А. А. Краснопевцев

РАЗРАБОТКА АВТОМАТИЧЕСКОЙ ЗАЩИТЫ ПРИЛОЖЕНИЙ И ПЕРЕДАВАЕМЫХ, ОБРАБАТЫВАЕМЫХ И ХРАНИМЫХ ИМИ ДАННЫХ[1]

При разработке программного обеспечения большинство разработчиков выбирают такие среды разработки приложений, как Java [1] или .NET. Действительно, функционал, предлагаемый данными средами разработки, позволяет значительно упростить процесс создания и развития программного обеспечения. Кроме того, важно отметить тот факт, что разрабатываемое подобным образом программное обеспечение является аппаратно независимым. Все проблемы с переносимостью разработанного приложения с одной платформы на другую ложатся на плечи разработчика самой среды разработки.

Таким образом, скорость и простота создания приложения, а как следствие, затраты человеческих и финансовых ресурсов значительно снижаются. В итоге разработчик способен больше времени уделять непосредственно качеству самого программного обеспечения, а не исправлять проблемы совместимости с той или иной платформой.

Однако у данного способа разработки программного обеспечения есть один большой недостаток. В результате полного описания на внутреннем языке состояния приложения в любой момент времени приложение достаточно легко анализируется злоумышленником [2]. Таким образом, для разработчика очень остро встает проблема защиты своего творения от несанкционированного копирования и распространения, т. е. защиты своих авторских прав.

При исследовании существующих способов защиты приложений от копирования [3, 4] в рамках проведенной работы было выявлено, что большинство способов защиты приложений являются лишь средствами, способными усложнить в небольшой мере анализ приложения, а не предотвратить его. Таким образом, возникает необходимость разработки системы защиты приложений от копирования, а также защиты передаваемых, обрабатываемых и хранимых этим приложением данных. Необходимость защиты данных продиктована тем, что зачастую системы защиты данных, предлагаемые разработчиком, не являются таковыми, а способны только маскировать или незначительно затруднить анализ передаваемых данных.

В рамках данной статьи предлагается способ обеспечения как защищенности самого приложения от несанкционированного копирования, так и защиты передаваемых, обрабатываемых и хранимых этим приложением данных.

Для определения метода, который будет использован, проводился анализ существующих методов защиты приложений от копирования. Было выявлено, что наилучшим образом для защиты приложений, компилируемых в некоторое промежуточное представление, подходят способы защиты, основанные на электронных ключах с загружаемым кодом. Причиной такого выбора стала возможность с использованием таких электронных ключей производить частичную или полную выгрузку кода защищаемого приложения во внешний защищенный контейнер.

При анализе свойств электронных ключей с загружаемым кодом было выявлено, что при исследовании данной системы защиты злоумышленнику придется использовать способы анализа, применяемые при анализе черного ящика. Более приемлемых методов для анализа предлагаемой в данной работе системы защиты не существует.

[1] Статья написана в рамках НИР «Обеспечение безопасности информации в открытых распределенных вычислительных системах», заданной Государственным контрактом № П2397 в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы.

В результате проведения исследований по проблеме был разработан способ защиты приложений от копирования, а также защиты передаваемых, обрабатываемых и хранимых этим приложением данных. Алгоритм защиты приведен ниже [5, 6]:

- приведение кода приложения к некоторой внутренней интерпретации, удобной для дальнейшей работы всего программно-аппаратного комплекса;
- выделение мест входа и выхода информации в/из приложения;
- определение функций, подлежащих защите;
- тестирование найденных функций на возможность выгрузки их во внешний электронный носитель;
- декомпиляция выбранных функций в язык высокого уровня;
- компиляция кода, полученного на предыдущем шаге, в ассемблер процессора электронного ключа;
- выгрузка скомпилированного модуля в электронный ключ;
- удаление оригинального кода функции из тела приложения;
- установка в код заглушек для обработки информации и вызова защищенных функций.

При следовании разработанному алгоритму приложение будет выполняться частично на процессоре хостовой машины, частично — внутри внешнего аппаратного модуля, т. е. электронного ключа с загружаемым кодом.

Таким образом, для проведения анализа защищенного приложения злоумышленнику будет необходимо производить динамическое исследование приложения, а кроме того, что наиболее важно, исследовать алгоритмы, находящиеся внутри электронного ключа. Причем вследствие архитектуры самого электронного ключа доступа к хранимой в нем информации извне не существует, т. е. злоумышленнику придется проводить анализ, основываясь только на зависимостях между входными и выходными данными. Таким образом, в случае предложенных в данной работе мер по защите приложения защищенность приложения сводится к стойкости черного ящика, что, в свою очередь, значительно усложняет, а в некоторых случаях делает совершенно невозможным анализ приложения.

Кроме того, в рамках проведения данной научно-исследовательской работы была предложена архитектура программно-аппаратного комплекса, обеспечивающего защиту как самих приложений от несанкционированного копирования, так и передаваемых, обрабатываемых и хранимых этими приложениями данных. Архитектура представляется таким образом, чтобы система могла быть доработана, поэтому архитектура программно-аппаратного комплекса представляет собой набор модулей, которые линейно между собой взаимодействуют. Основные модули:

- дизассемблер;
- декомпилятор;
- компилятор в ARM;
- утилита работы с электронным ключом;
- модуль генерации кода приложения.

Каждый модуль, как следует из его названия, выполняет свою строго описанную часть алгоритма.

В результате применения программно-аппаратного комплекса к защищаемому приложению получится защищенное приложение, которое будет жестко связано с внешним аппаратным модулем. Полная компрометация системы защиты возможна только в том случае, если злоумышленник каким-либо образом сможет восстановить полный функционал всех процедур, которые исполняются в рамках электронного ключа. Однако стоит отметить, что проведение анализа алгоритма с использованием только методов анализа черного ящика может не дать вообще никаких результатов, вследствие этого разработанная система защиты как самого приложения,

так и передаваемых, обрабатываемых и хранимых этим приложением данных имеет достаточно высокую стойкость к компрометации.

Изложенные результаты получены в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009—2013 годы.

СПИСОК ЛИТЕРАТУРЫ:

1. Технологии Java. URL: http://www.sun.com/software/index.jsp?cat=Java%20.
2. *Ахо А., Ульман Д.* Теория синтаксического анализа, перевода и компиляции. М.: Мир, 1978.
3. *Холанд Г., Мак-Гроу Г.* Взлом программного обеспечения, анализ и использование кода. М.: Издательский дом «Вильямс», 2005. — 400 с.: ил.
4. *Матросов А. А.* Курс лекций «Защита программного обеспечения». URL: http://aktivco.ru/course/lecture/.
5. *Краснопевцев А. А.* Разработка средств автоматизации для переноса байт-кода во внешний аппаратный модуль // Технологии Microsoft в теории и практике. М.: Вузовская книга, 2008. С. 139—141.
6. *Краснопевцев А. А., Букасов В. А.* Разработка средств автоматической защиты приложений, содержащих байт-код // Материалы 10-й Международной научно-практической конференции. Таганрог. 24—27 июня 2008 г. С. 15—16.