Средство скрытой защиты программного обеспечения от несанкционированного использования

Сероб Балян
Государственный Инженерный Университет Армении,
Ереван, Армения

Ел-почта: serob.balyan@gmail.com

Роберт Акопян
к.т.н., доцент
Государственный Инженерный Университет Армении,
Ереван, Армения

Ел-почта: rob.hakobyan@gmail.com

Аннотация

Защищенное от несанкционированного использования программное обеспечение, вполне вероятно, подвергается атакам с целью взлома защиты. В данной работе рассматривается разработанная система, которая защищает прикладную программу от несанкционированного использования, связывая ее с конкретным компьютером, при этом скрывая сам факт существования защиты даже от самого пользователя. То есть программа должна работать только на том компьютере, на котором она установлена. Для ее запуска необходимо пройти скрытую аутентификацию.

Ключевые слова

уникальность компьютера, стеганография заголовочной части IP-пакета, система скрытой аутентификации

1. Введение

Основная идея организационных мер защиты заключается в том, что полноценное использование продукта невозможно без соответствующей поддержки со стороны производителя[1]. И, как известно, идеального способа защиты программного обеспечения не существует, в связи с этим, разработчики защитных систем не стремятся лишить потенциального взломщика самой возможности нейтрализации защиты, но стараются максимально усложнить этот процесс.

Защита может решать одну или комплекс задач, таких как защита от копирования, нелегального использования, модификации и др., но какая бы конечная цель ни стояла перед таким продуктом, разработчикам каждого из них прежде всего необходимо решить одну общую для всех проблему - качественной защиты от обследования программы на предмет обнаружения факта защищенности. С этой целью предлагается использовать методы стеганографии для сокрытия наличия защиты.

2. Описание системы защиты

Данная система состоит из двух программ: пользовательской программы (User.exe), которая должна быть защищена, и программы администратора (Admin.exe), с помощью которой должна быть осуществлена защита пользовательской программы.

Admin.exe запускается на том компьютере, на котором вдальнейшем будет работать User.exe. После ряда определенных шагов, Admin.exe создает уникальную последовательность символов и скрытно передает его на сервер, которая затем проверяется пользовательской программой, чтобы узнать находится ли она у законного пользователя, или нет. Если проверка даст положительный ответ, то программа продолжит свою работу, в противном случае она не будет делать ничего, даже не будет сообщения о нелегальном использовании программы (чтобы как можно дольше сохранить факт защищенности программы). Программа администратора должна быть доступна только человеку, обладающему соответствующими правами (администратор), хранится на передвижном (мобильном) запоминающем устройстве, и ни при каких обстоятельствах не оказаться на жестком диске компьютера пользователя.

Теперь рассмотрим более подробно функционирование вышеуказанных программ.

3. Программа администратора

Программа администратора читает серийные номера процессора и материнской платы (выбраны именно эти устройства, потому что, как правило, в отличие от других устройств, они реже заменяются новыми). Считываются серийные номера именно двух устройств, чтобы подчеркнуть уникальность компьютера: если по каким-то причинам не удалось читать одну из них, или такой серийный номер будет не единственным (поддельное устройство, или с помощью программного обеспечения измененный серийный номер), то, по крайней мере, второй из них обеспечит оригинальность. После считывания, эти данные суммируются и хешируются посредством алгоритма md5, чтобы серийные номера не встречались в системе в открытом виде. В дальнейшем сравниваются хеш значения, а не значения серийных номеров. Затем, полученное хеш значение скрывается в IP-пакеты и отправляется на хранение на сервер – для дальнейшего сравнения (рис.1).

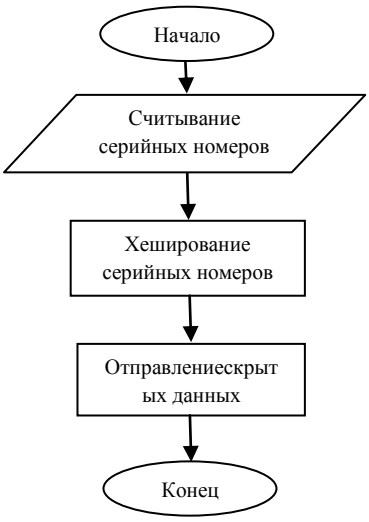

Рис. 1. Блок-схема работы программы администратора

3.1. Стеганография в заголовочной части IP-пакета

В описанном выше механизме используется стеганография в заголовке IP пакета. Выбран именно этот механизм, потому что оно имеет относительно высокую стеганографическую емкость, и в случае использования композитного сокрытия будет очень трудно обнаружить[2]. В заголовке IP есть несколько полей – пригодных для сокрытия в них информации. Первый – это Identification (ID) (рис. 2). Это поле длиною 16 бит и содержит

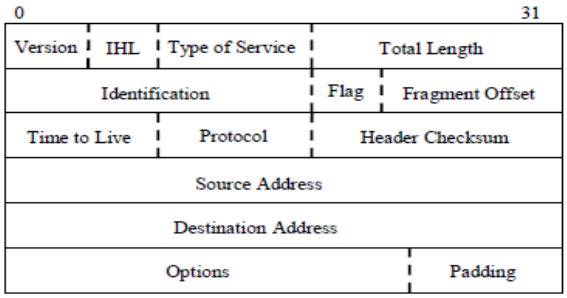

Рис. 2. Структура заголовочной части IP-пакета

уникальный идентификатор пакета, который используется для сборки фрагментированных датаграмм. Значение этого поля не зависит от значений других полей заголовка и сохраняется при фрагментации [3].

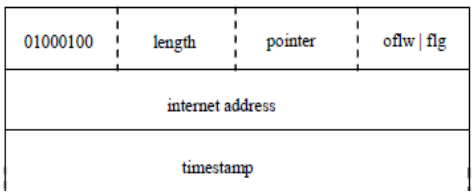

Рис. 3. Заголовок опции 'Timestamp'

Timestamp (32 бита), на рисунке 3 - временной штамп в миллисекундах (относительно полуночи по Единому Времени). Если время в миллисекундах неопределимо или не может быть отсчитано относительно полуночи по Единому Времени, то может быть внесено любое другое время при условии, что самый старший бит в поле временного штампа будет установлен в единицу (что указывает на использование нестандартного значения и дает некоторую свободу в заполнении данного поля) [3].

Перед сокрытием, необходимо учитывать следующие условия, чтобы не вызвать проблем в функциональности протокола [2]:

- Идентификатор должен быть уникальным в течение времени жизни (Time To Live) датаграммы. На практике это часто реализуется увеличением значения поля "ID" на единицу для каждой следующей датаграммы.
- Несмотря на возможность использования нестандартного временного штампа, для каждой следующей датаграммы значение этого поля тоже должно увеличиваться.
- В соответствии со спецификацией, отправитель должен создавать опцию "Timestamp" так, чтобы поля для временных штампов были достаточны для размещения всей ожидаемой информации. Размер опции не изменяется при добавлении временных штампов. Если поле с временными штампами уже заполнено, то датаграмма передается без вставки временного штампа, а счетчик переполнения увеличивается на единицу.

Таким образом, предлагается выполнить следующие шаги:

- Младшие 6 бит (0–5) 16-ти битного поля "ID" целесообразно использовать в соответствии с первым условием, объясненным выше. Остальные 10 бит используются для сокрытия пользовательских данных.
- Затем в IP-заголовоке необходимо создать опцию "Timestamp", которая содержит одно поле временного штампа. Опять, биты со второго до седьмого (1–6) используются для удовлетворения второго условия, первый бит устанавливается в "1", чтобы указать на использование нестандартного значения. В остальные 25 бит заносится остальная часть секретных данных.
- И последним шагом является создание действительной IP датаграммы, которая содержит данные - не вызывающие подозрения у противника.

В итоге, стеганографическая емкость каждого измененного IP-пакета (стего-контейнера) составляет 35 бит.

4. Пользовательская программа

Функции, выполняемые пользовательской программой, не имеют значения: обрабатывает сложные математические формулы или просто выводит на экран некое сообщение. Только необходимо поместить проверяющую подлинность функцию в определенных частях программы. То есть, либо пользовательская

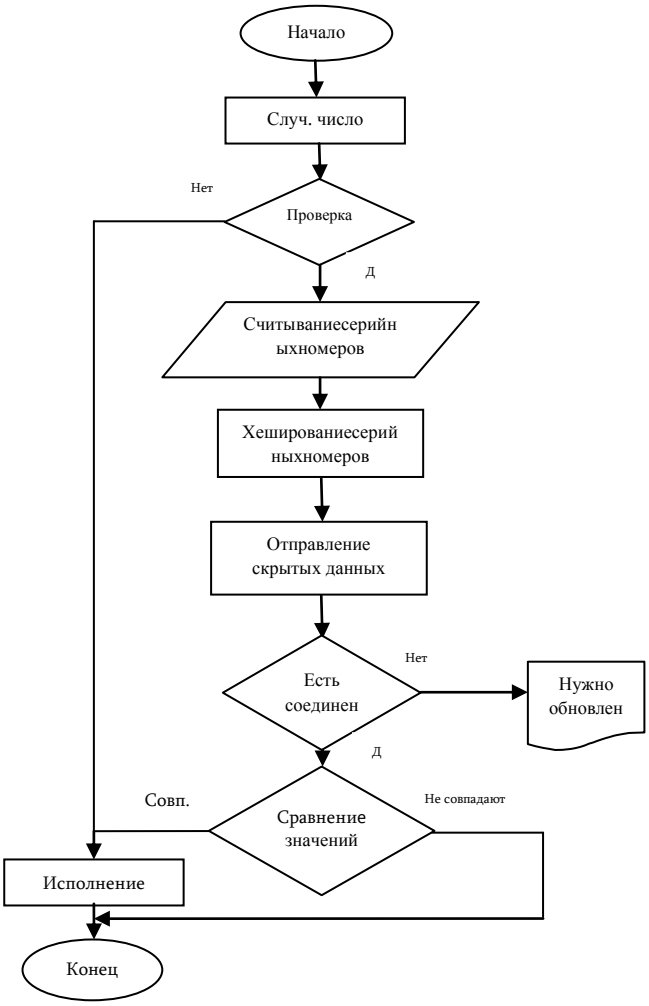

Рис.4. Блок-схема работы пользовательской программы

программа должна быть разработана нами, либо мы должны иметь доступ к ее исходному коду.

Допустим, что имеем программу, которая, при нажатии на кнопку, должна вывести на экран сообщение. Если она находится у полномочного пользователя, то должна работать как следует (вывести на экран сообщение), в противном случае не делать ничего. Проверяющая функция почти повторяет шаги администраторской программы. Но только вначале генерируется случайное число, в зависимости от которого программа проверяет или не проверяет подлинность данной копии программы – для случая не возможности подключения к интернету (рис. 4).

Если после генерирования оказалось, что программа должна пройти проверку, а соединение с интернетом отсутствует (т.е. нет ответа от сервера), то выдается сообщение о надобности соединения с интернетом для критических обновлений, без которых дальнейшее использование программы не возможно. Если есть соединение, то, как и администраторская программа, она читает серийные номера процессора и материнской платы, после чего хеширует их алгоритмом md5.

Затем хеш-значение скрытно отправляется на сервер, где и сравнивается с уже существующим (отправленным администраторской программой) хеш-значением. Если отправленное хеш-значение совпадает с хранящимся на сервере хеш-значением, то считается, что программа находится у законного пользователя, и тогда от сервера скрытно передается положительный ответ. В противном случае - ответ отрицательный. Следовательно, в зависимости от ответа, программа начинает функционирование (если результат положителен), или нет (в противном случае).

5. Возможные действия противника

Под термином "противник" подразумеваем ту личность, которая хочет не законными способами использовать программу и для этого предпринимает шаги.

Несмотря на то, что администраторская программа должна находиться только у администратора, тем не менее, для противника легче всего будет каким-то способом овладеть ею и запустить на своем компьютере. Для защиты от такой ситуации разработана система скрытой аутентификации. Предполагается, что для запуска администраторской программы необходимо нажать на кнопку в появляющемся на экране окне. Вся суть в следующем: для аутентификации в зависимости от текущего дня, данного часа и минуты, пользователь (в данном случае администратор) должен с определенной последовательностью с помощью «мышки» отметить некоторые углы окна, после чего, только, разрешается выполнение основной части администраторской программы (если соответствующие углы не нажаты, то программа выдает ошибку про отсутствие какого-то файла, с целью как можно долго продлить факт существования такой защиты). Теперь рассмотрим, как организована эта защита. Углы основного окна программы Admin.exe, начиная с левого верхнего угла, мысленно пронумерованы по часовой стрелке, начиная от 0-я до 3-х (рис. 5).

Рис.5. Основное окно Admin.exe и пронумерованные углы

Используем очередность составляющих текущего момента: день месяца, день недели, час, минута. Пользователь каждое полученное из компьютера значение (именно с этой очередностью) должен разделить по модулью 4 и нажать совпадающий с полученным результатом угол. Например - допустим, что сейчас 14-ое октября, пятница (4-ый день недели), время 16:31, тогда, разделив эти значения по модулью 4, получим:

14 mod 4 = 2
4 mod 4 = 0
16 mod 4 = 0

31 mod 4 = 3

То есть, углы должны быть отмечены с очередностью 2-0-0-3. После отметки углов (в правильном порядке), нажатием кнопки "Push" выполнится основная часть администраторской программы (будут считаны серийные номера, хешированы, зашифрованы, сокрыты). Зависимость от переменных значений обеспечивает исключение того случая, когда у противника есть возможность проследить за работой администратора: если противник заметит, что администратор до нажатия кнопки отмечает некоторые углы, то в дальнейшем, повторяя шаги администратора, более вероятно, что его отметки не совпадут с текущей минутой.

Если противник подозревает, что программа защищена таким образом, то он может декомпилировать программу с целью изучения ее работы. Поскольку Admin.exe написан на языке C#, который является языком платформы .NET и компилирует исходный код в промежуточный код (байт-код), который содержит достаточно информации для адекватного восстановления исходного кода, то ее декомпиляция достаточно проста[4]. Поэтому промежуточный код программы подвергаем обфускации. Обфусцировалась и пользовательская программа. Применялись две разные обфускаторы: один из них, кроме переименования классов, шифрует и строковые данные. Благодаря этому, для противника останутся секретными имена разных файлов, сообщения, которые выводятся на экран и другие подобные данные, которые могли бы стать причиной каких-либо предположений. Поэтому в некоторых частях программы употреблялись строковые типы, несмотря на то, что не было нужды в их использовании. У второго обфускатора есть возможность, которая просто не дает декомпилирующей программе открыть методы класса.

Если противник не смог овладеть администраторской программой, он будет вынужден строить все свои предположения на основе пользовательской программы (несмотря на то, что она обфусцирована). Какими могут быть действия противника, если он каким-то образом узнал о секретных данных системы защиты? В таком случае, он может повторить действия администраторской программы, и тогда система будет считаться взломанным.

6. Заключение

С помощью разработанной системы можно защитить программу от несанкционированного использования. Но система не идеальная: недостаток системы состоит в том, что при замене процессора и/или материнской платы компьютера надо будет запускать программу администратора заново.

В дальнейшем планируется создать механизм, который бы решил проблему замены устройств, т.е. секретно сохранить уникальные данные в компютере.

Список литературы

[1] Веб журнал “Pcweek”, “Рынок средств защиты ПО от копирования: анализ и рекомендации”, http://www.pcweek.ru/themes/detail.php?ID=66709,

[2] V. Danielyan, “Software protection based on IP steganography”, Proceedings of the conference Computer Science And Information Technologies (CSIT-2011), p. 359-361, Yerevan, Armenia.

[3] “Request for Comments 791, Internet Protocol”, http://www.faqs.org/rfcs/rfc791.htm

[4] Веб форум “Cit”, “Чернов А. В. - Анализ запутывающих преобразований программ”, http://www.citforum.ru/security/articles/analysis/