$$k_1 = \frac{P(D_{T1}, 1)}{P(1)}, \quad k_2 = \frac{P(D_{T1}, 1)}{P(D_{T1})}. \qquad (19)$$

Коэффициент k_1 характеризует коэффициент уверенности эксперта в результатах оценки знаний преподавателем, k_2 – в результатах теста.

Байесовская стратегия предъявления вопросов обучаемому обеспечивает минимум потерь при автоматизированном контроле знаний в рамках всякого репрезентативного подраздела базы вопросов. Для минимизации потерь при работе со всей базой подразделы можно, очевидно, выбирать произвольным образом, поскольку суммарное число условно исключаемых вопросов на каждом шаге тестирования обучаемого не зависит от порядка рассмотрения подразделов.

Для выбора репрезентативного подраздела тематического раздела гипертекста G необходимо найти в разделе R подраздел T, соответствующий упорядоченной паре вопросов (u, v), такой, что разность числа вопросов, которые расположены выше вопроса u ($|D_{T1}|$), и математического ожидания количества верных ответов в множестве D_T минимальна:

$$\min_{u \in D_R} \left(|D_T| \cdot P(D_{T1}) - |D_{T1}| \right), \qquad (20)$$

где

D_R и D_T – множества всех вопросов раздела R и подраздела T соответственно.

Литература

[1].Петрушин В.А. Интеллектуальные обучающие системы: архитектура и методы реализации (обзор) // Известия Академии наук. Техническая кибернетика. - 1993. - №2. - С. 164-189.

[2].Шибут М.С., Ярмош Н.А. Методика проектирования обучающих курсов средствами автоматизированной системы АОСПроект. – Минск: Институт технической кибернетики НАН Беларуси, 1999. - 42 с.

[3].Липницкий С.Ф., Ярмош Н.А. Моделирование интеллектуальных процессов в инженерных информационных системах. -Минск: Беларуская навука. -1996. 222 с.

ТЕХНОЛОГИИ ЗАЩИТЫ ПО ОТ НЕЛИЦЕНЗИОННОГО ИСПОЛЬЗОВАНИЯ

Ю.В. Климец[1], А.А. Иванюк[1], В.Н. Ярмолик[2]

[1] - Кафедра программного обеспечения информационных технологий, Белорусский государственный университет информатики и радиоэлектроники, ул. П.Бровки, 6, Минск, 220600, БЕЛАРУСЬ, тел. +375 (17) 239-80-20, klimets@bsuir.unibel.by

[2] – Communication Theory, Department of Electrical and Information Engineering, University of Wuppertal, Rainer-Grunter-Strasse 21, 42119 Wuppertal, GERMANY, yarmolik@uni-wuppertal.de

Аннотация

В данной статье рассматриваются основные принципы построения современных систем защиты программного обеспечения (ПО) от нелицензионного использования. Рассматриваются различные варианты построения защиты. Приводятся рекомендации по встраиванию элементов защиты в разрабатываемые программные продукты.

1. Введение

Постоянное расширение рынка телекоммуникационных технологий, программного и аппаратного обеспечения определяет актуальность систем защиты цифровой интеллектуальной собственности. Как известно, в России и странах СНГ ярко выражена проблема "правового нигилизма" в отношении интеллектуальной собственности, которая наиболее ярко проявляется в области информационных технологий. Вследствие этого цифровая интеллектуальная собственность во всех формах (программное и аппаратное обеспечение, базы данных, цифровые музыка, видео и графика, электронные книги и др.) не защищена от нелегального копирования и использования в коммерческих целях третьими лицами, несмотря на наличие действующих юридических положений в этой области. Особенно остро данная проблема проявляется в области защиты программного обеспечения от нелицензионного использования. Так, по данным аналитических компаний потери от нелегальных копий программ в 2000 году только в России составили более 1 млрд. долларов [1].

Необходимость использования систем защиты ПО от нелицензионного использования обу-

словлена рядом проблем, среди которых следует выделить:

- незаконное использование программных решений и алгоритмов, являющихся интеллектуальной собственностью автора, при создании аналогов продукта;
- несанкционированное использование программного обеспечения;
- несанкционированная модификация программного обеспечения с целью внедрения программных злоупотреблений;
- незаконное распространение и сбыт программного обеспечения, и многое другое.

Нарушение законов о защите интеллектуальной собственности наносит материальный и моральный ущерб государству, разработчикам и производителям программного обеспечения. Наиболее сильно от этого страдают отечественные разработчики программного обеспечения, которые не имеют ресурсов для защиты, продвижения и продаж своих продуктов по демпинговым ценам в отличие от зарубежных компаний, действующих на отечественном рынке.

Далее в статье будут рассмотрены основные варианты и принципы построения систем защиты ПО от нелицензионного использования, а также приведены рекомендации по встраиванию элементов защиты в разрабатываемые программные продукты.

2. Обзор основных методов защиты ПО

Вопрос о применении и выборе методов защиты необходимо рассматривать еще на начальной стадии проектирования и разработки программного обеспечения с учетом целей защиты. Обычно методы защиты используются с целью:

- предотвратить пиратское копирование и тиражирование программ и цифровых произведений;
- обеспечить целостность программ и цифровых произведений, т.е. предотвратить внесение неавторизованных изменений;
- обеспечить соблюдение пользователем условий лицензионного соглашения.

Для защиты программного обеспечения обычно применяются следующие технологии:

- регистрационные номера и ключевые файлы;
- защита носителей (дискеты, компакт-диски) от копирования;
- программно-аппаратная защита с использованием электронных ключей.

Рассмотрим подробнее каждую из технологий.

2.1. Регистрационные номера и ключевые файлы

Данный метод подразумевает связь каждой копии программы с некоторым регистрационным номером и обычно используется для защиты условно-бесплатных программ. Смысл защиты состоит в сложности получения регистрационного номера программы из регистрационных данных пользователя, при этом связка «номер-данные» обычно используется для однозначного дешифрования фрагментов кода или данных. Однако, стойкость подобной защиты к взлому очень низка. Например, одни и те же регистрационные данные могут использоваться для запуска произвольного числа копий программы.

Некоторым улучшением данного метода защиты является «привязка» регистрационного номера к уникальным характеристикам компьютера, например, номеру тома жесткого диска или идентификационного номера процессора. Однако из-за особенностей реализации механизма защиты он является самым неудобным для конечных пользователей, так как программу, защищенную подобным образом, нельзя перенести на другой компьютер, возникают трудности при модернизации самого компьютера и т. п.

2.2. Защита носителей от копирования

Данный метод предполагает невозможность физического создания точной копии носителя, на котором записан защищаемый программный продукт, который, в свою очередь, привязывается к «нестандартности» носителя. Для получения ключевых дисков используется целый ряд технологий, начиная от нанесения лазерной метки на диск и заканчивая записью неустойчиво читаемых фрагментов.

В настоящее время подобные методы в силу низкой стойкости защиты и сложности с распространением и использованием защищенного ПО практически не используются.

2.3. Программно-аппаратная защита с использованием электронных ключей

На сегодняшний день это наиболее надежный метод защиты тиражируемого ПО средней и высшей ценовой категории. При правильной реализации защиты он обладает высокой стойкостью к взлому и не накладывает ограничений на использование легальной копии программы. Применение этого метода экономически оправдано для программ стоимостью свыше $80, так

как использование даже самых дешевых электронных ключей увеличивает стоимость каждой копии ПО на $10-15.

Электронными ключами обычно защищают так называемый "деловой" софт: бухгалтерские и складские программы, правовые и корпоративные системы, строительные сметы, САПР, электронные справочники, аналитический софт, экологические и медицинские программы и т. п. Затраты на разработку таких программ велики, и, соответственно, высока стоимость софта, поэтому ущерб от пиратского распространения значителен. В этом случае электронные ключи являются оптимальным средством защиты.

Далее будет более подробно рассмотрена именно защита с использованием электронных ключей, как наиболее мощная и совершенная.

3. Варианты защиты

Есть два способа защиты с использованием электронных ключей - автоматическая (или навесная) защита и защита при помощи функций API. В первом случае защищается исполняемый программный модуль, во втором - функции защиты до компиляции встраиваются в исходный код программы.

При автоматической защите исполняемый модуль защищаемой программы обычно шифруется и помещается в своеобразный «конверт», который при запуске модуля получает криптоключ для дешифрования из электронного ключа. При использовании криптостойких алгоритмов шифрования запустить модуль на выполнение не имея электронного ключа невозможно. Однако, однажды запустив программу при наличии оригинального, взломщик может сохранить на диске исполняемый модуль уже после его дешифрования, и далее запускать его уже без ключа. Существует множество программ, позволяющих автоматизировать этот процесс, поэтому автоматическая защита является приемлемым решением только в том случае, когда важна скорость построения защиты. При проектировании более совершенных защит необходимо использовать API для доступа к функциям электронного ключа.

Перед тем, как описывать процесс проектирования защиты с использованием API, следует подробнее рассмотреть и проанализировать методы «взлома» защиты.

4. Как «ломают» программы

Основная цель любого взлома программы - заставить корректно работать нелицензионную копию программы. Для этого существуют два способа: пройдя по всем возможным ветвям программы, сохранить все запросы/ответы к электронному ключу с целью дальнейшей их подмены при помощи эмулятора ключа, и "ручной" взлом, когда взломщик находит все места в программе, в которых осуществляется обращение к ключу, и, анализируя логику работы программы, модифицирует код программы соответствующим образом.

"Ручной" взлом подразумевает интерактивное взаимодействие хакера с взламываемой программой. Обычно для этого используются два основных инструмента: интерактивный дизассемблер (IDA - Interactive DisAssembler) и отладчик защищенного режима Soft-ICE. Кроме этого, могут использоваться различные вспомогательные инструменты, например FileMon, RegMon, VxDMon, позволяющие отслеживать обращения программы к файлам, реестру и VxD-сервисам, соответственно; HIEW (BIEW, QVIEW и др.), позволяющие просматривать и редактировать код программы и др.

Рассмотрим вкратце основные возможности IDA и Soft-ICE, которые позволят более целенаправленно проектировать защиту приложения от взлома.

4.1. Интерактивный дизассемблер IDA

IDA представляет собой очень мощный инструмент для дизассемблирования и изучения логики работы программ. Из основных возможностей следует выделить возможность интерактивного внесения пользователем изменений в образ загружаемого файла, назначение имен функциям, меткам и данным, автоматическое распознавание большинства стандартных функций и вызовов, а также наличие встроенного СИ - подобного скриптового языка, позволяющего автоматизировать процесс дизассемблирования программы.

Именно интерактивность IDA не позволяет использовать простейшие методы против дизассемблирования, например, внесение попарно ссылающихся вызовов или примитивного шифрования с помощью операций XOR или ADD.

4.2. Отладчик Soft-ICE

Отладчик защищенного режима Soft-ICE работает на 0-м кольце защиты операционной

системы и позволяет управлять выполнением взламываемой программы не только на программном уровне, но также и на аппаратном, используя регистры аппаратной отладки процессора. Обычно при отладке в необходимых местах программы устанавливаются точки останова по выполнению и по доступу к памяти (портам ввода/вывода). Рассмотрим подробнее процесс обработки каждого из типов точек останова.

Для останова по выполнению процессор предоставляет два средства - это исключение №3 и флаг трассировки TF. В первом случае отладчик заменяет код команды, на которую устанавливается точка останова, на код 0xCC. По достижении данной команды процессор сгенерирует исключение №3. Обработчик этого исключения принадлежит отладчику, который, получив управление, предоставляет пользователю возможность просмотра и изменения текущего контекста программы. После продолжения работы отладчик эмулирует выполнение замененной команды и передает управление обратно программе.

При использовании для отладки флага необходимо установить флаг TF регистра флагов процессора FLAGS, после чего процессор на каждой команде будет генерировать исключение №1, которое обрабатывается отладчиком.

Для установки аппаратных точек останова по доступу к памяти используются отладочные регистры процессора DR0-DR3, DR6, DR7. В регистрах DR0-DR3 устанавливаются линейные адреса памяти (портов ввода/вывода), доступ к которым будет вызывать исключение. Регистры DR6 и DR7 используются для управления точками останова (разрешение/запрещение точек останова, установка типа точек останова и др.) Доступ к регистрам DR0-DR3, DR6, DR7 считается привилегированной операцией и возможен только с 0-го кольца защиты.

5. Технологии построения защиты

Для построения эффективной защиты используют следующие методы и технологии [3]:

- использование шифрования фрагментов кода и наиболее важных данных;
- использование виртуальных машин для выполнения операций обработки данных;
- использование случайных запросов к электронному ключу.

Рассмотрим подробнее каждый из методов.

5.1. Шифрование фрагментов кода и данных

Шифрование фрагментов кода используется для усложнения анализа принципов работы защиты [4]. Для построения надежной защиты рекомендуется не хранить в открытом виде криптоключ, а получать его путем одной или серии посылок к электронному ключу. Для еще большего усиления защиты можно воспользоваться "каскадным" шифрованием, когда последовательно расположенные фрагменты кода последовательно, начиная с последнего, шифруются различными криптоключами (и/или различными криптоалгоритмами). Тогда для «снятия» защиты взломщику придется последовательно, переключаясь от Soft-ICE к IDA и обратно, получить все ответы от электронного ключа и написать скрипты, выполняющие дешифрование фрагментов кода.

Шифрование данных следует применять для скрытия запросов и эталонных ответов электронного ключа, что предотвратит взлом защиты путем переноса этих данные в эмулирующую программу. Кроме этого, запросы и ответы ключа следует размещать более чем в одном месте внутри кода приложения.

5.2. Виртуальные машины

Идея заключается в использовании виртуальной (программной) реализации некоторой аппаратной платформы, имеющей собственный набор инструкций и архитектуру для обработки посылок и ответов от электронного ключа. При использовании виртуальной машины даже простейшая операция шифрования посредством наложения маски на данные будет выглядеть абсолютно неочевидно в объектном коде программы. Использование виртуальной машины существенно затрудняет анализ логики работы программы. Взломщику для снятия защиты с программы потребуется разобраться с архитектурой виртуальной машины.

5.3. Использование случайных запросов

Данный метод защиты нацелен на противодействие эмулятору электронного ключа и не позволяет использовать предварительно полученную базу данных записанных коммуникаций для подмены аппаратного ключа. Принцип защиты состоит в перемешивании ложных и истинных запросов к электронному ключу в коде приложения. Защищенное приложение достаточно часто генерирует запросы, не производя анализ откликов на них или производя лишь видимость обработки ответов. Но время

от времени приложение посылает истинный запрос, соответствующий какому-либо из известных ему откликов, для проверки присутствия оригинального ключа.

6. Лицензирование

Изначально электронные ключи предназначались исключительно для защиты программ. Однако потенциал современной программно-аппаратной защиты позволяет применять их для реализации различных форм лицензирования:

- предоставление демо-версий и пробных версий программ;
- аренда и лизинг программного обеспечения;
- помодульная продажа программного обеспечения;
- удаленное обновление ПО;
- сетевое лицензирование.

7. Заключение

В данной статье были рассмотрены основные принципы организации и встраивания защиты от нелицензионного использования в разрабатываемые программные продукты. Было проведено сравнение различных вариантов построения защиты, даны рекомендации по повышению эффективности защиты.

Литература

[1].Сайт «Правовая охрана, защита и управление цифровой интеллектуальной собственностью» - http://www.intellect.vsu.ru/.

[2].Официальный сайт компании «Актив» - www.guardant.ru.

[3].К. Касперски «Техника и философия хакерских атак». М.: Солон-Р, 1999, 272 с.

[4].Официальный сайт компании «Rainbow» - www.rainbow.com.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ КОМПАНИИ

Н.И.Сергеева, С.А.Краснов, А.Ф.Сергеев

Аннотация

Автоматизированная система управления компании позволяет решить проблему отпуска нефтепродуктов за наличный и безналичный расчет по единой смарт-карте и обеспечивает частично возврат денежных средств за нефтепродукты поставляемые в Белоруссию.

1. Назначение системы

Система предназначена для автоматизации розничной продажи нефтепродуктов и сопутствующих товаров за наличный расчет и по безналичному расчету при помощи пластиковых смарт-карт в сети АЗС региональной организации нефтепродуктообеспечения. Система обеспечивает возможность обслуживания на АЗС по картам из других регионов на основе межрегиональных взаиморасчетов. Целью создания системы является:

- получение дополнительной прибыли за счет увеличения объемов реализации и привлечения денежных средств клиентов по предоплате;
- автоматизация контроля и учета отпуска нефтепродуктов и сопутствующих товаров на АЗС;
- учет потребления топлива клиентами системы;
- автоматизация учета поступления, остатков и реализации нефтепродуктов на АЗС;
- автоматизация централизованного контроля за работой сети АЗС;
- сокращение расчетов наличными деньгами;
- повышение культуры обслуживания клиентов.

2. Характеристика системы

1. Функции системы.

- Эмиссия различных типов смарт-карт: дебетных - типа "электронный кошелек" или литровых, корпоративных - с автопополнением денежного или литрового ресурса.
- Продажа нефтепродуктов и сопутствующих товаров за наличный расчет и по смарт-картам.
- Обслуживание смарт-карт из других регионов на основе межрегиональных взаиморасчетов.
- Учет отпуска нефтепродуктов на АЗС в объемном и денежном выражении, подготовка отчетных данных о реализации нефтепродуктов.
- Централизованный сбор информации от центра эмиссии и комплексов для автоматизированного отпуска нефтепродуктов, установленных на АЗС.
- Ведение базы данных состояния смарт-карт.
- Формирование "черного списка" недействительных карт.