УДК 004.056.5

Крайнова Татьяна Сергеевна

студент, Санкт-Петербургский государственный университет телекоммуникаций им. проф. М. А. Бонч-Бруевича, Россия, Санкт-Петербург

ASLR И DEP: ПОЧЕМУ ЭКСПЛОЙТ 2005 ГОДА НЕ РАБОТАЕТ В WINDOWS 11

Аннотация

В статье рассматриваются ASLR и DEP как ключевые механизмы защиты памяти в Windows 11. Показано, почему старый эксплойт, рассчитанный на предсказуемые адреса библиотек и исполняемый стек, теряет работоспособность в современной операционной системе. Раскрывается принцип запрета выполнения кода из стека и кучи, а также объясняется рандомизация адресного пространства процесса. Сделан вывод, что ASLR и DEP не устраняют ошибки программирования, но значительно усложняют практическую эксплуатацию уязвимостей памяти.

Annotation

The article examines ASLR and DEP as key memory-protection mechanisms in Windows 11. It explains why an old exploit based on predictable library addresses and an executable stack becomes unreliable in a modern operating system. The paper describes the prohibition of code execution from stack and heap memory and explains address-space randomization. It concludes that ASLR and DEP do not remove programming errors, but they significantly complicate practical exploitation of memory-corruption vulnerabilities.

Ключевые слова: информационная безопасность, ASLR, DEP, Windows 11, переполнение буфера, защита памяти.

Keywords: information security, ASLR, DEP, Windows 11, buffer overflow, memory protection.

Введение

Тематика статьи относится к области информационной безопасности, а более точно — к защите операционных систем от эксплуатации уязвимостей памяти. Под «эксплойтом 2005 года» в данной работе понимается типичный для того периода сценарий: программа содержит переполнение буфера, атакующий записывает в память полезный машинный код, затем изменяет адрес возврата или указатель выполнения так, чтобы процессор перешел к этому коду. Такая схема была особенно опасной, когда расположение библиотек было предсказуемым, а стек мог использоваться не только для хранения данных, но и для выполнения инструкций процессора. Значение анализа кода и тестирования защищенности отмечается в работах [1; 2].

Windows 11 устроена иначе. Система не предполагает, что все приложения написаны без ошибок. Напротив, она использует дополнительные барьеры, которые должны помешать превращению ошибки в выполнение произвольного кода. Поэтому старая уязвимость может формально сохраняться в программе, но готовый эксплойт, написанный под старую среду, перестает работать надежно. Главную роль в этом играют два механизма: DEP и ASLR.

1. Механизм DEP: запрет исполнения данных

Data Execution Prevention, или DEP, разделяет память процесса на исполняемые и неисполняемые области. Код программы и подключенных библиотек должен находиться на страницах, где разрешено исполнение. Стек, куча и обычные буферы предназначены для хранения данных, поэтому они не должны использоваться как место запуска инструкций процессора. Если управление передается на страницу, помеченную как неисполняемая, возникает исключение доступа, и процесс обычно завершается [3].

Для старого эксплойта это критично. Раньше атакующий мог записать shellcode в буфер на стеке, после чего изменить адрес возврата и передать управление прямо на этот буфер. При включенном DEP такая передача управления не дает нужного результата: байты полезной нагрузки присутствуют в памяти, но процессор не рассматривает их как допустимый

код. Следовательно, простая схема «записать код в стек и перейти на него» в Windows 11 уже не является рабочей.

2. Механизм ASLR: нарушение предсказуемости адресов

Address Space Layout Randomization снижает предсказуемость виртуального адресного пространства процесса. При каждом запуске система может размещать исполняемый файл, динамические библиотеки, стек, кучу и другие выделения по различным адресам. Для поддержки такой загрузки приложение должно быть собрано с соответствующими параметрами, например с /DYNAMICBASE [4]. Из-за этого атакующему становится сложнее использовать заранее подготовленный адрес перехода.

Именно поэтому эксплойт 2005 года теряет устойчивость. В старом коде часто встречался заранее выбранный адрес: например, адрес инструкции перехода в DLL или адрес начала буфера. При включенной рандомизации такой адрес становится ненадежным. Библиотека может оказаться в другом месте, стек может начинаться по другому адресу, а переход по старой константе приведет к аварийному завершению процесса. Для обхода ASLR атакующему нужна дополнительная информация, например утечка адреса из памяти процесса.

3. Совместная работа ASLR и DEP в Windows 11

Наибольший эффект дают не отдельные механизмы, а их сочетание. DEP запрещает запуск внедренного кода из стека или кучи, а ASLR мешает заранее найти готовые исполняемые фрагменты в библиотеках. Если у атакующего есть только переполнение буфера, но нет точной карты памяти, управление трудно направить в нужную точку. Если адрес найден, но полезная нагрузка лежит в неисполняемой области, запуск также блокируется. Поэтому современная эксплуатация превращается в цепочку обходов, а не в один простой переход.

Windows 11 использует и другие средства Exploit Protection: принудительную рандомизацию образов, bottom-up ASLR, проверки целостности кучи, защиту от некоторых вариантов ROP и контроль потока

выполнения [5]. Эти функции не заменяют обновления и безопасное программирование, но меняют экономику атаки: для успеха требуется не одна ошибка, а целая цепочка обходов.

4. Практическое значение для защиты и разработки

Для разработчика вывод состоит в том, что защита должна включаться еще на этапе компиляции и настройки приложения. Программа должна поддерживать ASLR, не отключать DEP без необходимости, корректно работать с границами буферов и использовать современные средства компилятора. Статический анализ помогает обнаруживать часть ошибок до эксплуатации [1].

Для анализа безопасности важно понимать и ограниченность таких средств. ASLR может быть ослаблен утечкой адресов, а DEP может обходиться техниками повторного использования уже существующего кода. Однако защита остается полезной: она повышает стоимость атаки, снижает надежность старых эксплойтов и заставляет учитывать конкретную версию системы. Практическая проверка защищенности должна сочетать анализ кода и тестирование на проникновение [2].

Заключение

Эксплойт 2005 года не работает в Windows 11 не потому, что исчезли переполнения буфера, а потому, что изменилась среда эксплуатации. DEP делает стек и кучу непригодными для прямого запуска внедренного кода. ASLR разрушает надежность фиксированных адресов библиотек и областей памяти. Вместе эти механизмы переводят атаку из заранее подготовленного сценария в сложную цепочку обходов. Поэтому современная защита Windows строится не только на исправлении уязвимостей, но и на снижении вероятности успешного использования тех ошибок, которые все же остаются в программном обеспечении.

Список литературы

1. Бударный Г. С., Пестов И. Е., Штеренберг И. Г. Сравнение методов статического анализа исходного кода программы // Вестник СПбГУТД. Серия 1. 2024. № 1. С. 5-12. DOI 10.46418/2079-8199_2024_1_1.

2. Цветков А. Ю., Рузманов Е. Ю. Рассмотрение методов тестирования на проникновение для анализа защищенности компании // Приоритетные направления инновационной деятельности в промышленности. Казань: КОНВЕРТ, 2021. Ч. 2. С. 57-58.

3. Microsoft Learn. Data Execution Prevention. Документация Microsoft Win32. 2023.

4. Microsoft Learn. /DYNAMICBASE: Use address space layout randomization. Документация Microsoft Visual C++. 2022.

5. Microsoft Learn. Exploit protection reference. Документация Microsoft Defender for Endpoint. 2025.

References

1. Budarny G. S., Pestov I. E., Shterenberg I. G. Comparison of static code analysis methods // Bulletin of SPbSUTD. Series 1. 2024. No. 1. P. 5-12. DOI 10.46418/2079-8199_2024_1_1.

2. Tsvetkov A. Yu., Ruzmanov E. Yu. Penetration testing methods for company security analysis // Priority Directions of Innovation Activity in Industry. Kazan: KONVERT, 2021. Part 2. P. 57-58.

3. Microsoft Learn. Data Execution Prevention. Microsoft Win32 documentation. 2023.

4. Microsoft Learn. /DYNAMICBASE: Use address space layout randomization. Microsoft Visual C++ documentation. 2022.

5. Microsoft Learn. Exploit protection reference. Microsoft Defender for Endpoint documentation. 2025.