ИНТЕГРИРОВАННЫЙ КОМПЛЕКС ЗАЩИТЫ ПРОГРАММНЫХ ПРОДУКТОВ

УДК 004.056.5

ПАРАМОНОВ Антон Иванович
доцент кафедры компьютерных технологий Донецкого национального университета.
Научные интересы: автоматизированное управление, информационные технологии.
e-mail: a.paramonov@donnu.edu.ua, paramonov_anton@mail.ru

ВВЕДЕНИЕ

Современное сообщество характеризуется широким применением информационных технологий в различных процессах. Область применения информационных систем (ИС) постоянно растет. В условиях большой конкуренции разработчики программных продуктов стремятся обеспечить свои программы мощной функциональностью, при этом все меньше времени уделяют их защите. В связи с этим возникает потребность в системах защиты данных, с которыми работают программы и самих программ. Несмотря на то, что уже существует множество различных подходов для обеспечения защиты программного обеспечения, данная задача остается сегодня одной из актуальных в сфере информационных технологий. Разработки, которые нацелены на решение указанной проблемы, традиционно принято разделять на следующие классы: технические (аппаратные), программные и комплексные (аппаратно-программные) средства обеспечения безопасности ИС. Аппаратные средства защиты преимущественно представляют собой технические или сложные программно-технические решения, такие как, например, специальные регистры для хранения реквизитов защиты, устройства измерения индивидуальных характеристик человека и т.д. Основными методами работы технических средств защиты можно назвать предотвращение физического доступа к носителям информации, предотвращение доступа к информации и ИС, обнаружение и блокирование каналов утечки информации. Системы защиты такого класса, как правило, имеют высокий уровень надежности, обладают устойчивостью к модификации, независимостью от субъективных факторов, однако не могут обеспечить достаточной гибкости решений, сложны в реализации и очень дорогостоящие. Программные средства защиты – это системы, работающие на уровне операционной системы или её подсистем (драйверы файловой системы, работа в нулевом кольце и пр.). К таким системам относятся программы, которые встраиваются в исходный код защищаемой системы, надстраиваются над программным продуктом (ПП), либо внедряются в готовый продукт после компиляции. Такие системы зачастую менее дорогостоящие и предоставляют, в зависимости от используемых алгоритмов, вполне допустимую степень надежности защиты. Применение программных средств защиты позволяет найти компромисс между уровнем защиты (степенью надежности системы защиты) и стоимостью ее реализации и сопровождения. Одними из немаловажных аспектов выбора средств защиты ИС выступают свойства систематичности и масштабности. Это означает, что защитная система должна гибко настраиваться на различные условия применения и стабильно работать с различными видами программного обеспечения.

ОБЗОР СУЩЕСТВУЮЩИХ ПОДХОДОВ

Наиболее популярными сегодня являются системы в виде модулей, устанавливаемых на скомпилированный программный продукт. Они представляют собой исполняемые модули, переносимые на компьютер клиента вместе с защищаемым ПП. Таким образом,

каждый запуск этого ПП, сопровождается запуском дополнительного модуля, который и выполняет функции контроля. Стоит отметить, что таким методом достаточно сложно обеспечить высокий уровень защиты от взлома (возможны атаки – деактивация контролирующего модуля, обходные пути запуска защищаемого ПП и др.). С другой стороны, системы, которые встраиваются в исходный код ПП до компиляции, неудобны для производителя этого продукта, так как возникает необходимость обучать персонал работе с программным интерфейсом (API) системы защиты, что влечет за собой финансовые и временные затраты. Кроме того, усложняется процесс тестирования программного обеспечения и снижается его надежность, так как кроме самого ПП ошибки может содержать еще и API системы защиты или процедуры, его использующие. Такие системы являются более стойкими к атакам, ведь здесь практически исчезает четкая грань между системой защиты и ПП как таковым. Стоит отметить, что такие системы защиты достаточно сложны и дороги, что вызывает ощутимое возрастание цены защищаемого программного обеспечения. Одними из наиболее перспективных и надежных программных решений являются те, которые построены на основе механизма внедрения программного кода в готовый исполняемый модуль приложения. К преимуществам таких систем относится простота использования, стойкость ко взлому, относительно невысокая сложность реализации алгоритма защиты, что обуславливает приемлемую стоимость системы.

Целью данной работы является разработка подхода к организации защиты ПП на основе метода внедрения фрагмента кода в исполняемый файл.

Метод внедрения кода подразумевает интеграцию фрагмента кода системы защиты в защищаемое приложение [1]. Защищаемое приложение будем называть также родительским (или базовым) приложением. Внедряемый фрагмент кода будем называть модулем защиты или X-кодом. Для платформы Win32 существует единственный унифицированный формат исполняемых файлов, так называемый PE (Portable executable) файл [2]. PE файлы изначально проектировались очень разумно, поэтому изменения, связанные с переходом на 64-битную архитектуру будут минимальны, и можно спокойно использовать данный формат. Ряд характерных особенностей, присущих файлам PE формата [1, 2] позволяют разрабатывать и реализовывать эффективные методы внедрения программного кода для защиты приложения. К числу таких особенностей можно отнести секционную структуру исполняемого файла, использование файлового выравнивания. Среди основных и наиболее часто применяемых методов внедрения кода можно выделить:

1. Внедрение путем сжатия части файла – часть исполняемого файла сжимается с применением алгоритма RLE (или более совершенного). В освободившееся пространство помещается внедряемый код. Такой метод достаточно прост и надежен, однако не подходит для сжатых файлов. Кроме того, метод характеризуется значительным временем выполнения, необходимым на распаковывание сжатого кода, что является серьезным недостатком для применения на файлах большого размера.

2. Внедрение путем растяжения заголовка PE файла. Сущность алгоритм заключается в увеличении размера PE заголовка исполняемого файла и размещении на освободившемся пространстве внедряемого кода. Данный метод представляет значительные трудности при реализации, так как растяжение заголовка влечет за собой серьезные изменения в структуру исполняемого файла. В большинстве реализаций метода не удается достичь надежности, приемлемой для продуктивной эксплуатации.

3. Внедрение путем создания дополнительной секции. Метод предполагает создание в конце исполняемого файла дополнительной секции, содержащей внедряемый код. Такой метод наиболее прост и удобен, так как практически отсутствуют ограничения по размеру внедряемого кода. В то же время, алгоритм внедрения достаточно прост, что предопределяет высокую надежность алгоритма внедрения. Однако, в силу того, что внедряемый код расположен компактно и вынесен в отдельный нестандартный элемент (дополнительную секцию кода), метод характеризуется низкой устойчивость к взлому.

РАЗРАБОТКА МЕТОДА ВНЕДРЕНИЯ ЗАЩИТНОГО КОДА В ПРИЛОЖЕНИЕ

Анализ подходов к внедрению Х-кода показал, что использование метода внедрения путем создания дополнительных секций будет наиболее эффективным. Задача внедрения кода в приложение включает два этапа: создание приложения, реализующего непосредственно один из алгоритмов внедрения программного кода; и разработка необходимого для внедрения кода. Внедряемый Х-код должен отвечать жестким требованиям, предъявляемым средой, в которую он будет внедрен. Во-первых, Х-код должен быть полностью перемещаем, то есть сохранять работоспособность независимо от базового адреса загрузки. Решением этой проблемы является использование относительной адресации. Во-вторых, Х-код не должен модифицировать свои ячейки. Это обусловлено тем, что по умолчанию секция кода доступна только для чтения. В-третьих, на Х-код накладываются жесткие ограничения по размеру. Это обусловлено особенностями защищаемого файла. Таким образом, вводится критерий допустимости внедрения:

$$X_{IMPL} = O_{HB} + S_{IH} + S_{OP} + (N_S + 1) * \text{sizeof (IMAGE\_SECTION\_HEADER);} \quad (1)$$

$$X_{IMPL} \le \min (O_{DS}),$$

где O_{HB} (OffsetOfHeaderBegin) – смещение заголовка от начала файла, S_{IH} (SizeOfImageHeader) – размер заголовка файла, S_{OP} (SizeOfOptionalHeader) – размер опционального заголовка, N_S (NumberOfSections) – количество записей в таблице секций, IMAGE_SECTION_HEADER – стандартная структура, описывающая секцию PE файла [3], O_{DS} (OffsetOfDataSection) – смещение данных секции, относительно начала файла. Из (1) видно, что внедрение возможно, если встраиваемый код (X_{IMPL}) меньше либо равен «свободного» для вставки места.

Дополнительную трудность представляет собой то, что внедряемый код не должен задерживать управление на длительный период времени. Таким образом, предположим, что для большей устойчивости к взлому необходимо, чтоб внедряемый код защитной системы был максимально оптимизирован, как по размеру, так и по времени работы.

Работу системы защиты на основе принципа встраиваемого кода можно описать в виде диаграммы деятельности, которая представлена на рисунке 1.

При запуске родительского приложения, в которое имплементирован код подсистемы защиты, управление передается на определенный фрагмент этого Х-кода. Х-код выполняет проверку разрешения на запуск родительского приложения в соответствии с заданными критериями. Поскольку система защиты позволяет устанавливать различные комбинации критериев, то проверка реализуется в виде многошагового процесса. В системе предусмотрены следующие критерии защиты: ограничения по числу запусков, ограничения по времени использования, ограничения по количеству пользователей единовременно работающих с системой, составные ограничения, являющиеся комбинациями вышеперечисленных.

Для конфигурирования механизма защиты программного комплекса используются лицензии. Лицензия каждого вида предоставляет пользователю право использования функционала информационной системы в переделах ограничений, установленных лицензией. Программный комплекс поддерживает несколько видов лицензий. В качестве критериев защиты приложения используются «время жизни» (предполагается ограниченный срок использования защищенного программного продукта от момента установки), «время использования», количество запусков и количество пользователей, единовременно работающих с системой. Дополнительной опцией является тип лицензии «Компьютер». Ограничивающим критерием этой лицензии является компьютер пользователя: лицензия содержит в себе набор данных, уникально идентифицирующий компьютер пользователя и запрещает запуск защищенного приложения на других компьютерах. При конфигурировании системы можно сформировать комплексную лицензию, представляющую собой комбинацию нескольких простых лицензий.

Если при проверке удовлетворяются все критерии защиты, то выполняется возврат управления родительскому приложению для дальнейшей работы, в противном случае если какие-то из критериев защиты приложения не удовлетворяются, то работа защищаемого программного продукта прекращается. Классическим путем завершения в данной ситуации является выдача

информативного сообщения о невозможности работы программного продукта в текущем окружении. Однако для скрытия факта защиты системы с целью уменьшения попыток, и как следствие, вероятности взлома предложен альтернативный подход. Суть данного подхода основывается на том, что управление потоком выполнения передается в случайную позицию сегмента кода защищаемой системы. В этом случае, при исключении из числа возможных позиций исходной точки входа, такое действие повлечет за собой некорректное завершение родительского приложения. Для пользователя это будет отображаться просто как внутренняя ошибка приложения, а для потенциальных нарушителей скроет факт защиты.

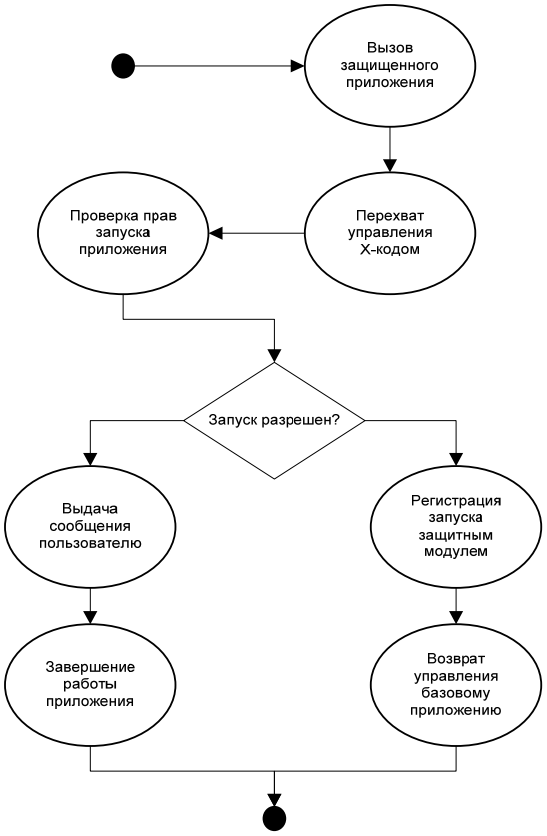

Рисунок 1 – Диаграмма деятельности системы защиты ПП

С целью повышения надежности, система была снабжена превентивным механизмом диагностики нарушений в системе защиты. Алгоритм работы подсистемы превентивной диагностики основан на применении параметрического контроля. Суть предлагаемого метода параметрического контроля заключается в следующем. Каждому i-му критерию защиты ставится в соответствие заданный коэффициент (KSi), характеризующий сложность взлома (стойкость) данного критерия. Вместе с лицензией пользователю выдается «кредит доверия» – коэффициент, характеризующий поведение пользователя (KD), при первом обращении клиента равный некоторому начальному значению KD_0. На основе информации о типе лицензии для каждого случая генерации лицензии рассчитывается порог ее взлома – коэффициент β, который рассчитывается как сумма стойкостей используемых критериев:

$$\beta = \sum KS_i, i = \overline{1,n}, \quad (2)$$

где n – число критериев используемых в лицензии.

Порог лицензии характеризует максимальное воздействие на механизмы защиты ПП, при котором продукт остается гарантированно защищенным. Когда система обнаруживает попытку взлома некоторых критериев, то вычисляется уровень потенциальной угрозы (α):

$$\alpha = \sum KS_j, j = \overline{1,m}, \quad (3)$$

где m – число критериев подвергаемых взлому.

В случае, если (β – α) ≥ KD, то система выдает предупреждение пользователю «о потенциальной угрозе взлома и возможном отказе в использовании приложения». Если (β – α) < KD, то система блокирует работу защищаемого приложения. При попытке взлома также оповещается центральная база данных, хранящая информацию о защищенных приложениях, и в дальнейшем при выдаче лицензии пользователь получит пониженный «кредит доверия».

Как было определено ранее, одним из важных свойств системы защиты является её возможность применяться в различных условиях для различных видов программного обеспечения. Эти требования были одними из базовых при разработке программного комплекса защиты. Для достижения поставленной задачи в системе были реализованы следующие свойства: используемый механизм защиты применим к большинству программного обеспечения, наличие гибкой системы настройки критериев защиты, минимальная зависимость от среды (машины) исполнения. Возможность применения механизма защиты к различным типам программного обеспечения обусловлен тем, что принцип работы всех исполняемых файлов для

платформы Win32 един, и, как следствие, есть возможность унифицировать методы защиты и распространить их на все Win32-приложения. На данном этапе идет исследование работы механизма на 64-битных платформах. Как уже указывалось ранее, система защиты обладает возможностью устанавливать различные критерии защиты и их комбинации, что позволяет гибко настраивать ее на нужды пользователя. За счет отсутствия дополнительных ресурсов, которые могут быть зависимы от рабочего места, разработанная система защиты минимизирует требования к среде исполнения.

ОСОБЕННОСТИ РЕАЛИЗАЦИИ ПРОГРАММНОГО КОМПЛЕКСА ЗАЩИТЫ ПРИЛОЖЕНИЙ

Разрабатываемый интегрируемый комплекс защиты программных продуктов реализован в виде программного комплекса «License Control» (LcCtl), который состоит из следующих модулей:

- *ICtl (XCode)* – модуль проверки, представляет собой исполняемый код, который внедряется в защищаемое приложение. Код ICtl организован в виде дополнительной секции кода и содержит в себе все данные, необходимые для его нормального функционирования. Часть данных, находящихся в секции, формируется индивидуально, в зависимости от конкретного защищаемого файла, во время внедрения кода.
- *LcCtl (Injector)* – модуль защиты, обеспечивает внедрение кода ICtl в тело защищаемого приложения. Детали реализации могут варьироваться в зависимости от используемой платформы и технических характеристик рабочей станции лицензирования.
- *LcCtl Admin* – модуль администрирования, представляет собой приложение, обеспечивающее удобный интерфейс для настройки защиты родительского приложения (выбор родительского приложения, выбор критериев и т.д.).
- *LcInstall* – модуль инсталлятор, представляет собой приложение, позволяющее устанавливать и конфигурировать комплекс LcCtl на компьютере конечного пользователя в автоматическом режиме.

Схема взаимодействия описанных модулей представлена на рисунке 2 в виде диаграммы развертывания.

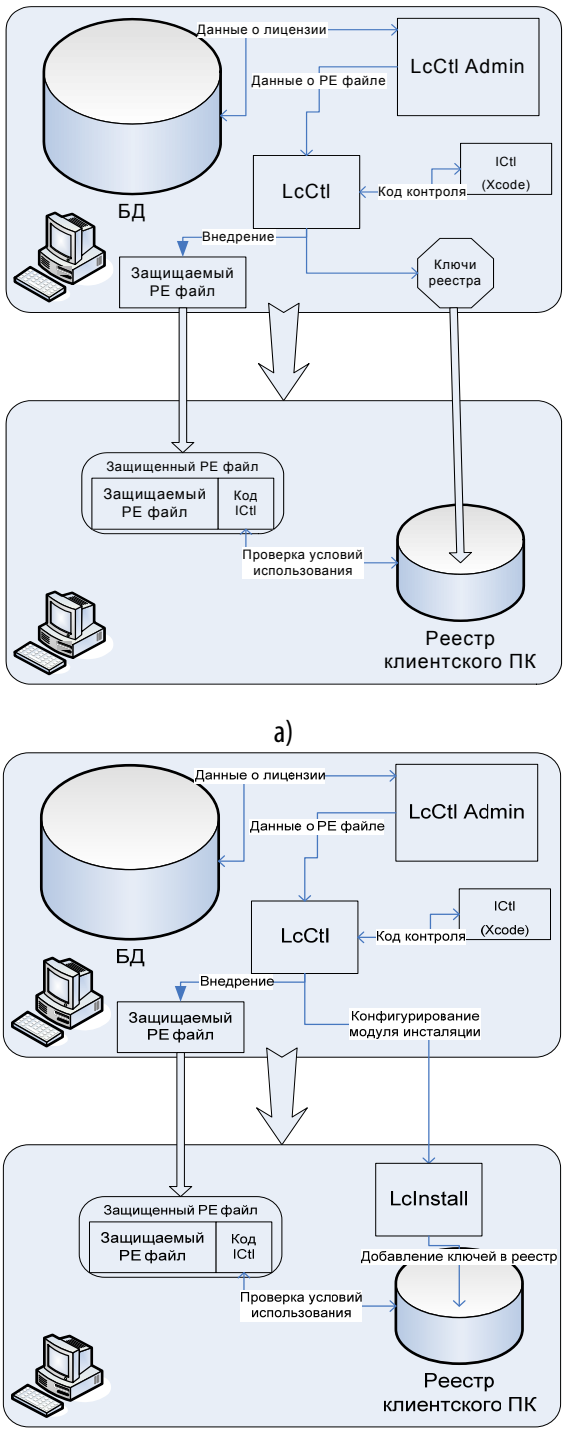

Рисунок 2 – Схема взаимодействия модулей системы: а) первый вариант установки 1; б) второй вариант установки.

Система позволяет настраивать используемое программное обеспечение различными способами. Конфигурация защитных ограничений системы производится до

этапа установки и эксплуатации. Установка системы возможна в двух вариантах. Первый подход предусматривает конфигурирование системы, внедрение исполняемого кода ICtl в защищаемое приложение и генерация ключей для системного реестра Windows, содержащие данные, необходимые для полноценного функционирования системы. При таком подходе есть возможность использовать собственный пакет установки, дополненный необходимыми ключами (см. рис. 2а). Второй подход в отличие от первого основан на конфигурировании модуля инсталляции LcInstall, который в дальнейшем позволяет производить инсталляцию защищенного продукта и настройку компьютера конечного клиента в автоматическом режиме (см. рис. 2б).

Поскольку система обладает широкими возможностями по конфигурированию и предоставляет средства для защиты различного программного обеспечения для большого числа пользователей, это предполагает хранение и обработку больших объемов данных. В реализованном программном комплексе вся информация о клиентах, использующих защищенные продукты, защищаемых продуктах, а также сведения о конфигурации каждого продукта хранятся в базе данных. В сочетании со встроенной системой отчетов, это позволяет эффективно управлять работой комплекса. Для обеспечения максимального быстродействия системы при возможных больших объемах данных была выполнена работа по оптимизации модуля базы данных. В процессе оптимизации была спроектирована соответствующая внутренняя структура базы данных и подобраны технические средства её реализации.

ВЫВОДЫ

В процессе создания комплекса был разработан и программно реализован метод внедрения защитного кода в исполняемый файл для организации разграничения доступа.

В работе были получены следующие результаты:

- достигнута полная интеграция кода защитного модуля и родительского приложения, что существенно уменьшает возможность потенциального взлома;
- предложена двойная система защиты информации о выданной лицензии, что практически исключает возможность взлома и изменения этой информации. При условии, что сама лицензия содержит минимально возможный полный набор информации, позволяющий гарантировать защиту приложения;
- предложен механизм легкого интегрирования модуля администрирования в различные приложения, что позволяет создавать интерактивные распределенные системы управления приложениями;
- выполнена оптимизация внедряемого кода модуля защиты, что позволяет минимизировать время задержки при загрузке приложения, которое необходимо для проверки критериев;
- предложены гибкие схемы формирования критериев защиты на использование родительских программных продуктов, а также различные варианты инсталляции системы защиты, которые предоставляют возможности для применения программного комплекса LcCtl в различных условиях.

Предложенная система защиты информационных систем «License Control» решает задачу обеспечения интеллектуальной собственности разработчика программного обеспечения и предотвращения несанкционированного использования ПП. В ходе компьютерных экспериментов были подтверждены выдвигаемые гипотезы о надежности предлагаемых решений. Механизм самодиагностики системы защиты находится в стадии опытной эксплуатации группой независимых специалистов по взлому ПП.

ЛИТЕРАТУРА:

1. Russinovich M. Vnutrennee ustrojstvo Microsoft Windows: Windows Server 2003, Windows XP i Windows 2000. Izdanie 4. Master-klass /M. Russinovich, D. Solomon. Per. s angl. – M.: Izd-vo «PITER», 2008. – 992 s.
2. Kasperski K. Tehnika otladki programm bez ishodnyh tekstov. – SPb.: Izd-vo «BHV Peterburg», 2005. – 832 s.
3. Microsoft Corporation. IMAGE_SECTION_HEADER structure. MSDN Library (WWW document) URL: http://msdn.microsoft.com/en-us/library/ms680341.aspx

Рецензент: *д.т.н., проф. Каргин А.А., Донецкий национальный университет, Донецк.*