Криптографическая защита информации персонального компьютера при работе в сети Internet: принципы разработки собственных программ

С. И. Почуев, В. П. Большаков, З. Ф. Остапенко

Введение

«Как защитить конфиденциальные данные, хранящиеся на несъемных носителях информации персонального компьютера от посторонних глаз?» Этот вопрос все чаще задают себе «продвинутые» пользователи сети INTERNET, отчетливо понимая, что любые способы защиты, кроме исключения непосредственного физического контакта компьютера с сетью, не дают 100 % гарантии…

В настоящее время гражданам РФ доступны для свободного приобретения и применения компьютерные программы защиты информации, не имеющие официального государственного статуса. Ввиду многочисленности последних, их легко можно «скачать» в сети, купить на CD, позаимствовать у друзей, самостоятельно разработать и т. п. Ряд программ защиты информации штатно входят в состав средств операционной системы (ОС) персонального компьютера [1, 2].

Вместе с тем известно, что любая «чужая» и тем более не сертифицированная «публичная» программа такого класса, применяемая Вами в виде исполняемого модуля, является «черным ящиком». Это может таить в себе неприятные неожиданности, несмотря на возможно широкую рекламу и искренние заверения производителей, в том числе зарубежных, в ее эффективности и безопасности… Поэтому у людей, знакомых с основами программирования, возникает искушение написать собственную программу для защиты информации — простую, надежную, прозрачную для себя и недоступную для посторонних.

Некоторым принципам создания подобных программ посвящена настоящая статья.

Принципы создания индивидуальных программ криптографической защиты

Очевидно, что одним из наиболее действенных способов защиты информации является ее хранение в зашифрованном виде. Допускаем, что читатель, в отличие от авторов данной статьи, является специалистом в области криптографии, поэтому все последующие рассуждения могут показаться ему наивными или давно и хорошо известными. Однако большинство людей, использующих INTERNET (в том числе и хакеров), все-таки не являются профессионалами в области криптографической защиты, поэтому нижеизложенные принципы разработки личных криптографических программ зашиты информации, как нам кажется, позволяют самостоятельно создавать достаточно эффективные (по крайней мере, против любопытствующих сетевых дилетантов) программные продукты.

По нашему мнению, эффективная криптографическая защита информации может осуществляться штатными средствами операционной системы Windows любых версий, установленной на большинстве персональных компьютеров. Такими средствами являются встроенные в операционную систему датчики псевдослучайных (RND) чисел [1,3].

Рассмотрим принципы криптографической защиты информации с использованием RND датчика на примере файла часто используемого *. rtf формата (международная ASCII кодировка алфавита из 256 символов, имеющих собственные уникальные номера):

1 Предлагаются два базовых подхода (для конкретного алгоритма и в зависимости от фантазии и опыта разработчика-программиста возможны любые их ad hoc модификации и комбинации) к криптографической защите информации:

– псевдослучайное (с помощью RND датчика) изменение значений символов исходного файла на фиксированных позициях;
– псевдослучайное (с помощью RND датчика) изменение позиций символов исходного файла без изменения их значений.

В первом случае, например, исходный символ «**X**», находящийся на **i**-той позиции защищаемого файла, становится символом «**Y**» (или любым другим из 256 возможных, за счет псевдослучайного приращения значения ASCII кода) на той же позиции.

Во втором случае, меняются местами символы «**X**» и «**Y**», находящиеся соответственно на **i**-той и на **j**-той псевдослучайных позициях в пределах защищаемого файла.

Как было отмечено ранее, при разработке конкретного алгоритма оба подхода или любые их разновидности могут использоваться в различной последовательности и, в принципе, для повышения стойкости криптографической зашиты информации неограниченное число раз.

2 Ключом для раскрытия зашифрованной информации в рассматриваемом случае

являются знания начальных значений использованных псевдослучайных последовательностей, автоматически сгенерированных ОС Windows и изменивших значения и месторасположение символов исходного файла. Тогда для расшифровки файла достаточно восстановить результирующую последовательность псевдослучайных чисел (совокупность нескольких детерминированных последовательностей с псевдослучайными начальными значениями) и просто повторить ее в обратном порядке.

Начальные значения для каждой из использованных последовательностей могут формироваться на основе задания и запоминания пользователем ключевого слова-пароля. При этом пароль в компьютере, в программе криптографической защиты и в создаваемых ей файлах не хранится!

Например, задав (и запомнив) с клавиатуры компьютера пароль из 6-ти символов, мы получаем 18-ти значное или 6-ти значное целое число в строковом представлении. При этом каждому символу пароля соответствует в ASCII кодировке 3-х значное или однозначное целое число соответственно. Затем на основе данного целого числа формируется начальное значение (первое число) псевдослучайной RND последовательности. Последующие числа данной последовательности автоматически генерируются ОС Windows по некоторому детерминированному алгоритму (не важно, по какому именно) на основе предыдущего значения.

Предварительная оценка криптостойкости

Как следует из вышеизложенного, для «лобового» (путем перебора всех возможных вариантов) вскрытия пароля, при наличии у взломщика созданной Вами программы криптографической защиты, ему необходимо перебрать **P = (L ^ N) ^ M** комбинаций.

Здесь: **L** — объем алфавита клавиатуры компьютера;
N — длина пароля;
M — число псевдослучайных последовательностей, использованных при шифровании;
^ — знак возведения в степень.

Даже в простейшем случае, когда **L**=256 (используется стандартный набор символов клавиатуры компьютера), **N**=6 (задан пароль из 6 символов для определения начального значения каждой из псевдослучайных последовательностей), **M**=2 (для шифрования использованы лишь две псевдослучайные последовательности) число возможных **P** комбинаций составляет ~ 10 ^ 29. Если предположить, что для «лобового» взлома зашифрованного соответствующей программой файла исполь-зуется не существующий в природе суперкомпьютер, позволяющий провести обработку одной комбинации (дешифрование файла средних {~100 Кб} размеров и проверку результата за 10 ^ –12 сек.), то для перебора всех возможных комбинаций потребуется ~ 1 миллиард лет!

Предположим теперь, что злоумышленнику полностью известен алгоритм работы Вашей программы криптографической защиты. Он точно знает (хотя не тут-то было, ведь программа то эксклюзивная — лично Ваша и хранится, например, на съемном носителе информации в виде *.exe файла) число, последовательность, авторские особенности и уникальные модификации описанных способов криптографической защиты. Поэтому хакер предпринимает более хитроумную попытку — не «уродуясь» с лобовым взломом пароля непосредственно подобрать последовательность необходимых ключевых чисел в течение приемлемого времени, используя свойство псевдослучайности начальных значений RND последовательностей.

Поясним данное свойство. Дело в том, что количество генерируемых ОС Windows псевдослучайных чисел, используемых для задания начальных значений последовательностей на основе введенных паролей хотя и огромно, но конечно. Это теоретически является «слабым звеном» в предложенной цепи криптографической защиты информации, т. к. число проверяемых комбинаций уменьшается и составляет **P = R ^ M**.

Здесь: **R** — неповторяющиеся псевдослучайные числа, генерируемые операционной системой для задания начальных значений псевдослучайных RND последовательностей;
M — число используемых в алгоритме последовательностей.

Однако если предположить, что число **R**, например, при использовании 6–ти символьного пароля превышает 10 ^ 6 (в этом, при желании, каждому нетрудно убедиться, протестировав встроенный в Windows датчик RND чисел методом статистических испытаний), то все равно необходимо проверить, как минимум, **P = (10 ^ 6) ^ M** возможных комбинаций. Иными словами, нужно одновременно подобрать последовательность из **M** конкретных псевдослучайных чисел, выбранных из конечного множества RND чисел, генерируемого ОС Windows при задании начальных значений на основе введенных паролей.

Для вышерассмотренного примера (при **M**=2) необходимо будет проверить как минимум 10 ^ 12 возможных комбинаций. Таким образом, для суперкомпьютера с быстродействием 10 ^ 10 операций в секунду (если такой имеется в наличии у взломщика) при размере взламываемого файла ~100 Кб (пусть время выполнения дешифрования для одной комбинации ~ 0.0001 сек.), потребное машинное время гарантировано превысит 300 лет! Вновь напомню, что применение данного способа взлома потребует детального знания злоумышленником авторского алгоритма криптографической защиты, что на практике маловероятно.

Особенности защиты e-mail

Очевидным приложением рассмотренного подхода к криптографической защите является его применение для обеспечения конфиденциальности личной электронной почты. Данная задача, безусловно, является более сложной, чем просто защита файлов собственного винчестера от несанкционированного сетевого просмотра. Детальное ее рассмотрение выходит за рамки настоящей статьи. Поэтому в иллюстративных целях ограничимся лишь описанием очевидных возможных схем защиты электронной корреспонденции.

В тривиальном случае Ваш корреспондент может просто использовать созданную Вами программу криптографической защиты, имея соответствующий *.exe модуль. При этом Вы должны изначально назначить общий пароль. Для личной переписки это вполне приемлемый вариант защиты электронной почты!

В более сложном случае может быть использована разновидность схемы криптографической защиты с открытым ключом. Реализация данной схемы потребует от Вас создания и передачи Вашему корреспонденту программы, которая будет автоматически шифровать вводимый им пароль по известному только Вам RND алгоритму и хитро помещать пароль в защищаемый файл электронной почты. Тогда, получив от корреспондента по E-MAIL зашифрованный файл, Вы сначала дешифруете пароль, использованный корреспондентом. Затем, расшифровав пароль, вы дешифруете само информационное сообщение.

Недостатком данной схемы является теоретическая возможность вскрытия и использования хакером Вашего алгоритма криптографической защиты пароля. Правда для реализации данной возможности злоумышленник будет вынужден: похитить *. exe файл с программой у Вашего корреспондента, восстановить исходный текст Вашей программы по исполняемому модулю (а это не просто), понять использованный Вами алгоритм криптографической защиты (а это тоже не просто) и написать соответствующие программы дешифрования.

Заключение

В силу простоты и прозрачности изложенных принципов криптографической защиты информации для защиты персонального компьютера от доступа к конфиденциальным файлам, написание собственной программы криптографической защиты является чисто технической процедурой и не представляет большой сложности для программирующего пользователя. Поэтому пример исходного текста подобной программы в настоящем очерке не приводится. (За пару дней программист средней квалификации, используя изложенный подход и быструю среду проектирования типа DELPHI, VISUAL BASIC и т. п., в состоянии самостоятельно создать собственную уникальную и вполне приличную по эффективности версию «шифровалки») Кроме того, любые конкретные алгоритмы и создаваемые на их основе программные продукты все же считаются авторским ноухау и могут представлять самостоятельную ценность…

Кстати, если Вы не поленитесь написать еще несколько вспомогательных строк программы криптографической защиты, придающих дополнительный блеск Вашему «программному произведению» и изменяющих атрибуты создаваемого зашифрованного файла (например, делая его графическим или звуковым), то будущий взломщик Ваших «секретных» файлов будет «приятно» удивлен! При попытке открытия данных файлов с помощью обычных программ-редакторов (графических или звуковых) вместо адресной книги, личной переписки или номера кредитной карты с текущим балансом он услышит из динамиков компьютера Ваше «злобное шипение» или увидит на экране монитора произведение «абстрактной живописи»…

Что же касается эффективности предложенного подхода к созданию криптографических программ защиты информации для домашнего применения с использованием «подручных» средств ОС Windows, то авторы статьи пребывают в состоянии понятной эйфории преуспевших дилетантов (тестовые программы написаны, надежно работают с неплохим быстродействием, а зашифрованные файлы не взламываются известными средствами).

Справедливости ради необходимо отметить, что, скорее всего, существуют и более эффективные, чем рассмотренные выше, способы взлома файлов, зашифрованных с помощью предложенных принципов криптографической защиты информации. О подобных способах дешифрования, основанных на специализированных криптографических методах анализа, имеется достаточно много «туманных» и, по понятным причинам не раскрывающих сути дешифрования, статей в INTERNET. Поэтому приведенные оценки стойкости криптозащиты могут оказаться, мягко говоря, оптимистичными (конечно, если за дело дешифрирования возьмется профессионал), а предложенный подход к криптографической защите и создаваемые на его основе программы, по-видимому, требуют дополнительной оценки и тестирования специалистами.

Литература

1. Осмаловский С. А. Статистические методы защиты информации. - М.: Радио и связь, 2003.
2. Шаныгин В. Ф. Информационная безопасность компьютерных систем и сетей: Учебное пособие. - М.: «ФОРУМ»: ИНФРА - М, 2007.
3. Иванов М. А., ЧУГУНКОВ И. В. Теория, применение и оценка качества генераторов псевдослучайных последовательностей. - М.: КУДИЦ - ОБРАЗ, 2003.