МЕТОДЫ ВЗЛОМА И ЗАЩИТЫ ПРИЛОЖЕНИЙ. НАРУШЕНИЕ АВТОРСКИХ ПРАВ

Протасов Евгений Олегович

Гродненский государственный университет им.Янки Купалы,
факультет математики и информатики,
кафедра системного программирования и компьютерной безопасности
студенты 3 курса специальности «Компьютерная безопасность»

Научный руководитель: Кадан Александр Михайлович, кандидат тех. наук, доцент.

В работе описываются краткие сведения о методахзащиты приложений. Анализируются методы взлома от приложений и их применение на практике. На сегодняшний день перед каждым предприятием, обеспокоенного вопросами безопасности своих информационных ресурсов, встает вопрос об организации системы защиты информации, которая бы позволила в полной мере обеспечить безопасность систем и в данной работе будут рассмотрены некоторые методы для обеспечения безопасности системы.

Ключевые слова: ЗАЩИТА ПРИЛОЖЕНИЙ. ВЗЛОМ ПРИЛОЖЕНИЙ. НАРУШЕНИЕ АВТОРСКИХ ПРАВ. ЗАЩИТА АВТОРСКИХ ПРАВ. ПАТЧИНГ. ОБФУСКАЦИЯ.

Проблема компьютерного пиратства остается актуальной до настоящего времени. Под пиратством понимается нелегальное использование программного обеспечения (ПО). Даже если не принимать во внимание распространение бесплатных копий программ, существует вероятность, что эти копии будут обладать дополнительным функционалом, который был добавлен теми, кто занимался взломом программы. Дополнительным функционалом может быть как самое обычное всплывающее окно с рекламой, так и вирус удалённого доступа, через который злоумышленник может получить доступ к данным на компьютере пользователя, пожалевшего деньги на приобретении лицензии. В связи с этим возникают различного рода угрозы безопасности и целостности данных.

Несмотря на огромный технический прогресс методов защиты информации, злоумышленники тоже не сидят на месте. Параллельно введениям новых методов защиты появляются и новые методы атак. д Взломы программ обычно производятся для получения бесплатной версии программы или же для получения доступа к исходному коду программу. Для предотвращения угроз утечки какого-либо рода данных существуют определённые средства защиты, но даже с учётом использования всех этих средств присутствует вероятность взлома или же кражи информации. С учётом вышесказанного, в данной работе будут рассмотрены различные методы взлома и защиты приложений.

Способы взлома приложений

Среди самых распространённых методов взлома приложений можно выделить:

1. Использование генераторов — Создание серийного ключа, используя знания о поведении программы в момент проверки ключа. Типичной ошибкой разработчиков является то, что для создания ключа они используют регистрационные данные (имя, фамилия). Лучше использовать серийные номера, соответствующие набору правил, и не привязанные ни к чему-либо.
2. Декомпиляция и дизассемблирование — Изучение внутренней структуры кода. На сегодняшний день существует большое количество программ дизассемблеров (IDA [1], W32DASM [2]).
3. Патчинг — Внесение определённых изменений в файлы программы. Основной идей этой технологии является поиск строк, связанных с выводимыми сообщениями пользователю. Скажем, когда пользователь попытается ввести пароль ему, вероятно, после неудачной попытки программа выведет сообщение “Неверный пароль”, “Попробуйте ещё раз”. Хакеру остаётся найти эти строки в сегменте данных и узнать, где используются ссылки на эти строки.
4. Полный перебор пароля — способ, который нужно использовать в последнюю очередь, так как он может занять от нескольких дней до столетий в зависимости от количества всех возможных решений задачи (см. рис. 1).

Таблица 1 – Время подбора пароля в зависимости от количества символов в пароле

Знаков	**Количество вариантов**	**Время перебора**
1	36	менее секунды
2	1296	менее секунды
3	46656	менее секунды
4	1679616	17 секунд
5	60456176	10 минут, 5 секунд
6	2176681336	6 часов, 2 минуты
7	78363154096	9 дней, 2 часа, 16 минут, 26 секунд
8	$2.8211099x10^{12}$	10 месяцев, 23 дня, 52 минуты, 37 секунд
9	$1.0156995x10^{14}$	32 года, 3 месяца, 7 дней, 12 часов, 11 минут
10	$3.6565584x10^{15}$	1161 год, 8 месяцев, 26 дней, 18 часов, 33 минуты, 40 секунд

Примечание: Источник: Контроль разума | Fandom [3]

Способы защиты программы

Для защиты следует использовать то, что плохо поддаётся декомпиляции или дизассемблированию. Можно использовать CRC (контрольные суммы) [4]. Блок данных, строки или даже файл можно защитить, используя контрольную сумму, которую затем можно рассчитать и сравнить с первоначальной. Также контрольные суммы спасают от вирусов или внедрения троянских программ [5]. Чтобы усложнить взломщику разбор программы, необходимо обфусцировать код.

Сегодня можно часто встретить защиту, называемую «ограничение времени работы». Её смысл в том, что фиксируется первый запуск программы на компьютере и программа работает специально заданное время (обычно 30 дней). По истечению этого срока программа перестаёт работать. Самое главное в этой защите — «спрятать» дату первого запуска. Однако хранение в реестре можно обнаружить при помощи программы Regmon [6], а в изменение файлов при помощи Filemon [7].

Среди самых распространённых методов защиты приложений можно выделить:

1. Электронные ключи — Ключ не защищает программу от копирования и распространения, но без электронного ключа невозможно использование программы, то есть само по себе копирование является бессмысленным. Обычно ключ подсоединяется к компьютерным портам, USB или LPT. Ключ состоит из платы с микросхемами. Контроллер, в свою очередь, содержит набор команд, которые содержат функции для генерации информационных блоков защиты программы. В памяти ключа содержится информация о его характеристиках и пользователей.
2. Обфускация кода — Изменение структуры программы для усложнения исследования кода. Например, используя несколько переименований можно изменить понятный код класса (см. лист. 1):

```
public class Book {

    String name;

    public Book(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

}
```

Листинг 1 – Код класса “книга”

Примечание: Источник: Собственная разработка

в непонятный набор инструкций (см. лист. 2):

```
public class a {

    String c;

    public a(String c) {
        this.c = c;
    }

    public String getC() {
        return c;
    }
}
```

Листинг 2 –Обфусцированный код класса “книга”

Примечание: Источник: Собственная разработка

3. Цифровая подпись — Метод, который связывает что-либо с цифровыми данными. Данная связь может быть независимо проверена получателем, а также любой третьей стороной. Открытый ключ доступен любому через публичный репозиторий или каталог. Закрытый ключ должен быть конфиденциальным строго для соответствующего владельца. Так как эта пара математически связана, всё, что зашифровано открытым ключом может быть расшифровано только закрытым и наоборот. В реальном мире получатель сообщения нуждается в гарантии того, что сообщение принадлежит отправителю, и он не должен быть в состоянии сомнений относительно происхождения этого сообщения. Это требование очень важно, поскольку вероятность возникновения спора по поводу подлинности обмениваемых данных очень высока.
4. Маркер автора программы — Хорошо спрятанный, чаще всего зашифрованный, набор символов, который свидетельствует о том, что вы являетесь разработчиком. Такой маркер можно оставить, открыв исполняемый файл своей программы в 16-ричном виде и вписав зашифрованный текст в свободные блоки одного из заголовков файла.

Внедрение маркера автора в файл приложения

Предположим, у нас есть приложение, которое мы хотим пометить своим маркером. В качестве маркера будет взята строка “Yauheni Pratasau GRSU”, предварительно зашифрованная при помощи симметричного алгоритма блочного

шифрования AES (Rijndael). Для начала представим файл нашего приложение в 16-ричной системе счисления, открыв её в любом из редакторов. Видим набор значений из 16-ричной системы счисления. Находим свободное место в директории данных (рис. 1) и в этой области помещаем нашу зашифрованную строку (рис 2.). Самый простой способ маркирования программы произведён.

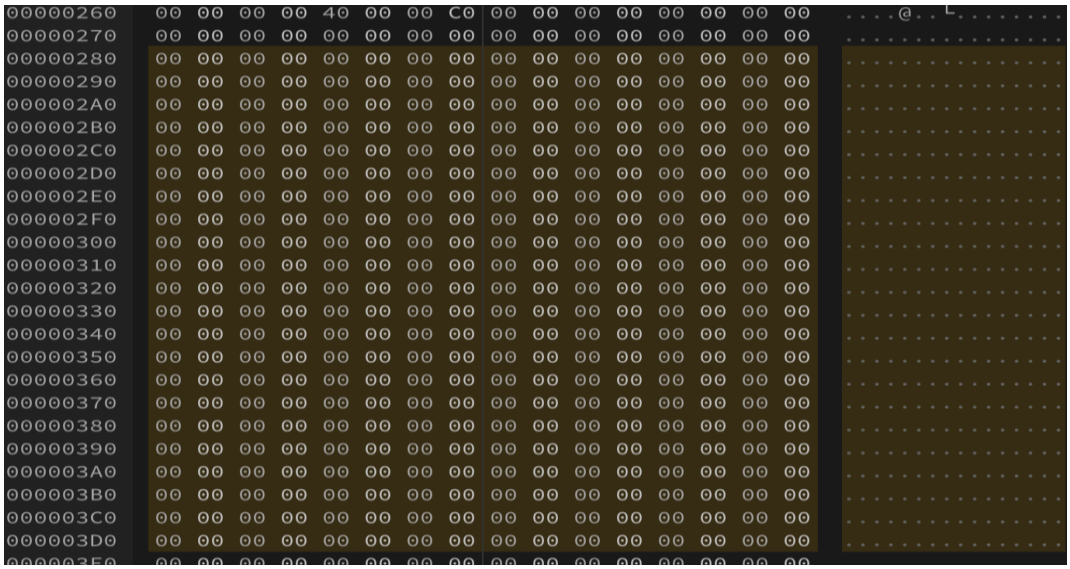

Рисунок 1 – Директория данных

Примечание: Источник: Собственная разработка

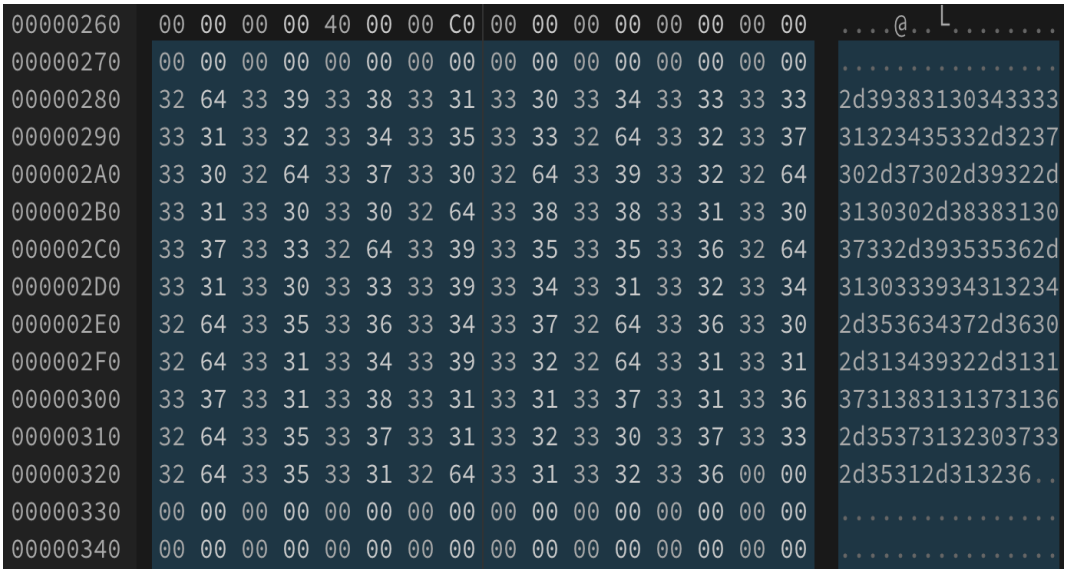

Рисунок 2 – Внедрение маркера автора

Примечание: Источник: Собственная разработка

Заключение

Защищать программу одним лишь паролем стоит лишь в том случае, когда атака методом перебора будет мало эффективной. Не стоит анализировать введённый пароль или регистрационный ключ сразу же после ввода, проверять его в одном методе, не хранить результаты проверки в переменных и не использовать их для явного ограничения методов незарегистрированного ПО. Нельзя хранить данные в открытом виде, особенно в виде «Ключ не верен», «Продукт зарегистрирован».

Намного эффективнее будет если, требуемый для активации, пароль необходим для шифрования некоторого функционала программы. Тогда хакеру, после перебора паролей, придётся ещё расшифровывать код.

Свести процесс взлома к нулю, можно попробовать поставив защиту отлавливающую пошаговую отладку программы. Одним из примеров такой защиты является таймер. При работе программы отмечаем системное время и вычисляем время работы определённых частей кода. И, скажем, если 100-300 команд процессора выполняются более 2ух минут, то есть над чем подумать.

Литература

1. IDA About [Электронный ресурс] / Hex-Rays. – Hex-Rays SA. – Режим доступа: https://www.hex-rays.com/products/ida/. – Дата доступа: 22.03.2020
2. W32DASM [Электронный ресурс] / Wade. – James Wade Blog. Режим доступа: https://wade.be/w32dasm/. – Дата доступа: 22.03.2020
3. Полный перебор | Контроль разума | Fandom [Электронный ресурс] / Fandom. – Mind-control Fandom. – Режим доступа: https://mind-control.fandom.com/wiki/Полный_перебор. – Дата доступа: 22.03.2020
4. Циклический избыточный код [Электронный ресурс] / Wikipedia: платформа для хранения медиа информации. – Wikipedia, 2016. – Режим доступа: https://ru.wikipedia.org/wiki/Циклический_избыточный_код. – Дата доступа: 22.03.2020
5. Троянская программа [Электронный ресурс] / Wikipedia: платформа для хранения медиа информации. – Wikipedia, 2016. – Режим доступа: https://ru.wikipedia.org/wiki/Троянская_программа. – Дата доступа: 22.03.2020
6. Regmon for Windows [Электронный ресурс] / M.Russinovich. – Microsoft 2019. – Режим доступа: https://docs.microsoft.com/ru-ru/sysinternals/downloads/regmon. – Дата доступа: 22.03.2020
7. Filemon for Windows [Электронный ресурс] / M.Russinovich. – Microsoft 2019. – Режим доступа: https://docs.microsoft.com/ru-ru/sysinternals/downloads/filemon. – Дата доступа: 22.03.2020