УДК 004.42

КОМПЛЕКСНАЯ КОЛЬЦЕВАЯ СИСТЕМА ЗАЩИТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ОТ НЕЛИЦЕНЗИОННОГО ИСПОЛЬЗОВАНИЯ

Частикова Вера Аркадьевна
к.т.н., доцент

Остапов Дмитрий Сергеевич
студент
Кубанский государственный технологический университет, Краснодар, Россия

В статье приведён алгоритм реализации многоуровневой системы защиты программного обеспечения от нелицензионного использования. Предложенная система защиты состоит из 4 уровней

Ключевые слова: ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ, ЗАЩИТА ИНФОРМАЦИИ, ЗАЩИТА ОТ КРЭКИНГА, ЗАЩИТА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ВНЕДРЕНИЕ КОДА, КРИПТОГРАФИЯ

UDC УДК 004.42

COMPREHENSIVE RING SYSTEM OF PROGRAM PROTECTION AGAINST USING BY ILLEGAL USERS

Chastikova Vera Arkadyevna
Cand.Tech.Sci., associate professor

Ostapov Dmitriy Sergeevich
student
Kuban State Technological University, Krasnodar, Russia

This article shows the algorithm of the realization of multilevel system of program protection against using by illegal users. The proposed system consists of 4 protection levels

Keywords: INFORMATION SECURITY, INFORMATION PROTECTION, PROTECTION AGAINST CRACKING, PROGRAM PROTECTION, CODE IMPLANTATION, CRYPTOGRAPHY

Проблема использования нелицензионных копий программ в настоящее время довольно актуальна, в том числе и в Российской Федерации. Разработчики в большинстве случаев защищают программное обеспечение (ПО) от взлома посредством сжатия или шифрования exe-файла (чтобы его невозможно было дизассемблировать) и ключей, которые находятся либо в exe-файле, либо в отдельном файле. Но, как показывает практика, подобный метод защиты нуждается в значительной доработке. Основная уязвимость состоит в том, что ключ, находящийся на компьютере пользователя, или какая-либо другая информация может быть изменена при помощи любого HEX-редактора.

Рассматриваемая в данной статье система защиты программного обеспечения от нелицензионного использования является комплексной и состоит из нескольких колец защиты.

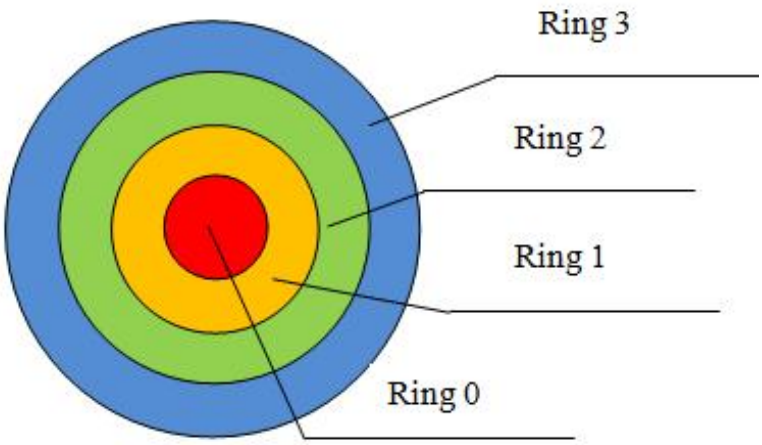

Рисунок 1 – Кольцевая система защиты

Кольцо Ring 0 представляет собой основную часть защиты. Оно осуществляет проверку того, используется ли данная платная версия программного обеспечения только одним компьютером. Ring 1 выступает в качестве защиты от отладки. Если хакер сможет обойти защиту кольца Ring 2, то Ring 1 не позволит злоумышленнику выполнить трассировку кода (или его части) программы в отладчике, что не даст возможности хакеру понять алгоритм работы защиты, и, следовательно, он не сможет написать crack для взлома защиты. Ring 2 показано в виде шифрования защищаемого файла с помощью специализированного программного обеспечения. Ring 3 должно быть представлено в виде участка кода, который будет постоянно перемещаться между специально отведёнными файлами и осуществлять проверку на целостность (неизменность) защищаемого файла. Код кольца Ring 3 должен перемещаться с места на место, чтобы хакер не смог по-отдельности анализировать файлы и писать crack к каждому из них.

Ring 0

Кольцо Ring 0 представляет собой основу защиты программы. Основной идеей работы Ring 0 являются проверки уникальных параметров пользовательского компьютера при каждом выходе в

Интернет: серийных номеров оборудования и индивидуального ключа программного обеспечения. При этом также осуществляется проверка IP-адреса: принадлежит ли IP одному региону или же через короткий промежуток времени IP изменяется с диапазона IP-адресов Приморского края на диапазон Краснодарского края и так далее. Данный алгоритм позволяет отслеживать несанкционированное использование одного ключа двумя и более людьми.

Предложенная авторами защита ключей от их несанкционированного использования состоит из четырёх этапов.

1) Обход сетевого экрана.
2) Генерирование первичного ключа и занесение его в базу данных.
3) Генерирование вторичного ключа и занесение его в базу данных.
4) Проверки.

Первоначально необходимо убедить пользователя зарегистрироваться. Регистрацию необходимо провести в диалоговом окне и именно при установке программы с использованием подключения к разным серверам для обхода сетевого экрана.

Далее необходимо осуществить генерацию ключей и проверки на уникальность оборудования компьютера.

Проверки

Для того чтобы отследить несанкционированное использование ключей, рекомендуется выполнить ряд действий.

1. При каждом первом подключении к Интернету, программа отправляет на сервер свой вторичный ключ.

2. С помощью секретного ключа вторичный ключ расшифровывается по алгоритму RSA. Получается строка типа «Фамилия Имя Отчество Первичный_ключ» (без кавычек).

3. Строка разбивается на 4 подстроки: фамилия, имя, отчество, первичный_ключ. Осуществляется проверка, соответствуют ли данные

подстроки одному пользователю. В случае соответствия происходит переход к следующей проверке, в случае несоответствия ставится пометка в базе данных о блокировке, пользователю посылается уведомление о том, что его ключ заблокирован, и выводится сообщение с просьбой связаться со службой технической поддержки.

4. Выполняется проверка, был ли заблокирован ключ. Если текущая проверка дала положительный результат – необходимо заблокировать программу, выдав сообщение с просьбой обратиться в службу технической поддержки, если отрицательный – осуществить переход к следующей проверке.

Пункты 4-5 посвящены предотвращению использования одного ключа несколькими пользователями.

5. Если не записан серийный номер модема, необходимо его записать. В случае записи следует сравнить его с тем номером, который записан в базе данных. Если серийный номер другой, следует записать второй серийный номер модема. Всего на каждого пользователя предоставлено 3 серийных номера для его модемов (рабочего, домашнего, дополнительного).

Если в течение одного дня пользователь выходил в Интернет, применяя 4 разных модема, ему выводится уведомление о том, что его ключ заблокирован, ставится соответствующая пометка в базе данных, и выдаётся сообщение с просьбой связаться со службой технической поддержки, если всё нормально – следует перейти к следующему шагу.

6. Для каждого региона существует определённый диапазон IP-адресов. При каждой проверке IP-адрес пользователя заносится в базу данных; всего пользователю предоставляется 3 варианта диапазона IP-адресов. При первой проверке из IP-адреса пользователя получаем диапазон, в котором он может находиться. При последующих проверках следует определить, лежит ли IP адрес пользователя в нужном диапазоне.

Если не лежит – осуществляется запись второго диапазона, если второй диапазон тоже записан, но IP-адрес не принадлежит первым двум - происходит запись в третий диапазон. В случае, если три диапазона заполнены, и это произошло в течение одной недели и, если при последующей проверке появляется IP-адрес, не принадлежащий первым трём, ставится соответствующая пометка в базе данных, и пользователю выдаётся сообщение о том, что его ключ заблокирован и ему необходимо обратиться в службу технической поддержки. Если же данная проверка прошла успешно, следует приступить к последней проверке.

Для того чтобы исключить возможность выхода в Интернет с одного модема (что может быть с лёгкостью произведено, например, на предприятии) необходимо реализовать следующую проверку.

7. В базу данных при первой проверке заносится серийный номер наиболее редко меняемых в ПК деталей: серийные номера материнской платы и процессора. Далее в базе данных создаётся поле, в котором отражается число смен серийных номеров процессора и материнской платы одновременно (первоначальное значение при первоначальной проверке устанавливается на нуль). В течение всего периода пользования данным ключом (12 месяцев) пользователю даётся возможность 3 раза входить в Интернет с неодинаковыми серийными номерами процессора и материнской платы одновременно. Если при очередном контроле будет обнаружено, что изменились серийные номера материнской платы и процессора, число смен серийных номеров материнской платы и процессора увеличивается на 1. Когда это число примет значение 3, будет произведена блокировка ключа, и пользователь получит соответствующее сообщение.

В пунктах 4-5 срок один день был выбран по ряду причин. Если взять слишком маленький срок, к примеру, один час, то могут возникнуть варианты несанкционированного использования ключей, которые система

не сможет отследить, например, если много пользователей будут запускать программное обеспечение с одним и тем же ключом и выходить в Интернет 1 раз в 2 часа. Если же взять слишком большой срок, например, одна неделя, то возникнут случаи ошибочного блокирования ключа «честного пользователя». Например, пользователь часто меняет своё место нахождения и за один месяц он может побывать в 10 городах России, осуществляя оттуда выход в Интернет. В таком случае система заблокирует его ключ, что существенно повлияет на имидж компании, которая будет иметь высокие риски потерять некоторую часть своих клиентов.

В пункте 7 предоставлена возможность смены процессора и материнской платы по следующим причинам:

1) это наименее редко меняемые детали;

2) вероятность того, что эти детали будут обе заменены в течение года 3 раза, мала. Скорее всего, это произойдёт в случае механического повреждения, например, если пользователь уронит компьютер, или в результате резкого скачка напряжения.

Если по какой-то причине при подключенном к Интернету компьютеру не удаётся подключиться к серверу, а это возможно только в том случае, если процесс соединения с сервером был заблокирован сетевым экраном, то следует заблокировать программу и попросить пользователя разрешить защитному модулю доступ к сети.

Важно отметить, что в Ring 0 должна быть реализована проверка на целостность Ring 3.

Ring 1

Кольцо Ring 1 производит защиту от отладчиков. Задача данного кольца защиты – обеспечить невозможность трассировки и анализа исходного кода в случае, если хакер справится с защитой кольца Ring 2.

Ring 1 может быть использовано с помощью защиты, основанной на применении флага трассировки, или на подсчёте количества тактов. Но гораздо эффективнее эти технологии использовать совместно.

Использование флага трассировки

Процессоры x86 содержат специальный трассировочный флаг TF (Trap Flag). Когда TF==1, после выполнения каждой инструкции генерируется специальное отладочное прерывание, обрабатываемое отладчиком[1,5]. Формально флаг трассировки является частью регистра флагов, и потому он может быть сохранён в стеке командами pushf/pushfd.

```
pushf; сохранение флагов в стеке, включая TF
pop EAX; помещение сохранённых флагов в EAX
and EAX, 100h; выделение флага трассировки
jnz EXIT; переход к метке EXIT (если TF==1)
```

Данный метод достаточно эффективен, но, в основном, для защиты ПО от неопытных хакеров. Однако настоящий профессионал заметит эту группу команд и сможет обойти её: достаточно лишь после условного перехода поставить точку останова, и выполнить программу без трассировки до этой точки. Поэтому для защиты от трассировки необходимо модифицировать данный код. Для этого следует самостоятельно установить флаг трассировки в «1» и передать управление SEH-обработчику, в котором будут дальнейшие инструкции по

выполнению программы. Если же программа будет запущена под управлением отладчика, то SEH-обработчик не будет вызван.

```
mov EAX,0
push offset SEH_address; помещение в стек адреса SEH-обработчика
push dword prt FS:[EAX]
mov FS:[EAX], ESP; регистрация нового SEH-обработчика
; далее - установка флага трассировки в «1»
pushf
pop EAX
mov AH,1
push EAX
popf
jmp Exit; если управление не было передано SEH-обработчику,
; то программа выполняется отладчиком
; SEH-обработчик. Может располагаться в любом месте программы
mov ESI, [ESP+0Ch]
assume ESI: ptr context; теперь ESI указывает на контекст регистров
mov [ESI].regEip, offset continue
mov EAX,0
ret
continue; если программа выполняется не отладчиком
; то следует продолжить её выполнение[5].
```

Подсчёт количества тактов

При запуске программы под отладчиком процессор должен произвести гораздо большее количество действий, чем без отладчика. Это связано с тем, что во время отладки хакер должен установить хотя бы одну

точку останова, и на некоторое время приостановить работу программы, иначе теряется весь смысл использования отладчика. В случае прерывания работы программы на 1 секунду процессор с тактовой частотой 2 Ггц выполнит 2000000000 операций. Идея данного метода заключается в подсчёте количества тактов процессора, затраченных на выполнение исходного кода с помощью ассемблерной инструкции rdtsc[4].

```
rdtsc
xor EAX, EAX; очистка содержимого EAX
;фрагмент кода, количество тактов при выполнении которого
;необходимо узнать
;рекомендуется использовать большой объём кода, чтобы быть
;уверенным, что хакер воспользуется точкой останова внутри
;рассматриваемого участка
rdtsc;после выполнения второй инструкции rdtsc, в EAX будет
;помещено количество тактов, выполненных процессором на замеряемом
;участке кода.
```

При разработке программного обеспечения необходимо произвести замер числа тактов, выполненных процессором на определённом участке кода. Если оказалось, что в ходе работы программы между двумя участками кода процессор совершил 2000 тактов, то после замера можно осуществить сравнение значения EAX со значением, полученным при лабораторных испытаниях. Однако эталонное значение числа тактов следует увеличить на число операций, реализуемых среднестатистическим процессором за 1 секунду, поскольку в практических условиях даже при выполнении программы не под управлением отладчика, процессор способен реализовать гораздо большее число действий. Это связано с работой сторонних программ, которые пользователь может использовать совместно с защищаемой (Winamp, Microsoft Word, Internet Explorer и др.), не имеющих отношения к отладке. Тем не менее, если будет использована

точка останова, и выполнение программы приостановится хотя бы на 1 секунду, процессор с тактовой частотой 2 ГГц совершит $2*10^9$ тактов.

Ring 2

Задача рассматриваемого кольца – обеспечить защиту от дизассемблирования. На сегодняшний день существует огромное количество уже готового программного обеспечения, позволяющего это сделать. Одним из лучших вариантов является программа ExeCryptor, которая может осуществлять многоуровневую защиту, поместив один зашифрованный блок внутри другого[2,7].

Если открыть защищаемую программу при помощи дизассемблера без использования защиты, осуществляемой при помощи ExeCryptor, то дизассемблер IDA Pro покажет содержимое кода.

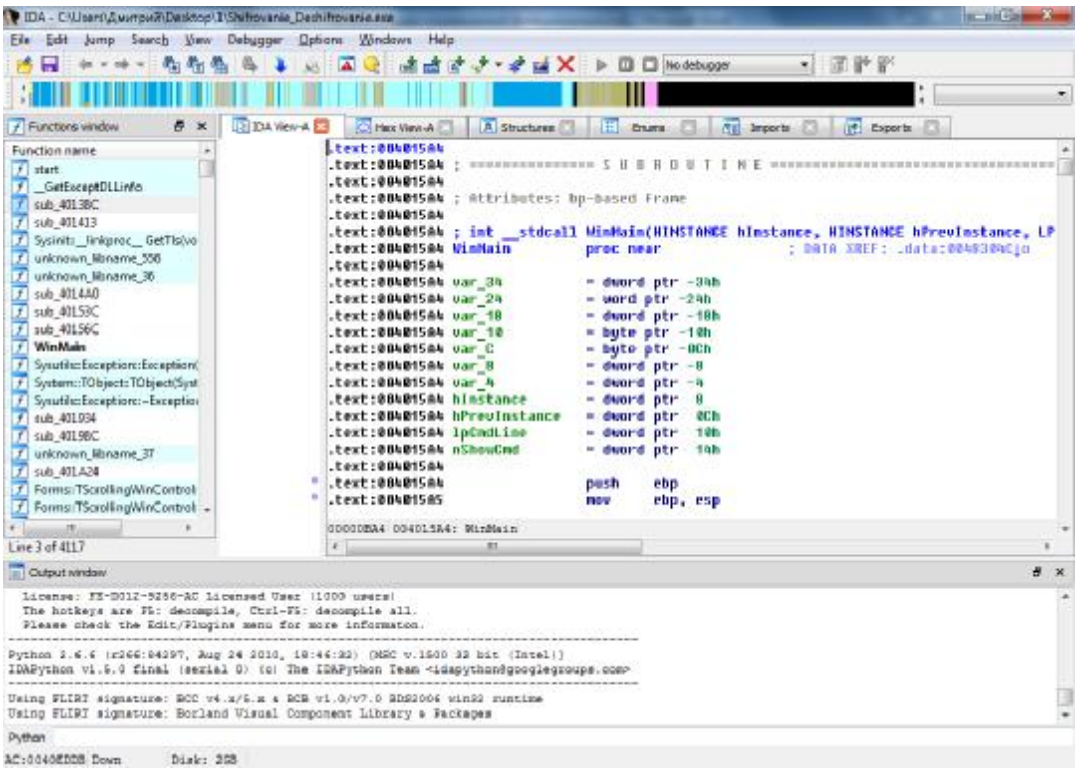

Рисунок 2 – Код exe-файла до обработки в ExeCryptor

Однако, если же открыть данный файл после обработки в ExeCryptor, то IDA Pro покажет совершенно другой набор операндов.

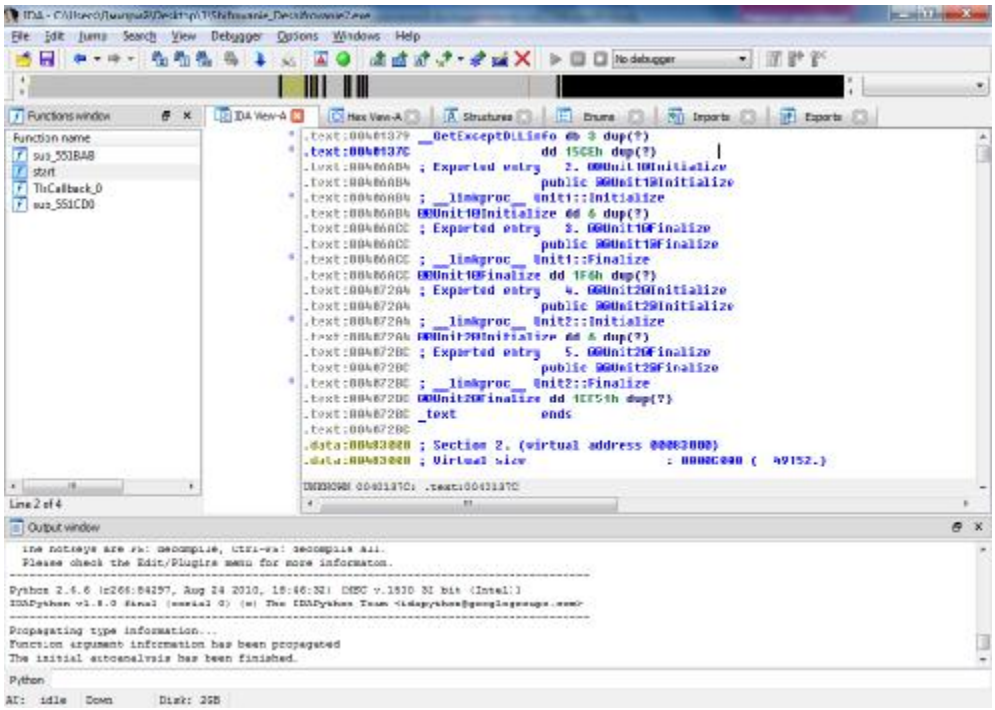

Рисунок 3 – Код exe-файла после обработки в ExeCryptor

Как видно из рисунка 3, дизассемблер смог преобразовать набор битов в ассемблерный код, однако, данный код лишён какого-либо логического смысла.

Ring 3

Ring 3 – заключительное кольцо защиты. Идея указанного кольца состоит в том, что оно должно проверять целостность защищаемого файла на предмет, был ли изменён хотя бы один его байт или нет. Для этого необходимо весь код рассматриваемого файла записать в 2 массива: double_array_sum, double_ mas_xor. Как видно из названий, в массивах будут содержаться контрольные суммы по операциям «+» и «xor. Были выбраны две операции (а не только «+») в связи с тем, что есть вероятность возникновения случая, когда даже при изменении нескольких байт защищаемого файла контрольная сумма по операции «+» останется неизменной. Можно рассмотреть пример. Пусть в защищаемом файле были байты 90h, 70h. Сумма этих байт 100h (сумму записываем в ячейку массива DWORD). Допустим, хакер изменил эти байты на 8Eh и 72h.

Сумма 88h и 72h так и останется 100h. Если проводить проверку только по операции «+», то изменений в файле найдено не будет. Поэтому следует проверить, будут ли найдены изменения в файле, если осуществить проверку по двум операциям (исходные байты 90h и 70h):

Таблица 1 - Контрольная сумма

Название операции	Сумма в исходном файле	Сумма в модифицированном хакером файле
+	100h	100h
Xor	0E0h	0FCh

В каждую ячейку массива должен записываться результат суммы 16777216 байт (2^{24} байт или 16 мегабайт); ячейка должна быть 4-байтовой. Ограничение в 24 байта было сделано для защиты от переполнения, так как максимальное значение суммы байт по операции «+» 2^8*2^{24} не должно превышать числа 2^{32}. В случае, если проверка на контрольные суммы покажет изменение файла, необходимо заблокировать защищаемую программу и вывести сообщение о том, что программа используется нелегально и пользователю следует купить её лицензионную копию.

Далее необходимо осуществлять перезапись кода по специальному алгоритму. Пусть вместе с защищаемым программным модулем присутствуют PE-файлы 1.exe, 2.exe … M.exe. Если записывать код в файл с помощью вставки в заголовок, расширения последней секции или создания новой секции, то, скорее всего, перезаписываемый код будет распознан антивирусным программным обеспечением как вирус-троян и будет удалён. Для исключения подобной ситуации с помощью команды NOP следует выделить место для перезаписываемого кода в файлах программы. Данная команда не выполняет никаких действий, а, следовательно, можно предварительно на этапе создания программ 1.exe,

2.exe … M.exe в коде программы выделить место для записи кода Ring 3. Схематично вышеизложенное изображено на рисунке 4:

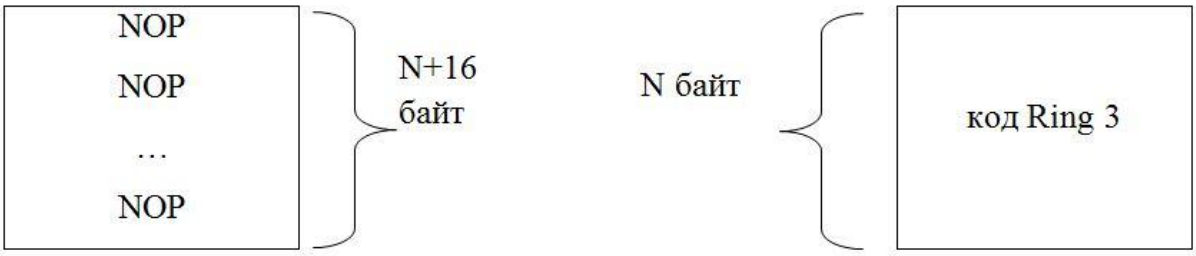

Рисунок 4 – Схема внедрения кода

Дополнительно вводится 16 резервных байт, для того чтобы мог функционировать алгоритм блочного симметричного шифрования AES-128, работающий с блоками по 16 байт[2,3].

Схема внедряемого кода ring 3 представлена на рисунке 5.

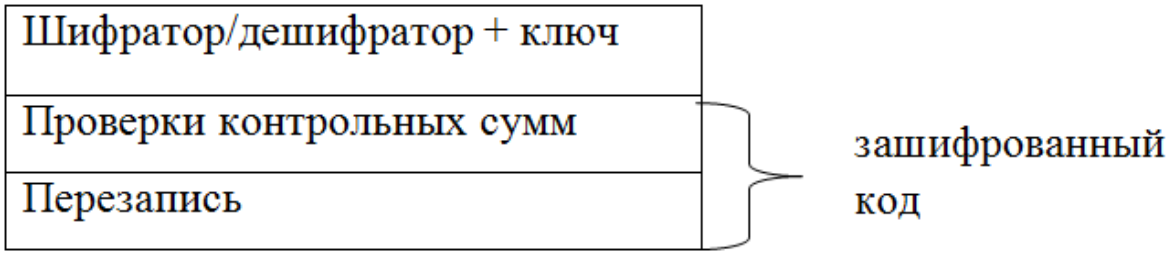

Рисунок 5 – Схема внедряемого кода Ring 3

При запуске кода Ring 3 сначала запускается дешифратор, чтобы расшифровать код проверки контрольных сумм и код, необходимый для записи Ring 3 в другой exe-файл.

Шифрование внедряемого кода необходимо только для того, чтобы антивирусное программное обеспечение не смогло обнаружить схожесть кода кольца Ring 3 с некоторыми вирусами. Именно по этой причине, в рамках поставленной задачи, проблема безопасного хранения ключа не является принципиальной[1].

Внедрение кода должно происходить по описанному ниже алгоритму.

1. Вычисляется псевдослучайный ключ.

2. Копируется код шифратора/дешифратора.

3. Копируется ключ.

4. Код зашифровывается по алгоритму симметричного шифрования AES-128.

5. Запускается файл, куда был скопирован код.

6. Осуществляется выход из процесса, который копировал код.

7. В файле, откуда был скопирован код, код Ring 3 заменяется операндами NOP.

Следует обозначить: чтобы не было отчётливо видно, что содержимое файлов 1.exe, 2.exe … M.exe постоянно меняется (это видно по времени изменения файла), при каждом копировании кода необходимо вызывать функцию SetFileTime, в которой устанавливать время изменения файла, равное времени его создания.

В связи с тем, что изменения файла по времени отследить невозможно, и размер файла не меняется, данный метод защиты незаметен для хакера и, следовательно, вероятность его взлома очень низка.

В заключение следует отметить, что внедрение разработанной методики поможет решить ряд перечисленных ниже задач.

1. Сведение к минимуму возможности нелегального использования программного обеспечения.

2. В связи с тем, что многие хакеры в crack-программы встраивают дополнительный вредоносный код, осуществляющий либо слежку за компьютером пользователей, либо создающий ботнет для DDOS атаки, рассылки спама и других видов незаконной деятельности, предложенная система защиты снизит число вредоносного программного обеспечения, распространяемого как с участием, так и без участия человека.

Литература

1. Карабейников А.Г. Математические основы криптографии. Учебное пособие - СПб: СПб ГИТМО (ТУ), 2002. - 41с.
2. Зензин О.С., Иванов М.А. Стандарт криптографической защиты AES. Конечные поля – М.: Кудиц-образ, 2002. - 173 с.
3. Joan Daemen, Vincent Rigmen AES Proposal: Rijndael – 47p.
4. Юров В.И. Assembler. Учебник для вузов - М.: Питер, 2010. - 636 с.
5. Крис Касперски, Ева Рокко. Искусство дизассемблирования – СПб: БХВ-Петербург, 2009. - 884 с.