Запропоновано метод захисту найбільш важливих ділянок коду комп'ютерної програми шляхом авторизації через Інтернет, заснований на використанні механізму шифрування. Наводиться опис протоколу і методика захисту платного функціоналу в програмному забезпеченні. Виконана програмна реалізація протоколу і на основі нього створено модуль захисту комп'ютерних програм

Ключові слова: захист комп'ютерних програм, піратство, метод авторизації, умовно, протокол, реалізація, шифрування

Предложен метод защиты наиболее важных участков кода компьютерной программы путем авторизации через Интернет, основанный на использовании механизма шифрования. Приводится описание протокола и методика защиты платного функционала в программном обеспечении. Выполнена программная реализация протокола и на основе него создан модуль защиты компьютерных программ

Ключевые слова: защита компьютерных программ, пиратство, метод авторизации, shareware, протокол, реализация, шифрование

УДК 004.056

МЕТОД АВТОРИЗАЦИИ ЧЕРЕЗ ИНТЕРНЕТ ДЛЯ ЗАЩИТЫ SHAREWARE ПРОГРАММ

Д. М. Андрущенко
Младший научный сотрудник, ассистент*
E-mail: andrush85@mail.ru
Г. Л. Козина
Кандидат физико-математических наук, доцент*
E-mail: ainc@ukrpost.net
*Кафедра защиты информации
Запорожский национальный технический университет
ул. Жуковского, 64, г. Запорожье, Украина, 69063

1. Введение

Производители программного обеспечения несут большие убытки из-за нелегального использования их продукции. Чаще всего юридические методы борьбы с правонарушителями не являются эффективными, поэтому для защиты своих интересов разработчикам целесообразно прибегать к техническим средствам защиты программного продукта от нелегального использования. К техническим методам защиты относят программные и программно-аппаратные средства, а также использование программ как онлайн-сервисов [1, 2].

Программные методы защиты обычно [2] подразумевают использование привязки программы к конфигурации оборудования, на котором ее установили. Привязка происходит в момент установки программного обеспечения и требует либо ввода лицензионного ключа, либо прохождения процедуры активации – получения одноразового кода (ключа), зависящего от конфигурации оборудования пользователя и пригодного для использования только на этом компьютере. В первом случае ничего не мешает пользователю незаконно распространять продукт вместе с лицензионным ключом. Во втором случае добросовестному пользователю необходимо согласиться с неудобствами, возникающими при каждой смене оборудования, на котором он использует программное обеспечение. Недобросовестный пользователь имеет возможность незаконно использовать и распространять программное обеспечение вместе с виртуальной машиной [3], программно имитирующей необходимое конфигурационное оборудование.

Программно-аппаратные средства защиты [1, 2] подразумевают проверку наличия некоторых аппаратных средств, которые поставляются вместе с программой и для которых невозможно либо очень трудно изготовить копию. Например, широко используют специально изготовленные оптические диски (CD, DVD) и аппаратные ключи, подключаемые через USB-порт. Такие методы считаются более надежными, чем программные, но они имеют существенные недостатки. Во-первых, в этом случае программное обеспечение можно поставлять только в «коробочной» версии, во-вторых, предлагаемое средство защиты может существенно увеличить стоимость программного продукта, в-третьих, пользователю придется согласиться с некоторыми неудобствами. В связи с этим данный метод защиты не пригоден для большинства недорогих и используемых в повседневной жизни программных продуктов.

В настоящее время известны [2] неоднократные случаи взлома защиты обоих типов и дальнейшего нелегального и беспрепятственного распространения программного обеспечения. Именно поэтому актуальными могут считаться исследования, направленные на поиск новых недорогостоящих методов защиты программного продукта от нелегального использования.

2. Анализ литературных данных

Более надежным средством от копирования программ можно считать исполнение их как онлайн-сервисов Software as a service («Программное обеспечение как услуга») [4], которые подразумевают исполнение

программного обеспечения на стороне производителя, не доступной для пользователя. Пользователь же может только ввести входные и получить выходные данные через веб-интерфейс. Однако, такой вид защиты малопригоден для программного обеспечения, которое требует больших объемов входных либо выходных данных, и в любом случае он не сможет заменить традиционные способы распространения программ.

В последнее время для разработчиков перспективным является предоставление условно бесплатной (Shareware) версии программы, когда пользователь может использовать полностью рабочую версию программы ограниченное время [5]. Либо часть функций программы отключено в бесплатной версии и подключается только после получения полной лицензии. Однако такая возможность чаще всего является слабым местом в защите программного обеспечения, поскольку ограничение на использование традиционно выполняется путем проверки промежутка между текущим временем и временем первого запуска программы, либо подсчета количества запусков программы. При этом результат сохраняется в памяти компьютера. В этом случае появляются так называемые «кряки» для сброса счетчика и обеспечивается возможность пользоваться программой неограниченное время.

Таким образом, проблема обеспечения технической защиты при разработке программного продукта остается до конца нерешенной и требует дополнительных исследований.

Наиболее перспективным для защиты недорогих условно-бесплатных программ является метод авторизации через Интернет [6]. Он подразумевает первоначальную активацию продукта на вычислительной машине пользователя, а также авторизацию пользователя на сервере разработчиков при каждом запуске программы [7]. Однако данный метод в настоящее время имеет очень малое распространение, и готовых решений в открытом доступе авторами найдено не было.

3. Постановка проблемы

Авторизация пользователя при каждом запуске программы позволяет разработчику следить за статистикой использования программы, выявлять случаи нарушения лицензий, лишать лицензий недобросовестных пользователей, а также гибко изменять лицензионную политику в соответствии со своими нуждами [7, 8].

При использовании условно-бесплатных программ могут быть введены ограничения: на количество запусков программы в месяц, на количество просмотренных и созданных документов за некоторый промежуток времени либо на запуск программы с нескольких компьютеров одновременно. Такие ограничения должны быть прописаны в лицензионном соглашении вместе с санкциями за их нарушения. Под санкцией может подразумеваться как полное, так и временное лишение лицензии. Проверка нарушения лимитов должна производиться в недоступном для пользователя модуле контроля лицензий, например, удаленном сервере, куда программа должна посылать данные и проверять наличие разрешения на запуск. Если, например, нелегальная копия программы окажется опубликованной в публичном месте, то лимиты, введенные ограничениями, достаточно быстро будут превышены в связи с использованием программы большим количеством пользователей и лицензия будет заблокирована.

Для построения защиты с введением ограничений, необходимо организовать безопасный обмен данными по открытому каналу связи между программой и модулем защиты. В связи с повсеместным распространением Интернета такой обмен можно организовать достаточно просто, не вызывая значительных неудобств у пользователей. Однако, данные пересылаемые по Интернету, может читать и изменять любой недобросовестный пользователь программы. Это должно учитываться при организации безопасного обмена данными. Кроме того, программа должна контролировать попытку изменения своего кода и посылать сообщение серверу при обнаружении таких действий.

Для реализации такой схемы безопасной передачи данных необходимо:

1) Разработать протокол обмена данными между программой и сервером.

2) Разработать способ защиты от нелегального использования платных функций программы и выполнить программную реализацию.

4. Протокол безопасного обмена данными между защищаемой программой и сервером

Авторами разработан протокол защиты программного обеспечения, который основан на использовании механизма электронной цифровой подписи [9]. Суть его состоит в следующем.

Пусть имеется защищаемая компьютерная программа *Prog*, установленная на компьютере пользователя *U*, и удаленный сервер *S*, принадлежащий разработчикам программы *Prog* либо их доверенному лицу.

Разработчик должен выбрать систему электронной цифровой подписи *C* и сгенерировать пару ключей – открытый ключ *e* и закрытый ключ *d*. Закрытый ключ *d* должен храниться на сервере *S*, а открытый ключ *e* – в приложении *Prog*. Перед первым запуском программы пользователь должен получить идентификатор (логин) *Id* и пароль *P*. Каждый раз, когда пользователь *U* пытается выполнить одно из действий, установленных разработчиком (например, запуск программы, создание, открытие или сохранение документа), программа *Prog* должна посылать запрос серверу *S* о возможности продолжить работу, совершив следующие шаги передачи данных (рис. 1):

1. В программе *Prog* генерируется случайное число *RND*.
2. В программе *Prog* вычисляется некоторое число *F* – привязка к программно-аппаратному обеспечению вычислительной машины, где она установлена.
3. Программа *Prog* передает данные *Id, P, RND, F* серверу *S*.
4. Сервер *S* проверяет возможность использования программы пользователем с идентификатором *Id*, паролем *P* и привязкой *F*.
5. В случае подтверждения возможности запуска программы *Prog*, сервер *S* вычисляет элек-

тронную цифровую подпись *C* как функцию от случайного числа *RND* и закрытого ключа разработчика *d* .

6. Сервер *S* отправляет значение *C* программному обеспечению *Prog*.
7. В программе *Prog* осуществляется проверка подлинности подписи сервера *C* с использованием известного открытого ключа *e*. Если подпись подлинная, то программа *Prog* продолжает выполняться, в противном случае завершает работу.

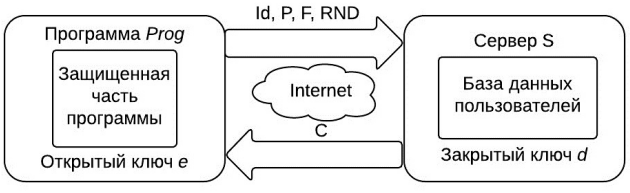


Рис. 1. Схема обмена данными между защищаемым программным обеспечением и удаленным сервером

5. Защита платного функционала в программе

При разработке программы ее платные функции помещаются в отдельную динамически подключаемую библиотеку DLL. Оригинальный файл библиотеки хранится только на удаленном сервере и не распространяется вместе с программой при ее приобретении.

В папке с программой находится только шифр данной библиотеки $module. Библиотека шифруется методом AES либо любым другим симметричным алгоритмом шифрования [10]. Для ее подключения к программе требуется ключ для расшифровывания Q. Однако ключ Q не хранится в самой программе, а хранится на удаленном сервере. Он может быть получен программой только по определенному запросу и успешной авторизации на сервере. Таким образом, нельзя будет получить исходный код библиотеки.

Для передачи данных между программой и сервером используется сессионный ключ, который генерируется в программе. Для безопасной передачи сессионного ключа от программы к серверу используется ассиметричный алгоритм шифрования (например, RSA). При этом секретный ключ D для расшифровывания храниться только на сервере, а открытый ключ E для шифрования вшивается в код программы.

При первом запуске программы, она проверяет наличие данного файла DLL в своей папке. Если его нет, то выполняются следующие действия:

1) Программа генерирует четный сессионный ключ S0, шифрует его открытым ключом E и отправляет на веб-сервер модулю «encrypted.php».
2) Веб-сервер дешифрует полученное сообщение ключом D и проверяет четность полученного сессионного ключа S0.
3) Если ключ четный, то веб-сервер шифрует библиотеку M сессионным ключом S0 алгоритмом AES и отправляет программе (сама библиотека M зашифрована методом AES при помощи ключа Q).
4) Программа получает зашифрованную библиотеку, расшифровывает ее сессионным S ключом и сохраняет в директорию.

Если файл существует в директории, то при первом и последующих запусках программы выполняются следующие действия:

1) Программа генерирует нечетный сессионный ключ S1, шифрует его открытым ключом E и отправляет на веб-сервер модулю «encrypted.php».
2) Веб-сервер дешифрует полученное сообщение ключом D и проверяет четность полученного сессионного ключа S1.
3) Если ключ нечетный, то веб-сервер с помощью сессионного ключа S1 шифрует алгоритмом AES ключ от библиотеки Q и отправляет сообщение программе.
4) Программа расшифровывает полученное сообщение при помощи сессионного ключа S1.
5) В итоге получается ключ AES для расшифровки библиотеки DLL, который не должен сохраняться на компьютере с запущенной программой.
6) Программа дешифрует библиотеку в защищенной области оперативной памяти и подключает ее для использования.
7) После остановки программы, расшифрованная библиотека стирается из оперативной памяти компьютера.

Для более надежной защиты программы необходимо включить также механизм привязки компьютерной программы к оборудованию. Это можно сделать следующим образом:

1) Программа вычисляет идентификатор оборудования компьютера ID.
2) Идентификатор ID в зашифрованном виде передается веб-серверу.
3) Веб-сервер дешифрует полученный идентификатор ID и сверяет с ранее зарегистрированным значением.
4) Если идентификатор ID разрешен, то запуск программы разрешается, иначе производится попытка смены оборудования.

6. Программная реализация протокола

В процессе реализации разработанного протокола клиентская часть была написана на языке программирования C# с использованием готовой крипто-библиотеки. Серверная часть было реализована на языке PHP. Блок-схемы алгоритмов клиентской и серверной частей приведены соответственно на рис. 2, *а*, *б*.

Фрагмент кода программы, реализующей клиентскую часть, приведен в листинге 1 (рис. 3).

Фрагмент кода программы, реализующей серверную часть, приведен в листинге 2 на рис. 4.

Таким образом, если пользователь решил с демо-версии программы перейти на ее полную версию (демо-версия, к примеру, может быть программой с урезанным функционалом), то ему необходимо нажать кнопку «получить лицензию» на сайте разработчика.

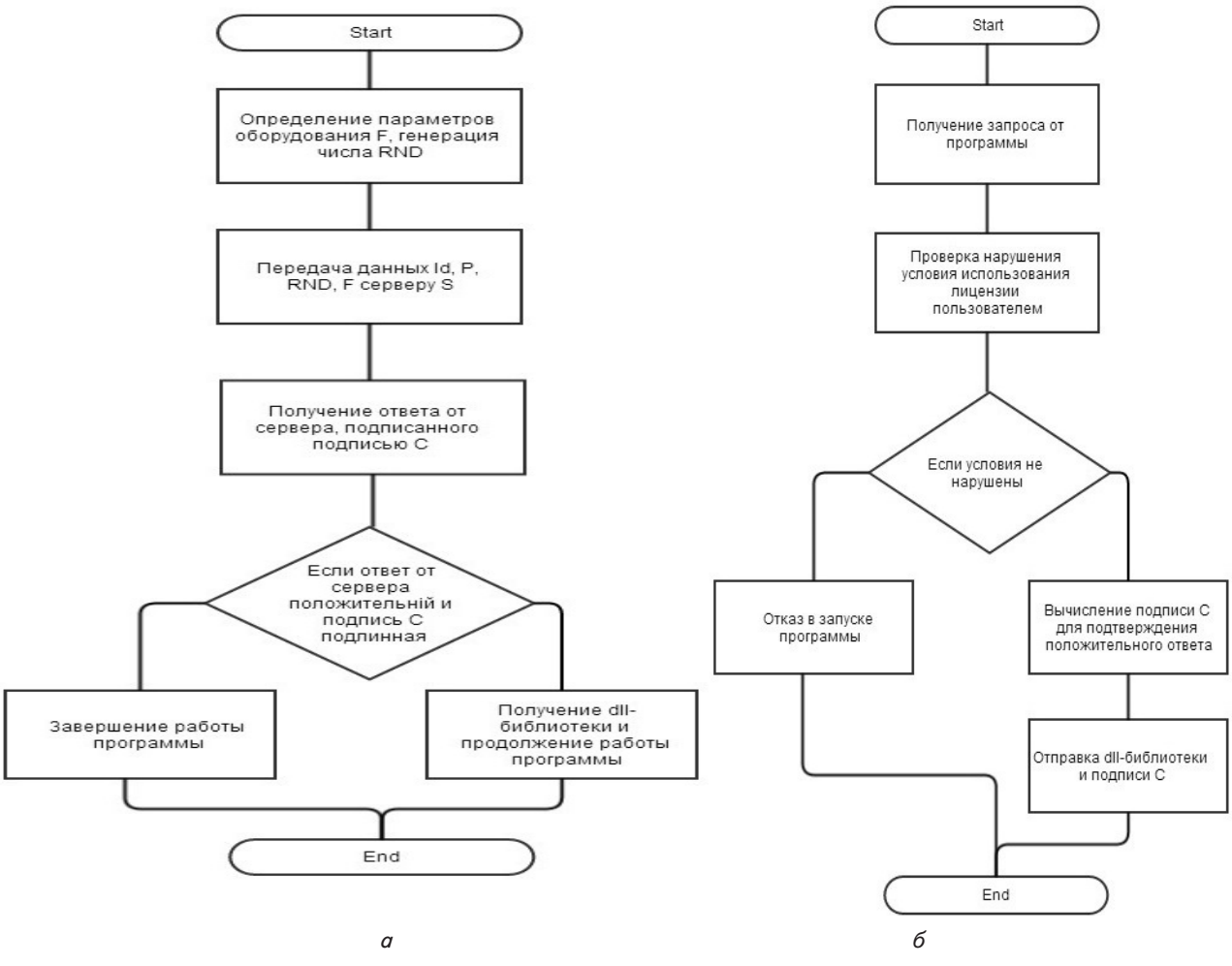
Далее ввести свои данные, в том числе логин и пароль для активации программы. После чего пользователь получает счет для оплаты лицензии. После оплаты пользователем счета производится активация путем привязки лицензии к оборудованию. Если в будущем потребуется сменить оборудование, то в автоматическом либо полуавтоматическом режиме будет произведена привязка к оборудованию повторно. Однако, по условию использования программы разработчику необходимо ограничить количество таких повторных привязок. Например, не чаще, чем 1 раз в сутки либо не более 10 раз в месяц. Также к одному аккаунту пользователя можно разрешать привязывать сразу несколько платформ. Например, рабочий компьютер, домашний компьютер и ноутбук. Это можно разрешать бесплатно в рамках одной лицензии либо требуя дополнительную стоимость за каждое дополнительное устройство (на усмотрение разработчика).

Кроме того, в защищаемой программе можно реализовать периодическую отправку запроса серверу. Тогда сервер сможет «следить» за программой и блокировать случаи ее нелегального использования. Поскольку в случае параллельно работающих программ на разных платформах одновременно начнут поступать запросы к серверу, то распространение программы внутри виртуальной машины будет невозможным.

Листинг 1 – фрагмент программной реализации клиентской части на языке C# (подключение платной библиотеки).

```
private object CryptLibrary(string name, params object[] args)
{
    Type type = assembly.GetType("CryptLibrary.CryptLibrary");
    object instance = A ctivator.CreateInstance(type);
    return type.InvokeMember(name,
        BindingFlags.InvokeMethod | BindingFlags.Instance |
BindingFlags.Public,
        null, instance, args);
}
private string ToMemory(string value)
{
    try
    {
        return (string)CryptLibrary("ToMemory", value);
    }
    catch{ NotLicensed(); Environment.Exit(1); }
}
```

Рис. 3. Фрагмент кода программы, реализующей клиентскую часть



а *б*

Рис. 2. Блок схемы алгоритмов клиентской и серверной частей: *а* – блок-схема алгоритма клиентской части; *б* – блок-схема алгоритма серверной части

Листинг 2 – фрагмент программной реализации серверной части на языке программирования PHP.

```
function addpadding($string, $blocksize = 32)
{
   $len = strlen($string);
   $pad = $blocksize - ($len % $blocksize);
   $string .= str_repeat(chr($pad), $pad);
  return $string;
}

if($type == "0")
{//Возвращаем библиотеку
  $iv = 'E465CDA32F584E81BF3B5DA781AC1126';
  $ciphertext = base64_encode(mcrypt_encrypt(MCRYPT_RIJNDAEL_256,
$sessionKey, addpadding($dllData), MCRYPT_MODE_CBC, $iv));
  echo $ciphertext;
}
else if($type == "1")
{//Возвращаем ключи
  $iv = 'E465CDA32F584E81BF3B5DA781AC1126';
  $ciphertext = base64_encode(mcrypt_encrypt(MCRYPT_RIJNDAEL_256,
$sessionKey, addpadding($dllKey.",".$dllIV), MCRYPT_MODE_CBC, $iv));
  echo $ciphertext;
}
```

Рис. 4. Фрагмент кода программы, реализующей серверную часть

7. Выводы

Таким образом, программная реализация позволила создать механизм защиты наиболее важных участков кода программы, которые располагаются внутри библиотеки DLL. Данный способ в отличие от существующих позволяет разработчику легко распространять программное обеспечение через Интернет, не опасаясь возможности появления нелегальных копий. Кроме того, у разработчика есть возможность следить за процессом использования предоставленных лицензий, выявлять правонарушения и блокировать лицензии. Разработчик также может предлагать несколько различных типов лицензий с различным функционалом программы. Серверная часть позволяет «следить» за такими действиями пользователя, как запуск программы, создание нового документа, использование обработчика данных, генерирование отчета и т. п. И блокировать работу программы в случае ее незаконного использования. Предоставление лицензий через Интернет обеспечивает пользователю возможность быстро продлить действие лицензии или заменить одну лицензию на другую.

В дальнейшем авторами планируется разработка более полного защитного комплекса для разработчиков программного обеспечения.

Литература

1. Erickson, J. Hacking: The Art of Exploitation, 2nd Edition [Text] / J. Erickson. – San Francisco: No Starch Press Inc, 2008. – 488 p.
2. Скляров, Д. В. Искусство защиты и взлома информации [Текст] / Д. В. Скляров. – СПб.: БХВ-Петербург, 2004. – 288 с.
3. Virtual machine [Electronic resource] / Available at: http://en.wikipedia.org/wiki/Virtual_machine
4. Software as a service [Electronic resource] / Available at: http://en.wikipedia.org/wiki/Software_as_a_service
5. Copy Protection [Electronic resource] / Available at: http://www.zappersoftware.com/copy-protection.html
6. Пат. 68078 Україна, МП(2012.01) G06F 12/00. Спосіб захисту ліцензійного програмного забезпечення від несанкціонованого використання [Текст] / Андрущенко Д. М., Козіна Г. Л, Карпуков Л. М. – заявник та патентовласник Запорізький національний технічний університет. – Опубл. 12.03.2012. Бюл. № 5/2012.
7. Андрущенко, Д. М. Метод защиты программного обеспечения [Текст] / Д. М. Андрущенко // Информационная безопасность регионов России (ИБРР-2011). VII Санкт-Петербургская межрегиональная конференция. Санкт-Петербург, 26-28 октября 2011 г.: Материалы конференции / СПОИСУ. – СПб., 2011. – С. 100–101.
8. Андрущенко, Д. М. Комп'ютерна програма "Захист програмних продуктів" [Текст] / Д. М. Андрущенко, Г. Л. Козіна // Свідоцтво про реєстрацію авторського права на твір №. 46740 – К.: Державний департамент інтелектуальної власності України. – Дата реєстрації: 11.12.2012.
9. Молдовян, Н. А. Теоретический минимум и алгоритмы цифровой підписи [Текст] / Н. А. Молдовян. – СПб.: БХВ-Петербург, 2010. – 304 с.
10. Paret, D. RFID and Contactless Smart Card Applications [Text] / D. Paret – Portland: Book News Inc., 2004. – 695 p. doi:10.1002/9780470016152