МЕХАНИЗМ РАБОТЫ С ХУКАМИ НА СИСТЕМНОМ УРОВНЕ В ОС WINDOWS

MECHANISM FOR WORKING WITH HOOKS AT THE SYSTEM LEVEL IN OS WINDOWS

УДК 004.453.2

Баев Д.А., студент 3 курс, факультет «Информатика и вычислительная техника» Донской государственный технический университет Россия, г. Ростов-на-Дону

Baev, D. A., e-mail: fantom-sj@yandex.ru

Аннотация

Статья посвящена обзору механизма работы с хуками в операционной системе Windows, с функциями Windows API. В данной статье рассматриваются принципы, по которым строится работа с хуками на системном уровне. А именно каким образом осуществляется перехват обработки событий, генерируемых в ОС. Также рассматривается организация в памяти процесса ссылок, на вызываемые в процессе его работы, функции Windows API.

Summary

The article is devoted to an overview of the mechanism for working with hooks in the Windows operating system, with Windows API functions. This article discusses the principles by which work with hooks is built at the system level. Namely, how the processing of events generated in the OS is intercepted. The organization in memory of the process of references to the Windows API functions called during its operation is also considered.

Ключевые слова: хуки, Windows, API, система, функции, процессы.

Key words: hooks, Windows, API, system, functions, processes.

На сегодняшний день существует множество способов обнаружения и борьбы с вредоносным программным обеспечением. Одним из таких способов является механизм проверки поведения программы, путём перехвата её работы, что позволяет отследить её поведение в операционной системе [1].

Несмотря на то, что на сегодняшний день существует множество инструментов перехвата работы программ на системном уровне, в операционной системе Windows предусмотрен стандартный способ организации подобного перехвата [2].

Данный механизм получил название «хуки», произошедшего от английского слова «hook», что переводится как «ловушка». В ОС Windows предусмотрен набор инструментов, к которым относятся специальные функции, структуры и сообщения хук точек, для реализации механизма хуков [2]. Все эти элементы содержатся в заголовочном файле winuser.h, который хранит в себе и другие элементы управления Windows.

Хуки в ОС Windows устроены следующим образом. Когда разработчик создаёт хук, операционная система создаёт в памяти структуру данных, которая содержит всю информацию о созданном хуке. При этом, созданная структура добавляется в связанный список ранее созданных хуков. В этом списке новый хук добавляется перед всеми ранее созданными хуками. И затем, когда случается событие, то в случае с локальным хуком вызывается фильтрующая функция внутри процесса [3]. В случае с глобальным хуком, системе необходимо вставить код хук функции в адресное пространство того процесса, для которого был установлен данный хук. Для этого система подключает к этому процессу динамическую библиотеку функций и подменяет оригинальную функцию на хук-функцию [3].

При запуске каждого процесса в оперативной памяти выстраивается таблица функций, где хранится информация о тех функциях, которые ему необходимы в работе. Она имеет название Import Address Table или сокращённо IAT, что переводится как таблица импорта адресов [1]. В этой таблице хранится для каждой вызываемой функции, хранится запись с именем динамической

библиотеки DLL, в которой можно найти данную функцию и её адрес в этой библиотеке [1].

Таким образом можно представить схематично как будет выглядеть запуск функции из библиотеки некоторым процессом. Даная схема представлена на рисунке 1.

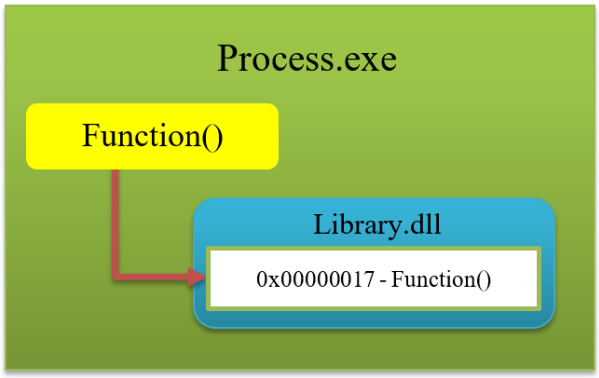


Рисунок 1 – Схема запуска оригинальной функции

Установка хука происходит через модификацию таблицы IAT. Данный способ заключается в том, что при установке хука происходит изменение оригинального адреса функции API в таблице IAT на адрес хук-функции, которая уже с свою очередь может запустить оригинальную для поддержания функционирования программы [1]. Схематичное изображение данного процесса показано на рисунке 2.

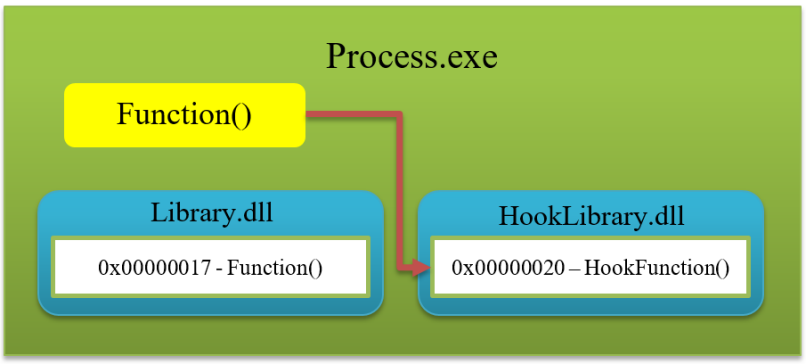


Рисунок 2 – Схема запуска хук-функции

Таким образом, при установлении хука происходит подмена указателя на вызываемую функцию из таблицы функций, созданной процессом при его запуске.

На практике установкой хука занимается функция SetWindowsHookEx(). Данная функция устанавливает один из определённых типов хуков, описанных

ранее, на перехват соответствующего ему тип системных событий [3]. Функция SetWindowsHookEx() принимает на вход 4 параметра, именно:

- int idHook – является типом хука, который необходимо установить;
- HOOKPROC lpfn – адрес процедуры, которая будет запущена если данный хук перехватит управление.
- HINSTANCE hMod – дескриптор динамической библиотеки, в которой находится процедура, запускаемая установленным хуком.
- DWORD dwThreadId – идентификатор потока, на который на который устанавливается хук.

Также можно деинсталлировать хук, вызвав функцию UnhookWindowsHookEx(), которая в качестве параметра принимает лишь одно значение, это дескриптор хука, который необходимо удалить [3].

В заключении стоит отметить, что это лишь основные функции для работы с хуками в ОС Windows, но тем не менее понимания принципа их работы и работы в целом всего механизма позволяет ознакомиться с одним из методов борьбы с вредоносным программным обеспечением, а также приблизиться к пониманию того, как работают процессы в операционной системе Windows и какие существуют способы для перехвата вызова функций внутри них, непосредственно во время их выполнения.

Использованные источники:

1. Syed Z.M. Mohd A.M. In memory detection of Windows API call hooking technique [Электронный ресурс] // ResearchGate: профессиональная сеть для ученых и исследователей. URL: https://www.researchgate.net/publication/283480063_In_memory_detection_of_Windows_API_call_hooking_technique (дата обращения 11.07.2020)
2. Hooks [Электронный ресурс] // MSDN: Центр разработки для Windows. URL: https://docs.microsoft.com/en-us/windows/win32/winmsg/hooks (дата обращения: 11.07.2020)

3. Using Hooks [Электронный ресурс] // MSDN: Центр разработки для Windows. URL: https://docs.microsoft.com/en-us/windows/win32/winmsg/using-hooks (дата обращения: 12.07.2020)