MICROSOFT DETOURS - ИНСТРУМЕНТ ПЕРЕХВАТА ВЫЗОВА ФУНКЦИЙ API В ОС WINDOWS

MICROSOFT DETOURS - API CALL INTERCEPT TOOL IN WINDOWS OS

УДК 004.453.2

Баев Дмитрий Александрович, студент кафедры кибербезопасности информационных систем, Донской государственный технический университет, г. Ростов-на-Дону

Baev D.A. fantom-sj@yandex.ru

Аннотация

В статье рассматривается инструмент работы с хуками для перехвата вызова функций из WinAPI на ОС Windows, такой как Microsoft Detours. Также рассмотрены принципы, по которым происходит осуществление перехвата, как это происходит на уровне деассемблированного кода и какие методы применяются для установки хуков. По мимо этого рассмотрено как именно устроена библиотека Microsoft Detours, даны базовые определения, используемые в данной библиотеки. Также рассмотрена схема перехвата функции при срабатывании установленного хука, и был произведён разбор самых основных функций, их входных параметров, а также результата их работы.

Annotation

This article discusses a tool for working with hooks to intercept function calls from WinAPI on Windows OS, such as Microsoft Detours. The principles by which interception is carried out, how it happens at the level of disassembled code, and

what methods are used to install hooks are also considered. In addition, it is considered how exactly the Microsoft Detours library is arranged, and the basic definitions used in this library are given. In addition, a scheme for intercepting a function when an installed hook is triggered was considered, and the most basic functions, their input parameters, and the result of their work were analyzed.

Ключевые слова: хуки, WinAPI, Windows, Microsoft Detours, функции, процессы.

Keywords: Hooks, WinAPI, Windows, Microsoft Detours, functions, processes.

Одном из механизмов изменения работы уже скомпилированной программы, является инструмент установки хуков в ОС Windows. Он позволяет перехватить работу программы на системном уровне и при этом внести изменения в её поведения. Слово «хук» с английского языка переводится как «ловушка» и пишется соответственно «hook».

Для более удобного и безлопастного способа работать с хуками в операционной системе Windows, 16 января 2002 года компания Microsoft выпусти такое решение как Microsoft Detours [1].

На данный момент существует 2 версии данной библиотеки, а именно:

1. Detours Express;
2. Detours Professional Edition.

Основное их отличие заключается в том, что Detours Express распространяется по бесплатной лицензии MIT, а Detours Professional Edition является платной, но при этом её можно использовать в коммерческих целях [1].

Также среди важных отличий между двумя версиями Microsoft Detours является то, что профессиональное издание Detours поддерживает все процессоры на которых работает ОС Windows, а также поддерживает 64-битный код на процессорах x64 и IA64 [1].

Среди общих положительных качеств библиотеки стоит отметить то, что она довольно проста в изучении, крайне эффективна, под неё написана хорошая документация и не требует знания ассемблера. Помимо этого, ввиду того что данная библиотека была разработана компанией Microsoft, она хорошо оптимизирована для работы в ОС Windows по сравнению с другими библиотеками.

Для понимания того как происходит процесс перехвата, стоит разобрать определения нескольких видов функция, участвующих в этом процессе, а именно:

Планируемая функция – это функция, вызов которой осуществляется при нормальном порядке работы программного обеспечения и перехватывается в случае установки хука на её перехват [2].

Исходная функция – это та функция, выполнение которой осуществляется непосредственно перед вызовом из неё планируемой функции [2].

Хук-функция – это функция, вызов которой осуществляется вместо планируемой функции [2].

В Detours код перехвата применяется динамически во время его выполнения и заменяет первые несколько инструкций планируемой функции, на безусловный переход к хук-функции. Инструкции, которые заменяются в планируемой функции, сохраняются в буфере. Помимо них в буфер также помещается безусловный переход к оставшейся части планируемой функции [3].

Когда выполнение достигает подменяемой планируемой функции, управление переходит непосредственно к пользовательской хук-функции. Хук-функция выполняет всю необходимую предварительную обработку перехвата. При этом она может вернуть управление исходной функции, или перейти к командам из буфера, из которых затем управление передаётся в оставшуюся часть планируемой функции. Когда эта функция завершается, она возвращает управление хук-функции, которая в свою очередь выполняет

соответствующую постобработку и возвращает управление обратно в исходную функцию [3]. На рисунке 1 схематично показан логический поток управления для вызова функции с перехватом.



Рисунок 1 – Контроль потока вызова функции с перехватом.

Библиотека Detours перехватывает планируемые функции, переписывая их на уровне двоичного кода внутри процесса. Для каждой планируемой функции Detours фактически создаёт две функции: новую планируемую функцию и функцию, в которой хранятся команды, помещённые в буфер и один указатель на хук функцию. Функция команд буфера создаётся динамически. До установки хука, буфер содержит только одну инструкцию, а именно команду перехода к планируемой функции. А после её создания, в буфер добавляются заменённые инструкции из начальной планируемой функции и переход к оставшейся её части [3].

При создании хука, как уже было ранее сказано, инициализируется указатель на планируемую функцию. После того, как хук присоединен к планируемой функции, указатель изменяется, чтобы указывать на функцию команд, записанных в буфер. После того, как хук отсоединяется от планируемой функции, указатель снова меняется так, чтобы указывать на исходную планируемую функцию [3].

Для перехвата планируемой функции Detours, в первую очередь выделяет память для функции с перезаписываемыми командами в буфере. После чего копирует все необходимые инструкции из планируемой функции в выделенную память. Данная процедура продолжается до тех пор, пока не

будет скопировано не менее 5 байт, достаточных для записи инструкции безусловного перехода. В случае если планируемая функция имеет размер менее 5 байт, Detours прерывает запись и возвращает код данной ошибки [3]. На рисунке 2 демонстрируется процесс изменения деассемблированного кода при установке хука на вызов планируемой функции.

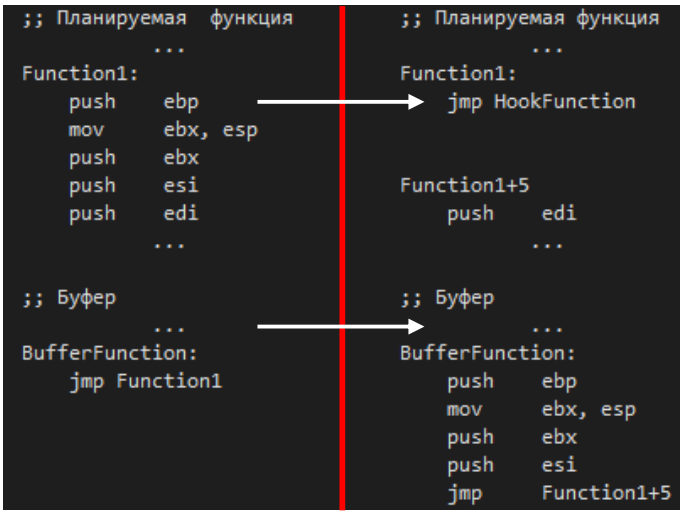

Рисунок 2 – Изменение планируемой функции и буфера при установке хука.

Для правильного перехвата планируемой функция, хук-функция и указатель на функцию команд в буфере должны иметь одинаковую сигнатуру вызова, включая количество аргументов и соглашение о вызове. Использование одного и того же соглашения о вызове гарантирует, что регистры будут должным образом сохранены и что стек будет должным образом выровнен между хук и планируемой функциями [4].

Для того чтобы использования библиотеку Detours необходимо подключить к проекту библиотеку detours.lib, а также уже непосредственно в коде подключить заголовочный файл detours.h [4].

В Detours установка и снятие хука осуществляется с помощью транзакций. Для того чтобы открыть транзакцию и зафиксировать её существуют такие функции, как DetourTransactionBegin() и DetourTransactionCommit(). Данные функции не принимают ни каких

параметров на входе, при этом в случае успешного выполнения они возвращают параметр NO_ERROR, или параметр ошибки в случае наличия таковой [4].

Перехват планируемой к запуску функции из исходной осуществляется с помощью вызова функции DetourAttach(). Её вызов должен осуществляться после открытия транзакции и до её фиксации, т.е. между DetourTransactionBegin() и DetourTransactionCommit() [4]. Данная функция принимает на вход 2 аргумента и имеет следующий прототип:

```
LONG DetourAttach(
    _Inout_ PVOID *ppPointer,
    _In_    PVOID pDetour
);
```

– _Inout_ PVOID *ppPointer – указатель на планируемую функцию;

– _In_ PVOID pDetour – указатель на хук-функцию.

Данная функция также в случае успешного вызова возвращает параметр NO_ERROR, при этом в случае возникновения ошибки, она возвращает соответствующей ей параметр.

Перед тем как непосредственно установить хук, необходимо указать Detours на то, что поток программы, в котором он устанавливается необходимо обновить. При этом, если не сделать этого, то есть вероятность, что данный поток может попытаться запустить недопустимую комбинацию старого и нового кода, что несомненно приведёт к ошибке [4].

Для того чтобы это сделать необходимо сразу после функции создания транзакции DetourTransactionBegin(), внутри неё сразу вызвать функцию DetourUpdateThread(), которая на вход принимает всего одно значение, а именно дескриптор потока, который ожидает данную транзакцию .

Далее уже в DetourAttach() происходит подготовка буфера и выделение памяти для него. После этого, когда хук функция установлена и транзакция фиксируется, планируемая функция и буфер переписываются, а указатель на

планируемую функцию изменяется с оригинальной функцию на функцию запуска сохранённых инструкций в буфере [4].

При попытке вызвать планируемую функцию из исходной, её вызов перехватывается и перенаправляется на хук-функцию, откуда в случае необходимости она может быть запущена. Ответственность за верное копирование аргументов при вызове планируемой функции через функцию с инструкциями из буфера лежит на хук-функции. Это обусловлено тем, что планируемая функция становится просто подпрограммой, вызываемой хук-функцией [4].

Перехват планируемой функции удаляется путем вызова функции DetourDetach() в новой транзакции. Также как и DetourAttach(), DetourDetach() принимает два аргумента: указатель на планируемую функцию, с вызова которой необходимо снять перехват и указатель на саму хук-функцию. Когда транзакция снятия хука фиксируется, планируемая функция переписывается и восстанавливается в исходном виде, функция с командами в буфере удаляется, а указатель на планируемую функцию восстанавливается, чтобы указывать на исходную планируемую функцию [4].

Таким образом данной статье был рассмотрен принципы работы с хуками, с помощью библиотеки Microsoft Detours. Было показано как именно происходит перехват запуска функция с помощью инструментов данной библиотеки. При этом было показано, что библиотека Detours является эффективным, удобным и безопасным средством, позволяющим устанавливать хуки в среде ОС Windows. Помимо этого был рассмотрен механизм работы с набором основных инструментов входящих в неё и принципы, по которым происходит перехват планируемой к запуску функции.

Литература

1. Detours / Microsoft Research URL: https://www.microsoft.com/en-us/research/project/detours/ (дата обращения: 13.10.2020)

2. Гален Хант, Дэвид Тардити, Брайан Джанфоркаро. Обзор пакета Microsoft Research Detours // Энциклопедия GitHub [Microsoft 2018] URL: https://github.com/microsoft/detours/wiki (дата обращения: 14.10.2020)

3. Гален Хант, Брайан Джанфоркаро. Перехват бинарных функций // Энциклопедия GitHub [Microsoft 2018] URL: https://github.com/microsoft/detours/wiki/OverviewInterception (дата обращения: 14.10.2020)

4. Дэвид Тардити, Брайан Джанфоркаро. Использование Detours // Энциклопедия GitHub [Microsoft 2019] URL: https://github.com/microsoft/detours/wiki/Using-Detours (дата обращения: 15.10.2020)

Literature

1. Detours / Microsoft Research URL: https://www.microsoft.com/en-us/research/project/detours/ (date of appeal: 13.10.2020)

2. Galen Hunt, David Tarditi, Brian Gianforcaro. Microsoft Research Detours Package Overview // Encyclopedia GitHub [Microsoft 2018] URL: https://github.com/microsoft/detours/wiki (date of appeal: 14.10.2020)

3. Galen Hunt, Brian Gianforcaro. Interception of Binary Functions // Encyclopedia GitHub [Microsoft 2018] URL: https://github.com/microsoft/detours/wiki/OverviewInterception (date of appeal: 14.10.2020)

4. David Tarditi, Brian Gianforcaro. Using Detours // Encyclopedia GitHub [Microsoft 2019] URL: https://github.com/microsoft/detours/wiki/Using-Detours (date of appeal: 15.10.2020)