**Научно-исследовательская часть**

1. **. Постановка задачи проектирования**

Задача курсовой работы состоит в программной реализации технологии внедрения DLL библиотеки в целевой процесс «Reflective DLL Injection», которая позволяет улучшить функционал уже существующего программного обеспечения. А также необходимо разработать удобный и понятный интерфейс.

1. **. Описание предметной области**

Предметной областью являются технологии внедрения программного кода в процессы. Рефлексивная DLL инъекция дает возможность выполнять свой код в адресном пространстве уже запущенного процесса. Такое внедрение часто используются внешними программами, чтобы повлиять на поведение другой программы так, как её авторы не задумывали и не предполагали. Например, внедрённый код может перехватывать системные вызовы функций или прочитать содержимое текстовых полей пароля, что невозможно сделать обычным способом. Но данный прием не обязательно применять в плохих целях и, например рефлексивное внедрение может применяться для обновления своего приложения или для его усовершенствования.

Внедрение кода используется для выполнения всевозможных функций в Windows. Эту технологию используют легитимные программы, а также используют вредоносные программы. Например:

Антивирусные программы часто внедряют код в веб-браузеры. Они могут использовать его, например, для отслеживания сетевого трафика и блокировки опасного веб-контента.

Вредоносные программы могут добавлять код в ваш веб-браузер, чтобы лучше отслеживать ваши действия в Интернете, красть защищенную информацию, такую как пароли и номера кредитных карт, а также изменять настройки браузера.

WindowBlinds Stardock, который задает темы для рабочего стола, внедряет код в изменить способ отображения окон .

AutoHotkey, который позволяет создавать сценарии и назначать им общесистемные горячие клавиши , внедряет код для этого.

Графический драйвер подобен встроенным DLL от NVIDIA для выполнения множества задач, связанных с графикой.

Некоторые программы внедряют библиотеки DLL для добавления дополнительных пунктов меню в приложение.

Инструменты для мошенничества в компьютерных играх часто вводят в игры код, чтобы изменить их поведение и получить несправедливое преимущество перед другими игроками.

Рефлексивное внедрение происходит по этапам:

1. Некоторый исполняемый файл считывает DLL-библиотеку с диска в адресное пространство своего процесса и передает управление на ее экспортируемую функцию ReflectiveLoader.

2. Поскольку теперь библиотека существует в произвольном месте в памяти, ReflectiveLoader вычисляет текущее местоположение самой DLL-библиотеки в памяти. Для этого ReflectiveLoader получает адрес текущей инструкции и, двигаясь в обратном направлении, ищет байты 4D5A, соответствующие MZ-сигнатуре. Это нужно, чтобы дать библиотеке возможность анализировать свои собственные заголовки для дальнейшего запуска.

3. ReflectiveLoader определяет адрес библиотеки kernel32.dll в текущем процессе, после чего анализирует таблицу ее экспорта и находит функции LoadLibrary, GetProcAddress и VirtualAlloc, необходимые для дальнейшей загрузки.

4. ReflectiveLoader выделяет непрерывный участок памяти (VirtualAlloc), где и размещает код DLL-библиотеки (заголовки и секции) в соответствии с виртуальными адресами. Затем ReflectiveLoader обрабатывает свою вновь загруженную таблицу импорта: загружает необходимые библиотеки (LoadLibrary) и импортируемые из них функции (GetProcAddress).

5. ReflectiveLoader вызывает DllEntryPoint внедряемой библиотеки.

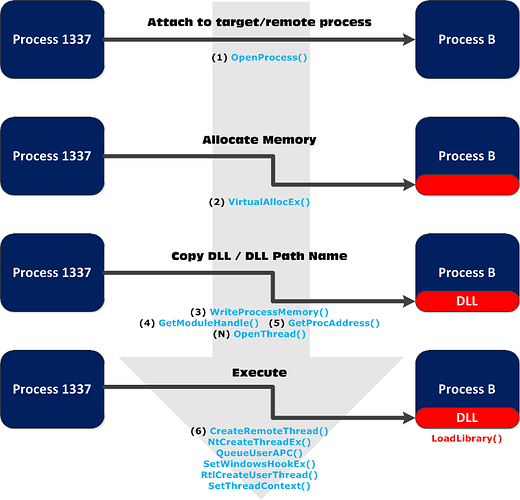


Рис. 1 – Как происходит рефлексивное внедрение в целевой процесс.

1. **. Анализ аналогов и прототипов**

Существует большое количество методов внедрения такие как:

* + 1. **Injection DLL**

Injection DLL - это один из наиболее распространенных методов внедрения в целевой процесс. Перед внедрением вредоносной программе потребуется копия вредоносной DLL, уже сохраненная в системе жертвы. Такая технология имеет преимущество, такое что в DLL библиотеке храниться все методы и функции, тем самым можно вызывать эти функции из внедряемого процесса. Такое внедрение происходит по этапам:

1. Программа вызывает функцию Windows API (OpenProcess) для подключения к процессу

2: Часть памяти выделяется в процессе с помощью VirtualAllocEx. Эта память выделяется с использованием доступа «для записи».

3. Программа должна найти адрес функции LoadLibrary в пространстве процесса.

4: Программа вызывает функцию CreateRemoteThread, создавая новый поток в процессе, передавая адрес LoadLibrary, найденный на шаге 3. Когда загружается вредоносная DLL, вызывается метод записи DLL, DLLMain. Здесь будут происходить все действия.

1. **Injection PE**

Portable Execution (PE) - это формат файла Windows для исполняемого кода. Это структура данных, содержащая всю необходимую информацию, чтобы Windows знала, как ее выполнять.

PE-injection - это метод, при котором программа внедряет PE-образ в уже запущенный процесс. Преимущество этого метода перед внедрением DLL состоит в том, что это бездисковая операция, то есть программе не нужно записывать свои полезные данные на диск перед внедрением. Внедрение происходит так же по этапам:

1. Программа получает базовый адрес и размер процесса-жертвы.

2. Программа выделяет достаточно памяти в процессе жертвы для вставки своего PE-образа.

3. Поскольку вставленное изображение будет иметь другой базовый адрес после его внедрения в затронутый процесс, программе необходимо сначала найти смещение таблицы перемещения процесса-жертвы. С этим смещением программа изменит изображение так, чтобы любые абсолютные адреса в изображении указывали на нужные функции. После обновления PE-образа программа копирует его в процесс.

4. Программа ищет функцию входа для выполнения и запускает ее с помощью CreateRemoteThread.

1. **Process Hollowing**

Process Hollowing- это метод, при котором Программа запускает процесс, и заменяет код процесса на свой. Преимущество этого метода заключается в том, что программа становится независимым от того, что в данный момент запущено в системе у пользователя. Более того, запустив системный процесс (например, Блокнот), пользователи не будут видеть исполняемый код во внедряемом процессе. Пример того как происходит внедрение:

1. Вредоносная программа создает новый процесс, например «Блокнот», но инструктирует Windows создать его как приостановленный процесс. Это означает, что новый процесс не запустится.

2. Программа делает процесс пустым, отключая связанные с ним области памяти.

3. Программа выделяет память для собственного кода и копирует его в пространство памяти процесса. Затем он вызывает SetThreadContext для процесса-пользователя, который изменяет контекст выполнения процесса на только что созданный.

4. Программа возобновляет процесс; тем самым выполняя свой код.

1. **Hook injection**

Hook injection это метод который использует ловушку WH\_GETMESSAGE , он устанавливаем процесс, который будет следить за сообщениями, обрабатываемыми системными окнами. Чтобы установить ловушку, мы вызываем функцию SetWindowsHookEx.

WH\_GETMESSAGE аргумент определяет тип сообщения, а functionAddress определяет адрес функции (в адресном пространстве процесса) , программа должна сообщить, когда система пытается обработать сообщение. Таким образом, этот метод можно применить для перехвата определенного процесса или всех процессов в системе. Пример работы hook injection

1. Поток application.exe собирается отправить сообщение в какое-то окно.

2. Система проверяет, установлен ли для этого потока ловушка WH\_GETMESSAGE.

3. Затем система выясняет, отображается ли Inject.dll, которая содержит обратный вызов сообщений.

4. Если Inject.dll еще не сопоставлен, система сопоставляет его с адресным пространством процесса application.exe и увеличивает счетчик блокировок библиотеки DLL в этом процессе.

5.Функция DllMain Inject.dll вызывается с параметром DLL\_PROCESS\_ATTACH .

6. Затем в адресном пространстве процесса application.exe вызывается обратный вызов.

**Преимущества выбранной технологии.**

Reflective DLL Injection позволяет внедрить код DLL-библиотеки в процесс из памяти. Основное преимущество такого подхода заключается в том, что библиотека не регистрируется в системе. В результате ее практически невозможно обнаружить ни на уровне системы, ни на уровне процесса.

1. **. Перечень задач подлежащих решению в процессе разработки**
2. разработка пользовательского интерфейса;
3. реализация технологии Reflective DLL injection с помощью WinAPI на языке С++;
4. реализация DLL библиотеки, с вызовом MessageBox;
5. Запись заголовков DLL в память.
6. Тестирование и проверка разработанной программы на виртуальной машине.
7. Вывод всплывающих окон с информацией о внедрении
8. **. Обоснование выбора инструментов и платформы для разработки**

Для выполнения курсовой работы были выбраны:

**1. Операционная система Windows 10.**

В операционной системе Windows реализована объектно-ориентированная идеология. Базовый объект системы – окно, поведение которого определяется методом, называемым функцией окна. Графический образ окна на экране дисплея – прямоугольная рабочая область. Независимо от своего типа любой объект Windows идентифицируется описателем или дескриптором (handle). Все взаимоотношения программного кода с объектом осуществляются только через его дескриптор. Интерфейс прикладного программирования (API – Application Programming Interface) представляет собой совокупность 32-битных функций (Win32 API), которые предназначены для создания приложений (программ), работающих под управлением Microsoft Windows. Функции объявлены в заголовочных файлах. Главный из них − файл windows.h, в котором содержатся ссылки на другие заголовочные файлы.

API — это аббревиатура названия Application Programming Interface (интерфейс прикладного программирования). API представляет собой совокупность функций и инструментов, позволяющих программисту создавать приложения (программы), работающие в некоторой среде.

Win32 API — это набор функций для создания программ, работающих под управлением Microsoft Windows 98, Windows NT или Windows 2000. Все функции этого набора являются 32 битными, что отражено в названии интерфейса

Причины выбора Операционной системы Windows 10:

1. Простота эксплуатации. Даже неподготовленному человеку легко разобраться в принципах работы операционной системы, ведь все интуитивно понятно, также система не требует знаний в программировании.

2. Большое количество программ сторонних компаний выпускают именно под данную операционную систему. Есть как платные, так и бесплатные версии.

3. Большая часть выпускаемого оборудования для ПК выпускается с поддержкой ОС Windows. Веб-камеры, сканеры, принтеры, игровые манипуляторы и так далее. Все драйвера, в первую очередь, выходят под данную ОС, и лишь по прошествии большого промежутка времени выходят под другие операционные системы.

4. Большая распространённость. Данная операционная система, по состоянию на апрель 2020 года, заняла второе место в общей мировой статистике использования ОС.

5. Различные формы и виды оформления. Кроме стандартных вариантов ОС позволяет сторонним программам вносить изменения во внешний вид рабочего стола, папок, заставок и так далее.

6. Можно осуществлять полное взаимодействие с ОС с помощью Windows API и его компонентами

**3. Язык программирования C++**

C++ — компилируемый, статически типизированный язык программирования общего назначения. Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником — языком C — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C. Одними из отличительных особенностей и очень важных преимуществ C++ является:

1. Поддерживаются различные стили и технологии программирования, включая традиционное директивное программирование, ООП, обобщённое программирование, метапрограммирование (шаблоны, макросы).

2. Предсказуемое выполнение программ является важным достоинством для построения систем реального времени. Весь код, неявно генерируемый компилятором для реализации языковых возможностей (например, при преобразовании переменной к другому типу), определён в стандарте. Также строго определены места программы, в которых этот код выполняется. Это даёт возможность замерять или рассчитывать время реакции программы на внешнее событие.

3. Автоматический вызов деструкторов объектов при их уничтожении, причём в порядке, обратном вызову конструкторов. Это упрощает (достаточно объявить переменную) и делает более надёжным освобождение ресурсов (память, файлы, семафоры и т. п.), а также позволяет гарантированно выполнять переходы состояний программы, не обязательно связанные с освобождением ресурсов (например, запись в журнал).

4. Пользовательские функции-операторы позволяют кратко и ёмко записывать выражения над пользовательскими типами в естественной алгебраической форме.

5. Windows API представляет собой множество функций, структур данных и числовых констант, следующих соглашениям языка Си. Все языки программирования, способные вызывать такие функции и оперировать такими типами данных в программах, исполняемых в среде Windows, могут пользоваться этим API. В частности, это язык C++

**3. Среда разработки Microsoft Visual Studio 2019**

Microsoft Visual Studio 2019— это набор инструментов разработки, основанных на использовании компонентов, и других технологий для создания мощных, производительных приложений. Кроме того, среда Visual Studio оптимизирована для совместного проектирования, разработки и развертывания корпоративных решений. Visual Studio предоставляет средства для проектирования, разработки b отладки.

Среда разработки Visual Studio представляет собой полный набор средств разработки для создания веб-приложений ASP.NET, XML (веб-службы), настольных приложений и мобильных приложений. Visual Basic, Visual C# и Visual C++ используют единую интегрированную среду разработки (IDE), которая позволяет совместно использовать средства и упрощает создание решений на базе нескольких языков. Можно создать обычные приложения Microsoft Windows и приложения с архитектурой "клиент-сервер" с помощью использования конструкторов в Visual Studio.

Почему именно эта версия продукта, потому что в нем есть такие удобные нововведения как:

1. Visual Studio IntelliCode повышает эффективность разработки программного обеспечения с помощью искусственного интеллекта (ИИ). Для создания рекомендаций IntelliCode анализирует 2000 проектов с открытым кодом на GitHub

2. Рефакторинг В C++ есть много новых удобных возможностей рефакторинга, которые помогают упорядочить код. Они отображаются как предложения со значком лампочки и включают такие действия, как перемещение элементов в интерфейс или базовый класс, настройку пространств имен в соответствии со структурой папок.