Подсистема лицензирования поможет предотвратить незаконное распространение программного обеспечения и будет способствовать оперативному переносу лицензионной информации с одного сервера на другой.

Литература

1 Стоимость нелицензионного программного [Электронный ресурс]. URL: http://gss.bsa.org/wp-content/uploads/2018/05/2018_BSA_GSS_Report_en.pdf (дата обращения : 20.05.2019).

2 Программный комплекс «ТехноДок» [Электронный ресурс]. URL: http://www.sms-automation.ru/solutions/technodoc/ (дата обращения: 20.05.2019).

Р.И. Канафеев, М.А. Кудрина

ИСПОЛЬЗОВАНИЕ ОБФУСКАЦИОННЫХ ПРЕОБРАЗОВАНИЙ ДЛЯ ЗАЩИТЫ ПРОГРАММНОГО КОДА

(Самарский университет)

Несмотря на наличие всего комплекса законодательных и правовых мер по защите авторских и смежных прав на интеллектуальную собственность, ситуация с пиратским рынком программного обеспечения (ПО) остается весьма плачевной. Согласно исследованиям международной ассоциации производителей программного обеспечения BSA, доля использования пиратского ПО в России в 2017 г. составила 62% [1]. Для производителей коммерческого ПО это означает огромную недополученную прибыль. Подобная проблема существует не только на отечественном рынке, но и во всем мире. Для ее устранения используются различные методы защиты программного обеспечения, которые получили широкое распространение и находятся в процессе постоянного развития, благодаря глубокой интеграции информационных технологий в общество.

Упаковка исполняемого файла и техники защиты ПО, основанные на недокументированных возможностях среды программирования, не защищают должным образом от анализа и модификации программ, поскольку в первом случае код доступен в момент передачи на него управления, а во втором – злоумышленнику достаточно знать какой именно прием используется, чтобы в дальнейшем создать механизмы, способные преодолеть такого рода защиту [2-4]. Более того, приемы, основанные на недокументированных возможностях, могут дестабилизировать работу приложения. Очевидно, что защитить код приложения от доступа к нему невозможно. А значит в обоих случаях возможно создать автоматические средства деактивации защиты. Следовательно, первоочередная задача защиты ПО от анализа – максимально затруднить понимание внутренней логики работы ПО злоумышленником и обеспечить невозможность создания им автоматической утилиты модификации этого кода. Тогда даже обладая доступом к коду приложения, его анализ будет крайне затруднен. В связи с этим наиболее актуальными в настоящий момент являются методы

защиты ПО, основанные на запутывании кода и данных программы – обфускации.

Обфускацией называется приведение исходного текста или исполняемого кода программы к виду, сохраняющему ее функциональность, но затрудняющему анализ, понимание алгоритмов работы и модификацию при декомпиляции [5].

Методов обфускации достаточно много. Подавляющее их большинство основано на компиляторных технологиях. Одни методы требуют наличия исходного кода программы, другие же оперируют машинным кодом целевой платформы. Методы, оперирующие конструкциями высокоуровневого языка, как правило, не защищают от модификации. Использовать их для автоматической защиты ПО затруднительно, т. к. необходимо соблюдать достаточно большое количество требований по компиляции приложения и создавать программы на том языке программирования, конструкциями которого оперирует обфускатор [6].

Методы, оперирующие машинным кодом целевой платформы, позволяют защитить код не только от анализа, но и от модификации, не требуют наличия исходного кода программы, не требуют создания программы на каком-то конкретном языке программирования. Однако эти методы являются менее стойкими к существующим технологиям деобфускации, основанным на оптимизирующих преобразованиях.

Методы обфускации программного кода можно разделить на две группы [7]:

1) преобразование графа потока управления;

2) трансформация данных.

Как показано на рисунке 1, преобразования графа потока управления делятся на: влияющие на агрегирование, упорядочивание и вычисление потока управления.

Преобразования агрегирования потока управления разбивают вычисления, которые логически принадлежат друг другу или объединяют вычисления, которые этого не делают. Преобразования упорядочивания потока управления делают случайной последовательность выполнения вычислений. Преобразования вычислений вставляют новый (избыточный или мертвый) код или вносят изменения в сам алгоритм исходного приложения.

Для преобразований, которые изменяют поток управления, характерно наличие большого количества дополнительных вычислений. Это означает, что разработчику придется выбирать между эффективностью программы и ее производительностью.

Преобразования, которые затрудняют понимание структур данных, используемых в приложении, можно разделить на преобразования, влияющие на агрегацию, упорядочивание, хранение и кодирование данных (рисунок 2).

Рисунок 7 – Классификация обфускационных преобразований графа потока управления

Рисунок 8 – Классификация обфускационных преобразований, использующих метод трансформации данных

Обфускация преобразований хранения пытается выбрать неестественные классы хранения для динамических и статических данных. Аналогично, преобразования кодирования пытаются выбирать неестественные кодировки для общих типов данных. Преобразования хранения и кодирования зачастую используются вместе.

В объектно-ориентированной программе управление организовано вокруг структур данных, а не наоборот. Это означает, что важная часть реверс-инжиниринга объектно-ориентированного приложения – восстановление структуры данных программы. И наоборот, для обфускатора важно скрыть эти структуры данных. В большинстве объектно-ориентированных языков существует всего два способа агрегирования данных: в массивах и в объектах. Эти структуры данных могут быть преобразованы с помощью объединения скалярных переменных, реструктуризации массивов и изменения отношений наследования.

Также полезно ранжировать порядок объявлений в исходном приложении. В частности, упорядочиваем порядок методов и переменных экземпляров внутри классов и формальных параметров внутри методов. В последнем случае, конечно, соответствующие изменения будут, разумеется, также переупорядочены. Во многих случаях также можно изменить порядок элементов внутри массива.

Среди различных средств, доступных для защиты кода от различных атак, обфускация является одной из самых популярных альтернатив для предотвращения от анализа и модификации. Все методы и техники обфускации ПО имеют свои достоинства и недостатки, поэтому зачастую разработчики комбинируют их для создания более надежной защиты.

Литература

1. BSA Global Software Survey [Электронный ресурс] / Software Management: Security Imperative, Business Opportunity. – Режим доступа: https://gss.bsa.org/wp-content/uploads/2018/05/2018_BSA_GSS_Report_en.pdf, свободный. – Яз. англ.

2. Щелкунов, Д. А. Разработка методик защиты программ от анализа и модификации на основе запутывания кода и данных. [Текст] : дис. канд. тех. наук : 05.13.19 / Щелкунов Дмитрий Анатольевич. – М., 2009. – 143 с.

3. Щелкунов Д.А. Автоматическая защита программ от исследования и отладки [Текст] / Интеллектуальные системы (INTELS 2006): Сб. Трудов VII Международ, симпоз. - Краснодар, 2006. - 221 с.

4. Рихтер Дж. Windows для профессионалов: создание эффективных Win32 приложений с учетом специфики 64-разрядной версии Windows [Текст] /Пер, англ - 4-е изд. - СПб; Питер; М.: Издательско-торговый дом «Русская Редакция», 2001. - 752 с; ил.

5.][-препарация: вскрываем хитрый Sality.aa: Учимся распознавать полиморфизм и обфускацию кода на примере известного вируса [Электронный ре-

сурс] / Журнал «Хакер». – Режим доступа: https://xakep.ru/2011/03/09/54790, свободный.

6. Чернов, А. В. Об одном методе маскировки программ [Текст] /А.В. Чернов // Труды Института Системного программирования РАН. – 2003. – С. 85-119. – ISSN 2079-8156.

7. Christian Collberg. A taxonomy of obfuscating transformations. [Текст] / Christian Collberg, Clark Thomborson, Douglas Low. – 1997.

В.А. Кардаков

БЕЗОПАСНОСТЬ ТЕХНОЛОГИИ IoT

(Казанский национальный исследовательский технический университет
им. А.Н.Туполева-КАИ)

Технология IoT (интернет вещей) набирает с каждым годом всё больше интереса и популярности. Выходят большинство новостей о «умных» бытовых приборах (чайник, холодильник, кофеварка и т.п.). Например, имеется статья про создание интернет-розетки, при помощи которой можно управлять чайником через интернет. Появляются компании, которые предлагают управлять холодильниками, кофеварками и даже автомобильными парковками. Так что же это такое IoT (интернет вещей)?

IoT – концепция вычислительной сети физических предметов, оснащённых встроенными технологиями для взаимодействия друг с другом или с внешней средой, рассматривающая организацию таких сетей как явление, способное перестроить экономические и общественные процессы, исключающее из части действий и операций необходимость участия человека [1].

IoT использует те же протоколы, которые используем мы для передачи информации с одного персонального устройства (ПК, ноутбук, смартфон и т.п.) на другой при помощи сети интернет. Только в качестве исходящих или принимающих устройств выступают бытовые приборы, двери, окна и т.п. Благодаря технологии IoT имеется возможность создания полноценных «умных» домов, когда удаленно при помощи смартфона и интернета мы можем открывать или закрывать окна, выключать и включать электричество, запустить кофемашину, включить телевизор и т.п.

Также IoT предусматривает автономную работу обмена данными между устройствами без участия человека. Благодаря этим двум факторам, бытовые предметы могут превратиться в интернет-узлы. И уже имеются экспертные исследования проблем безопасности IoT. Уже в 2008 г. (в этом году IoT появилось как явление) IoT фигурирует в отчете Национального разведывательного совета США в качестве одной из шести потенциально разрушительных технологий.

В 2015 году, компания OpenDNS представила собственные результаты исследования, проведенного в корпоративных сетях, которые используют технологию IoT. Бренд OpenDNS принадлежит компании Cisco, одному из лидеров