Víctor Cribeiro Pérez
Manuel Fernández Ferreiro
Víctor González Del Campo
Diego Moure González

SSI Trabajo tutelado

Ordenado por prioridades

Keyloggers

Un **keylogger** (abreviatura de «*keystroke logging*») es un tipo de *spyware* que registra todas las pulsaciones de teclas que realice en el ordenador. Pueden ser tanto basados en *hardware* como en *software*.

Son usados tanto para fines legítimos como ilegítimos, la legitimidad de su uso depende del permiso de la parte que es vigilada, si son utilizados con consentimiento se considerarán legítimos. Por ejemplo, los emplean los hackers para robar información sensible como contraseñas o códigos PIN o en otros casos pueden ser utilizados por empresas para controlar a sus trabajadores.

Este tipo de *spyware* habitualmente guarda la información que registra por teclado en un fichero de *log* que se encripta y se guarda en el propio sistema que está espiando para después enviarlo a un *remote server* o a un *email*. En el caso de los **keyloggers** basados en *hardware* la información puede quedar almacenada en el equipo hasta que se recupere el dispositivo.

Tanto los **keyloggers** basados en *hardware* como los basados en *software* no son fáciles de detectar, los primeros pueden estar integrados en un USB conectado al equipo, ubicados como una membrana en el teclado, modificando el *firmware* del equipo o en la propia conexión entre el teclado y el equipo. Dentro de la categoría de *hardware* se incluyen varios tipos de **keyloggers** con una metodología más sofisticada, que no requieren de conexión física directa, como pueden ser **keyloggers** *bluetooth* o para teclados inalámbricos, que capturan la paquetería enviada entre el teclado y el equipo e intentan descifrar o *crackear* la conexión segura que tienen establecida.

Los **keyloggers** basados en *software* más básicos pueden ser contrarrestados mediante un *anti-malware* o un *firewall*, pero también pueden actuar a nivel de *kernel*, que no son tan sencillos de detectar y eliminar. En general, la mayoría de estos se incluyen en un fichero infectado que es el que los ejecuta y les permite acceso al equipo desde donde trabajarán y enviarán la información al atacante. El envío de la información se puede realizar mediante algún servicio web como un correo o una subida a un servidor.

Nos centraremos en detallar el funcionamiento de este tipo de *spyware* en sus dos posibles métodos, así como sus usos más comunes con mayor profundidad que la explicada en este resumen.

Víctor Cribeiro Pérez
Manuel Fernández Ferreiro
Víctor González Del Campo
Diego Moure González

Anti-cheat

Un **anti-cheat** es una herramienta que se encarga de evitar el uso de programas externos dentro de aplicaciones, generalmente en videojuegos. Nosotros nos centraremos estos y sus distintos tipos a lo largo de la historia de los videojuegos, concretando más en algunos usados actualmente como son: **VAC** (*Valve Anti-Cheat*), ***EasyAnti-Cheat*** o ***Riot Vanguard***.

Para hablar de **anti-cheats**, primero vamos a profundizar un poco en lo que son los *cheats*, sus diferentes tipos y en cómo afectan a lo que sería el transcurso normal de un videojuego.

Los *cheats*, popularmente conocidos como "trampas" o "*hacks*", se pueden categorizar de diferente manera dependiendo de cómo se aprovechan de las vulnerabilidades del videojuego. A grandes rasgos se pueden dividir en ***Soft Cheats*** y ***Hard Cheats***.

Los ***Soft Cheats*** son métodos que se aprovechan de mecánicas jugables para ganar una ventaja injusta sobre otros jugadores. Estos pueden ser mecánicas no pensadas por los desarrolladores, como por ejemplo maneras de ganar dinero rápidamente dentro del juego. Estos suelen estar cubiertos por el **EULA** (*End-User License Agreements*) que en los juegos suele contener una cláusula que prohíbe el uso fraudulento de mecánicas *"unintended"*. Este tipo de *cheats* suelen ser parcheados con actualizaciones una vez son descubiertos y reportados. Los bugs y *exploits* podrían también considerarse ***Soft Cheats***.

Los ***Hard Cheats*** son el tipo de *cheat* que nos interesan a nosotros, pues son los que suelen ser tratados de contrarrestar por los ***Anti-Cheats***. Generalmente cuando hablamos de *cheats* nos referimos a este tipo pues son más llamativos. Por ejemplo, un tipo de *hard cheat* puede ser el uso de un programa que edite el cliente del juego o modifique paquetes que este cliente envía hacia el servidor. Los programas externos también pueden introducirse dentro del espacio de memoria del juego y crear nuevas funcionalidades dentro de este. Una herramienta muy conocida para este tipo de *cheats* es el ***Cheat Engine***, que es bastante sencilla y se suele utilizar para juegos un jugador, su principal funcionalidad es cambiar valores dentro de la memoria del juego. Dentro de los ***Hard Cheats*** también podríamos hablar del uso de *bots* y herramientas automatizadas, usar datos del juego que deberían estar ocultos para el jugador, modificación de paquetes o *spoofing*.

Ahora que ya hemos hablado un poco de los *cheats* podemos comenzar con las distintas formas de contrarrestarlos. Los ***anti-cheat*** pueden ser separados en dos categorías: ***server-side*** y ***client-side***.

Server-sided anti-cheats, trabajan únicamente dentro del servidor y se basan en la comprobación de los paquetes entrantes y asegurarse de que el estado de los datos del juego es correctamente manejado en el servidor. Algunos funcionamientos de estos son: No confiar en el cliente (verificación de archivos), diseño de un protocolo de aplicación resistente (generalmente se utiliza un tipo **UDP** que incluye funcionalidades de **TCP**), utilización de métodos estadísticos para descubrir *cheaters* (analizando información como el número de victorias).

Client-side methods, son programas que trabajan en la máquina del cliente y mandan información de esta hacia el servidor. Ejemplo de este tipo podría ser el **VAC** mencionado con anterioridad. Algunos de estos métodos son: encriptación de código, verificación de archivos mediante *hashing*, detección de programas de *cheats* ya conocidos, ofuscación de memoria y **anti-cheats** basados en *kernel* (como es el caso de **Riot Vanguard**) . Algunos tipos de *cheat* (como por ejemplo el *wallhack*) son complicados de detectar utilizando únicamente **anti-cheats server-sided**, por eso hoy en día los del cliente son muy relevantes.

Víctor Cribeiro Pérez
Manuel Fernández Ferreiro
Víctor González Del Campo
Diego Moure González

Víctor Cribeiro Pérez
Manuel Fernández Ferreiro
Víctor González Del Campo
Diego Moure González

Bibliografía

Keyloggers

https://www.avast.com/es-es/c-keylogger

https://ieeexplore.ieee.org/abstract/document/9814059 -> Secciones 5 y 7

https://www.ijraset.com/research-paper/windows-post-exploitation-msf-keylogger

Anti-cheats

https://helda.helsinki.fi/bitstream/handle/10138/313587/Anti_cheat_for_video_games_final_07_03_2020.pdf?sequence=2&isAllowed=y

https://github.com/adrianyy/EACReversing <- EasyAntiCheat code

https://github.com/danielkrupinski/VAC <- VAC code

https://github.com/RiotVanguard/Vanguard <- Riot Vanguard code