%	100%	100%
Endollia dum dum	?	3229	56%	15	73%	93%	4 2 2	No Stats			
Xin Rei	?	5093	53%	15	86%	45%	3 2 1	2	50%	100%	50%
Varizh	?	24	50%	15	73%	46%	5 3 1	No Stats			
mentally unstable	777	3572	57%	102	51%	91%	12 12 12	12	33%	100%	67%
聽見下雨的聲音	161	3043	54%	136	49%	89%	18 11 10	3	0%	100%	100%

Figura 33: Pantalla de Overwolf

¿Qué se puede ver en la pantalla de Overwolf de la Figura 33 por cada jugador?

- El rango
- La cantidad y el porcentaje de partidas ganadas en general
- La cantidad y el porcentaje de partidas ganadas el último mes
- El rol y la línea que suele escoger el jugador
- Los personajes más jugados en el último mes
- El personaje escogido en la partida más reciente y sus características

Con esta información se obtiene una ventaja respecto a los jugadores enemigos. Como se puede saber quien irá a cada línea y cuáles son sus personajes favoritos, se puede plantear una estrategia óptima y escoger los personajes que mejor funcionarían en cada caso.

Sin embargo, un jugador puede prohibir publicar sus estadísticas en público y en este caso su fila tendrá bastante menos información.

9. Resumen de cheats

Usando la información de los capítulos de Clasificación y los ejemplos, se puede hacer un mapa conceptual para resumir las formas de interacción con los juegos y relacionar cada ejemplo con un tipo de cheat. El resultado se ve en la Figura 34.

Los métodos que tienen varios ejemplos no están subrayados, mientras que los métodos finales o ejemplos están subrayados con un color. Si se trata de un método de cheat para los juegos de un solo jugador, el subrayado es azul. Si se trata de un método para los juegos multijugador, el subrayado es verde. En caso de un método universal, el subrayado es amarillo.

Existen 4 modos de interacción con el juego:

- Macros externos al juego
- Uso de los datos del juego
- Uso de los bugs del juego
- Cuentas secundarias

Los bugs y las cuentas secundarias son métodos finales debido a que no se pueden separar en más categorías. Los datos del juego pueden usarse por tres vías: memoria RAM, disco duro o paquetes.

El disco duro lo usan dos tipos de cheats orientados a editar la información del disco duro: edición de ficheros y edición de partidas guardadas.

Luego hay otra subcategoría que engloba la memoria y los paquetes. La mayoría de los cheats, como el aimbot o el triggerbot, pueden funcionar por ambas vías, aunque lo más desarrollado es usar la memoria. Los trainers solamente funcionan por memoria, debido a que los juegos de un solo jugador generalmente no son juegos online y toda la información la guardan en la memoria RAM.

Por otro lado, se encuentra la categoría de macros externos, que no manipulan datos del juego, sino que son combinaciones de pulsaciones del ratón y el teclado. Contiene dos ejemplos de cheats (bunnyhop y control de retroceso). Sin embargo, ambos tipos de cheats pueden funcionar también por la memoria y los paquetes.

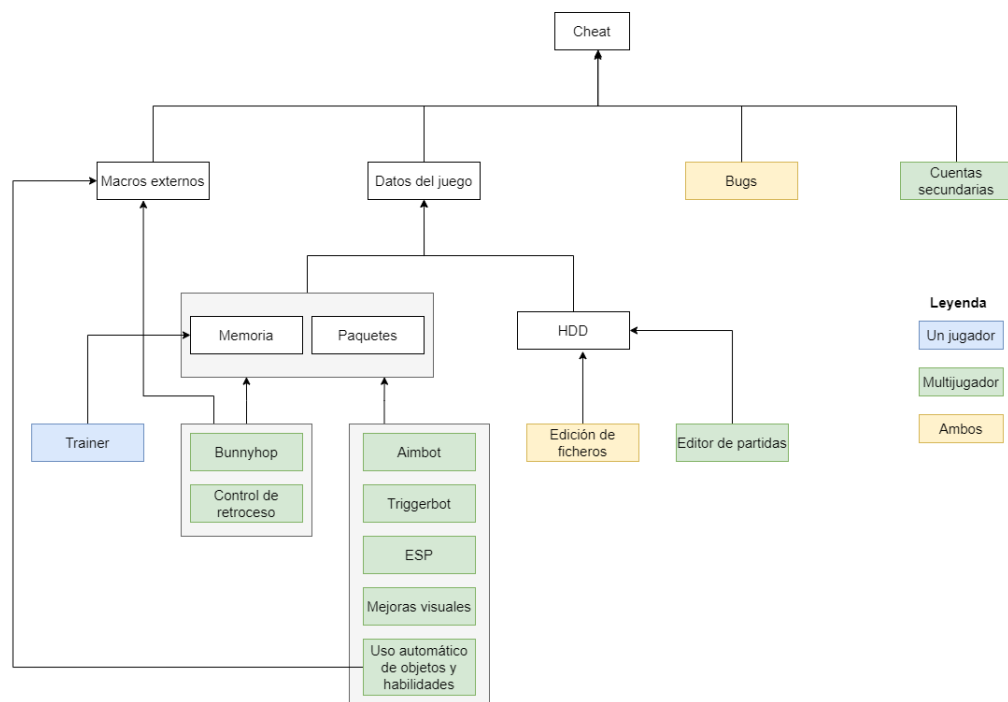


Figura 34: Clasificación de cheats y ejemplos

10. Anticheats

Un anticheat es un programa o un sistema que permite detectar jugadores que usan cheats.[3] La mayoría de los anticheats solamente funcionan en los juegos online, porque muchas veces necesitan conexión a internet para poder detectar cheaters. Existen dos tipos de anticheats: unos se encuentran en la parte del servidor, y los otros se encuentran en la parte del cliente. Los anticheats en el servidor trabajan más con los paquetes, detectando las posibles anomalías, y los anticheats en el cliente trabajan con el hardware del cliente, revisando los datos del juego y detectando acciones sospechosas del jugador, como podría ser ejecutar un cheat.

10.1. Anticheats en el servidor

Previamente, se ha explicado cómo el cliente puede manipular el tráfico y editar la información de los paquetes que se le envían al servidor. Es muy importante que el servidor no se fíe de los datos que recibe por parte del cliente y revise si el contenido llegado es posible. Por ejemplo, si el jugador produce una acción y se le envía un paquete con esta acción al servidor, este tiene que comprobar si esta acción de verdad ha podido ocurrir. Se tendría que comprobar que el jugador ha estado en condiciones para ejecutar esta acción. Para disminuir la cantidad de comprobaciones, se puede disminuir la información que envía el cliente moviendo una parte de los cálculos al lado del servidor. Así, el jugador tiene menos posibilidades de editar paquetes, pero aumenta la carga en el servidor. Tiene que haber un equilibrio entre las acciones que se producen en el servidor y las acciones que se producen en el cliente, para que no haya sobrecarga en ninguna de las partes ni tampoco haya una sobrecarga de tráfico.

Para que al interceptar paquetes no se pueda conseguir información, se puede ofuscar el tráfico. De esta forma, el servidor envía paquetes cifrados que solamente puede entender el cliente. Aunque igualmente sea posible interceptar paquetes, no se puede sacar la información de estos. Para incrementar la seguridad, se puede ir cambiando de algoritmo de cifrado para disminuir las posibilidades de que una persona encuentre la llave correcta para descifrar el contenido. Este método puede disminuir el rendimiento del juego debido a que se necesita un tiempo adicional para poder cifrar y descifrar paquetes en ambos lados.

Otro método de anticheat en el servidor son los métodos estadísticos. En los juegos online se guardan las características de los jugadores. Por ejemplo, en el juego PUBG: BATTLEGROUNDS se guardan el número de bajas, el número de asistencias, la distancia recorrida, el daño total de cada partida. Teniendo una base con estos datos, se puede hacer comparaciones para encontrar jugadores cuyas estadísticas se distinguen mucho de lo normal en su rango. Al haber encontrado un jugador así, no se puede aplicarle directamente restricciones debido a que no siempre un jugador con estadísticas raras es un cheater. Simplemente, puede ser un jugador con mucha experiencia o un jugador que ha tenido mucha

suerte en una partida. Entonces lo que se suele hacer es marcar a este jugador como sospechoso y luego hacer una revisión humana, para poder decidir si este jugador es un cheater o no. En el caso de Dota 2 se usa un sistema llamado Overwatch.[23] Cuando un jugador es denunciado muchas veces en una partida, esta partida se guarda y posteriormente se reparte entre varios otros jugadores enseñándoles momento en los que el jugador ha sido denunciado. Los revisores pueden ver esos momentos desde la perspectiva del jugador denunciado y pueden decidir si este ha usado cheats o no. Si el jugador es calificado por muchos revisores como cheater, pasa a ser calificado como cheater y se le aplica alguna medida restrictiva.

10.2. Anticheats en el cliente

Los anticheats que se encuentran en el ordenador del cliente se ocupan principalmente de verificar que todos los archivos y la memoria RAM del juego no han sido modificados. El primer método anticheat consiste en ofuscar el código fuente del juego. Un cheater puede usar un programa para descubrir el código fuente (decompilador) de un archivo para ver cómo funciona internamente alguna funcionalidad del juego. Con este conocimiento, el cheater puede descubrir cómo eludir una restricción o encontrar algún bug en el juego. Si el código fuente aparece ofuscado, el decompilador no podrá enseñar un resultado aceptable y no proporcionará información al cheater. Este método tiene sus inconvenientes, ya que el código fuente ofuscado se hace más dependiente de la plataforma y del compilador y no puede ser depurado.

Para asegurarse de que los archivos del juego no han sido modificados por el jugador, el anticheat puede verificar su código hash. Este método sirve para detener situaciones en las que un jugador puede cambiar ciertos parámetros del juego para hacerlo más simple, como por ejemplo disminuir el retroceso de un arma o desactivar un efecto visual como la granada de humo. En algunos juegos, después de iniciarse, se ejecuta un proceso que verifica los códigos hash de los archivos, enviándolos al servidor y comparándolos con los códigos guardados. En otros juegos, el código hash de un fichero solamente se calcula cuando el fichero se carga en la memoria RAM. Si el código de un fichero no coincide, el juego no deja entrar en la partida. Para modificar un fichero y no ser detectado, un cheater tiene que buscar un modo de cambiar el código hash del fichero de tal forma que coincida con el código del fichero original. También puede editar el paquete enviando el código hash original en vez del código hash real.

Otro método para detectar cheaters consiste en detectar programas de cheats conocidos. Para que esto sea posible, existen bases de datos de programas de cheats y el funcionamiento del anticheat solo es hacer un recorrido de los procesos ejecutados y verificar si alguno de estos coincide con algún proceso de la base de datos de cheats. Para detectar programas, el anticheat puede hacer uso de los nombres de sus ventanas, nombres de sus procesos, códigos hash de sus ficheros. Para un cheater es muy fácil evitar la detección de su programa porque puede usar nombres y códigos hash aleatorios, haciendo que la búsqueda nunca

encuentre el cheat. Este es el método de anticheat menos eficaz de todos.

Para poder complicar el escaneo de la memoria RAM de un proceso existe el método de mover las variables siempre y cuando cambia su valor. De esta forma se pretende hacer imposible encontrar direcciones correctas de variables usando programas como Cheat Engine, y como consecuencia, protegerlas de los posibles cambios hechos por el cheater. Para no influir mucho en el rendimiento del dispositivo, solamente hay que emplear esta técnica con las variables más importantes, como por ejemplo los recursos del jugador o las coordenadas de los enemigos, ya el proceso de cambiar la dirección de una variable consume recursos y según el cálculo que ha hecho Samuli Lehtonen en su trabajo de fin de máster, puede llegar a ser 40 veces más lento tener que mover una variable de sitio al cambiar su valor.

La mayoría de los programas anticheats en el cliente se ejecutan en el espacio de usuario, limitando sus posibilidades de controlar lo que está pasando en el ordenador. Muchos de los anticheats actuales trabajan en el espacio kernel. Esto significa que tienen el acceso ilimitado a la memoria, al disco duro, a todas las instrucciones de la CPU y a los datos críticos del sistema operativo. De esta forma, un anticheat por kernel tiene el control completo de todas las acciones que se están produciendo en el dispositivo y, obviamente, tiene un poder de detección mayor. Por ejemplo, puede ver si el usuario está intentando inyectar un fichero DLL en el proceso del juego o si el usuario pretende editar un fichero del juego. Si detecta una acción no deseada, puede enviar un informe al servidor y mientras tanto cerrar el juego, para que el usuario no pueda seguir jugando. Como consecuencia del aumento de los anticheats por kernel, se ha comenzado a desarrollar cheats que también funcionan en el espacio del kernel, contrarrestando estos anticheats.

Actualmente, muchos de los juegos nuevos usan anticheats que funcionan en el espacio del kernel. Por ejemplo, Easy Anti-Cheat es uno de los anticheats más usados actualmente y está presente en los juegos más populares como Apex Legends, Rust o Lost Ark.[31]

11. Medidas adoptadas por las empresas

11.1. Introducción

Con el aumento masivo de la producción de cheats, las empresas de videojuegos han sido obligadas a desarrollar medidas para combatir a los cheaters. Todos los derechos y todas las prohibiciones en un juego se regulan mediante el Acuerdo de Licencia del Usuario Final (se suele usar la sigla EULA), un contrato entre el propietario del juego (empresa) y el usuario final (jugador).[24] En este contrato se encuentran los puntos que regulan el uso del software externo y el uso intencional de los bugs en el juego. También se definen las medidas que se aplican a jugadores que han violado el Acuerdo. Cada juego se distribuye bajo su propio EULA y las empresas pueden escoger por cada juego qué es lo que pueden hacer los jugadores y qué es lo que no pueden hacer, pero la mayor parte de empresas prohíben el uso de cheats o herramientas para ganar ventajas en su EULA.

11.2. Medidas restrictivas

Las medidas que se les aplican a los jugadores pueden ser distintas dependiendo de la norma violada, del juego y de la empresa.

La medida menos restrictiva de todas es echar a un jugador de la partida. En muchos casos el jugador puede volver a entrar al servidor. Esta medida suele aplicarle en los casos más leves de cheats y también en los casos cuando el jugador viola alguna norma de la comunidad, ya sea abusar del chat o hacer alguna acción mal vista, como matar a un compañero del equipo. En el caso del juego Counter-Strike: Global Offensive, existe la herramienta llamada votekick, que permite votar dentro del equipo para echar a un jugador.[32]. En el caso de los juegos de la serie de Battlefield, el servidor puede echar a un jugador por asesinar más de 3 compañeros de su equipo, matar enemigos mientras se encuentran en su base o también por usar armas prohibidas en el servidor.

La medida clásica contra los cheaters es el bloqueo, ya sea temporal o permanente. El bloqueo puede verse de distintas formas dependiendo del juego.

En algunos casos el bloqueo de un jugador no permite hacer ninguna acción, enseñando un aviso al iniciar el juego, como por ejemplo el caso de PUBG: BATTLEGROUNDS. El bloqueo en este juego puede ser temporal (entre 3 y 30 días) o permanente.[25]

En otros juegos la medida solamente se aplica a partidas protegidas con otros jugadores, mientras que el jugador aún puede jugar contra el ordenador, crear partidas locales o jugar en servidores no protegidos. Este es el caso de los juegos de la empresa Valve.[26] De hecho, si un jugador es bloqueado por el anticheat de Valve (VAC) en un juego, se bloquea también en otros juegos protegidos por el mismo anticheat.

En la mayoría de los juegos, los cheaters se bloquean permanentemente. Como el número de cheaters aumenta, los juegos que no tenían medidas contra los

cheaters, ahora están creando medidas más restrictivas. Por ejemplo, en el Dota 2 desde el año 2019 se bloquea a los jugadores denunciados muchas veces durante 20 años.[33] Otra medida que se ha implantado en muchos juegos es el bloqueo por el identificador del hardware (HWID). Los juegos guardan los identificadores de los componentes de los ordenadores de los jugadores en el servidor (por ejemplo, el procesador). Al bloquear a un jugador, se marcan sus identificadores y si el jugador crea otra cuenta y abre el juego desde el mismo ordenador, el juego verá que su HWID está marcado y, por lo tanto, volverá a bloquear al jugador, haciendo este método muy eficaz.

Para disminuir el uso de cuentas secundarias, en algunos juegos se aplica una medida especial llamada shadowban. Este tipo de bloqueo implica que el jugador que usa una cuenta secundaria, solamente puede jugar con otros jugadores con cuentas secundarias, bloqueando la posibilidad de jugar con la mayoría de los jugadores que juegan desde su cuenta principal. En el juego Call of Duty: Warzone, no se ha publicado ninguna documentación oficial sobre el shadowban, sin embargo, algunos jugadores afirman que han sufrido este tipo de restricción.[34]

En algunos juegos se puede anular el progreso del personaje del jugador. En el juego de Grand Theft Auto V, se aplica esta medida para castigar a los jugadores que usan cheats para añadir dinero a su personaje o subir de nivel.[35]

11.3. Estado legal del uso de cheats

Por norma general, en la mayoría de los países el uso de cheats es legal si el cheater solamente está modificando los datos en su ordenador y no está sacando provecho monetario a partir de la actividad (por ejemplo, vendiendo el cheat).[36] Si un cheater pretende atacar o dañar un servidor, sí que se considera ilegal. De esta forma, cualquier modificación de datos o de paquetes en el ordenador del cliente sigue siendo legal. Pero, no es legal, por ejemplo, hacer un ataque DDoS para crear lags en el servidor.

Aun así, ha habido casos en los que los cheaters fueron multados. Por ejemplo, en el año 2018 un jugador de Corea del Sur fue multado con 10.000\$ por haber usado cheats en el juego Overwatch.[28] En el año 2017 en Corea del Sur entró en vigor una ley que prohíbe ciertos cheats en los juegos. Un cheater puede ser multado con hasta 50.000\$ o ser condenado a hasta cinco años de prisión.[29]

Otro ejemplo de país en el que usan cheats puede ser ilegal es China. Desde que ha salido a la venta el juego de PUBG: BATTLEGROUNDS, las autoridades han podido detener muchos cheaters. En el agosto del año 2022 se detuvo a 141 personas en total acusadas de haber usado cheats en el juego PUBG Mobile (versión del juego para dispositivos móviles). Para hacerlo, la empresa Tencent (creadora de ambos juegos) ha cooperado con la policía de China para poder detener a estos jugadores en varias ciudades del país.[27]

11.4. Estado legal del desarrollo y la venta de cheats

El desarrollo de cheats y su posterior venta sí que se consideran ilegales en muchos países, debido a que su creación y distribución viola las normas del derecho de autor. Es ilegal hacer una versión modificada de un juego sin el permiso de la entidad que posee los derechos de autor. Una violación más es distribuir el juego modificado.

En el año 2017, la empresa Riot Games presentó una demanda contra LeagueSharp, un servicio que ofrecía cheats para el juego de League of Legends, acusándolo de haber violado la Ley de Derechos de Autor de la Era Digital (DMCA).[37] El tribunal obligó a LeagueSharp pagar una compensación de 10.000.000\$ y dejar el control de sus páginas web a Riot Games.

Otro caso parecido es el caso de la demanda de la empresa de Epic Games contra un cheater menor de edad acusado de la infracción de derechos de autor por la venta de cheats para el juego de Fortnite. El cheater fue obligado a eliminar su versión modificada del juego, borrar todos los datos de sus cheats y también tuvo que eliminar los videos de su canal de YouTube en el cual publicaba videos jugando al Fortnite con el cheat.[38][39]

Los países asiáticos como Corea del Sur o China son los países que tienen las medidas legales más fuertes contra la creación y distribución de los cheats. En el julio del año 2021, las autoridades de China detuvieron a tres personas acusadas de haber desarrollado cheats para el juego de Genshin Impact.[40] Este grupo ganó alrededor de 300.000\$ vendiendo sus cheats. Los tres fueron condenados a entre un año y medio y cuatro años de prisión. Mientras tanto, en Corea del Sur en el año 2018 condenaron a un año de prisión a una persona por haber usado y vendido un cheat para el juego de Overwatch, habiendo ganado 180.000\$ de su venta.[41]

Referencias

- [1] *A brief history of cheats*. URL: <https://www.gamesradar.com/a-brief-history-of-cheats/>. Última visita: 2022-08-29.
- [2] *Cheating in video games*. URL: https://en.wikipedia.org/wiki/Cheating_in_video_games. Última visita: 2022-08-29.
- [3] Samuli Johannes Lehtonen. *Comparative Study of Anti-cheat Methods in Video Games*. Helsingin Yliopisto, 2020.
- [4] *Dying Light Solo Duplication Glitch After 1.09 Update! (Player Stash)*. URL: <https://www.se7ensins.com/forums/threads/dying-light-solo-duplication-glitch-after-1-09-update-player-stash.1464974/>. Última visita: 2022-08-14.
- [5] *Can I get banned for using bugs in the game?* URL: <https://support.pubg.com/hc/en-us/articles/115004191514-Can-I-get-banned-for-using-bugs-in-the-game->. Última visita: 2022-08-14.
- [6] *Cheat Engine*. URL: <https://www.cheatengine.org/>. Última visita: 2022-08-14.
- [7] *Hazedumper*. URL: <https://github.com/frk1/hazedumper>. Última visita: 2022-08-15.
- [8] *Console Commands (Skyrim)*. URL: [https://elderscrolls.fandom.com/wiki/Console_Commands_\(Skyrim\)](https://elderscrolls.fandom.com/wiki/Console_Commands_(Skyrim)). Última visita: 2022-09-05.
- [9] *FALLOUT 4: TRAINER +20 V1.0 - 1.10.26*. URL: https://gtrainers.com/load/categories/trainers/fallout_4_trainer_20_v1_0_1_10_26_fling/28-1-0-4470. Última visita: 2022-08-02.
- [10] *Mass Effect 3 Save Editor*. URL: <https://www.nexusmods.com/masseffect3/mods/695>. Última visita: 2022-08-02.
- [11] ifvrt12. *Un poco sobre las trampas en los juegos de disparos: qué son las trampas*. URL: <https://habr.com/ru/post/401813/>. Última visita: 2022-09-01.
- [12] *Aimbot Game Hacks / Auto Aim Bots / Headshot Hacks & Aim Assist Cheats explained*. URL: <https://hackerbot.net/wiki/42-aimbot>. Última visita: 2022-09-01.
- [13] {C#} *Writing a simple AIMBOT cheat using CS Source as an example*. URL: <https://lolz.guru/threads/3646207/>. Última visita: 2022-07-22.
- [14] *What are ESP Hacks?* URL: <https://wallhax.com/what-are-esp-cheats/>. Última visita: 2022-09-01.
- [15] Snaacky. *Diamond*. URL: <https://github.com/davidgfb/Diamond>. Última visita: 2022-07-21.
- [16] Snaacky. *Sapphire*. URL: <https://github.com/Snaacky/Sapphire>. Última visita: 2022-07-22.
- [17] *Strafing (video games)*. URL: [https://en.wikipedia.org/wiki/Strafing_\(video_games\)](https://en.wikipedia.org/wiki/Strafing_(video_games)). Última visita: 2022-09-01.
- [18] *Bunnyhop*. URL: <https://yougame.biz/threads/63116/>. Última visita: 2022-07-21.
- [19] Snaacky. *Bunnyhop*. URL: <https://github.com/Snaacky/Ruby>. Última visita: 2022-07-21.

- [20] WackyJacky101. *GUIDE: HOW TO CONTROL YOUR RECOIL in PUBG*. URL: https://www.youtube.com/watch?v=R_b7OuFkWYQ. Última visita: 2022-07-15.
- [21] Shivam338. *NoRecoil*. URL: <https://github.com/Shivam338/NoRecoil>. Última visita: 2022-07-22.
- [22] Snaacky. *Emerald*. URL: <https://github.com/Snaacky/Emerald>. Última visita: 2022-07-22.
- [23] *Overwatch llega a Dota*. URL: <https://www.dota2.com/newsentry/3025824821114909461?l=spanish>. Última visita: 2022-09-01.
- [24] *¿Qué es un EULA y cómo puedes generar uno?* URL: <https://www.iubenda.com/es/help/43961-que-es-un-eula-y-como-puedes-generar-uno>. Última visita: 2022-08-17.
- [25] *I want to know more about bans*. URL: <https://support.pubg.com/hc/en-us/articles/8463403858329-I-want-to-know-more-about-bans>. Última visita: 2022-08-31.
- [26] *Valve Anti-Cheat (VAC) System*. URL: <https://help.steampowered.com/en/faqs/view/571A-97DA-70E9-FF74>. Última visita: 2022-08-31.
- [27] Raphael Bastek. *141 hackers arrested in China after police bust of PUBG cheat software*. URL: <https://upcomer.com/141-hackers-arrested-in-china-after-police-bust-of-pubg-cheat-software>. Última visita: 2022-08-25.
- [28] *Korean Overwatch Hackers Arrested, Hit With \$10,000 Fine*. URL: <https://kotaku.com/korean-overwatch-hackers-arrested-hit-with-10-000-fin-1825701300>. Última visita: 2022-08-17.
- [29] Brian Ashcraft. *Game Hacks Are Illegal In South Korea*. URL: <https://kotaku.com/game-hacks-are-illegal-in-south-korea-1795917608>. Última visita: 2022-08-20.
- [30] *Videojuego multijugador de arena de batalla en línea*. URL: https://es.wikipedia.org/wiki/Videojuego_multijugador_de_arena_de_batalla_en_l%C3%ADnea. Última visita: 2022-07-22.
- [31] *Every game with kernel-level anti-cheat software*. URL: <https://levvvvel.com/games-with-kernel-level-anti-cheat-software/>. Última visita: 2022-08-17.
- [32] *Votekick*. URL: <https://cso.fandom.com/wiki/Votekick>. Última visita: 2022-08-25.
- [33] Scott Robertson. *Dota 2 bug causes players to be hit with 20 year matchmaking bans*. URL: <https://www.dexerto.com/dota2/dota-2-bug-causes-players-to-be-hit-with-20-year-matchmaking-bans-1095686/>. Última visita: 2022-08-31.
- [34] Jenny Zheng. *GTA Online Cheaters' Accounts Reset For Using Money Exploit*. URL: <https://www.gamespot.com/articles/gta-online-cheaters-accounts-reset-for-using-money/1100-6481632/>. Última visita: 2022-08-31.
- [35] Diego Emegé. *Te contamos cómo saber si has sido víctima de shadow ban en Call of Duty Warzone*. URL: <https://www.3djuegospc.com/guias->

- y-trucos/te-contamos-como-saber-has-sido-victima-shadow-ban-warzone. Última visita: 2022-08-31.
- [36] *What Video Game Hacks are legal? — Legality of Game Hacks explained.* URL: <https://hackerbot.net/wiki/51-game-hack-legal>. Última visita: 2022-08-17.
- [37] J. Fingas. *'League of Legends' creator wins \$10 million in cheating lawsuit.* URL: <https://www.engadget.com/2017-03-06-league-of-legends-creator-wins-cheating-lawsuit.html>. Última visita: 2022-08-21.
- [38] Blake Brittain. *Epic Games Settles Copyright Claims Against 'Fortnite' Hacker.* URL: <https://news.bloomberglaw.com/ip-law/epic-games-settles-copyright-claims-against-fortnite-hacker>. Última visita: 2022-08-22.
- [39] *¿Es ilegal hacer trampas en un videojuego? La particular batalla de Fortnite.* URL: <https://www.mundodeportivo.com/urbantecno/videojuegos/cheats-fortnite-ilegal>. Última visita: 2022-08-22.
- [40] *Three People Jailed for Illegally Producing and Selling Cheating Programs for Genshin Impact.* URL: <https://pandaily.com/three-people-jailed-for-illegally-producing-and-selling-cheating-programs-for-genshin-impact/>. Última visita: 2022-08-25.
- [41] Taylor Hatmaker. *An Overwatch hacker in South Korea just got sentenced to a year in prison.* URL: <https://techcrunch.com/2018/06/25/overwatch-hacker-seoul-jail-time/>. Última visita: 2022-08-25.