УДК 004.056

Н.И. Алишов, В.А. Марченко, А.Н. Мищенко

ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС ЗАЩИТЫ НА ОСНОВЕ НЕРАСКРЫВАЕМЫХ ШИФРОВ

Рассматриваются вопросы создания программно-аппаратного комплекса защиты информации. Приводится анализ особенностей создания аппаратного устройства на базе цифровых процессоров обработки сигналов для реализации не раскрываемых алгоритмов шифрования. Предложены алгоритмы согласования необходимых параметров связи между двумя удаленными устройствами. Описан программно-аппаратный комплекс защиты для систем потокового вещания.

Введение

В 1917 г. сотрудником фирмы AT&T Гильбертом Вернамом изобретена система симметричного шифрования, получившая впоследствии название в честь создателя – «шифр Вернама» [1]. В криптографии разновидность шифра Вернама известна под названием «схема одноразовых блокнотов» (one-time pad).

Исследуя шифр Вернама, Шеннон определил [2], что ключ имеет длину, равную длине самого передаваемого сообщения. Ключ используется в качестве гаммы, и если каждый бит ключа выбирается случайно, то вскрыть шифр невозможно (так как все открытые тексты будут равновероятны).

Несмотря на доказательство Шеннона, на практике у шифра Вернама есть серьезные недостатки, ограничивающие его повсеместное применение, главным из них является то, что ключ должен:

- быть истинно случайным;
- совпадать по размеру с заданным открытым текстом;
- применяться только один раз.

На данный момент одноразовые блокноты (one-time pad, OTP) являются единственной теоретически не взламываемой системой шифрования [3], которая представляет собой список чисел в случайном порядке, используемый для кодирования сообщения. Ключ OTP может применяться только один раз

Состояние проблемы

Недостатком одноразовых блокнотов является необходимость генерации действительно случайных блокнотов (последовательностей) и проблема их распространения. Другими словами, если блокнот выявляется, то раскрывается и та информация, которую он защищает. Если блокноты не случайные, возможно выявление схем, которые можно использовать для анализа частоты встречаемых символов, а также другие атаки.

Дальнейшим развитием идей, заложенных в одноразовых блокнотах, является «Метод косвенного шифрования» [4], основная идея которого заключается в том, что полезная для перехвата информация вообще не передается по каналу, передаётся только образ этой информации (рис. 1). Выглядит это следующим образом – у отправителя и получателя имеются одинаковые секретные файлы (выступающие в качестве одноразового блокнота, далее контейнер-ключ). Поток информации, подлежащий защите, условно делится на сегменты, после чего производится замена (по определенному алгоритму) сегментов полезной информации сегментами секретного контейнера-ключа. В результате получается образ исходной информации, как правило, такого же размера. Сегменты образа полезной информации можно в реальном времени передавать по каналу связи. При получении адресатом образ подвергается обратному преобразованию – его сегменты заменяются сегментами секретного контейнера-ключа, т. е. выполняется зеркальный алгоритм.

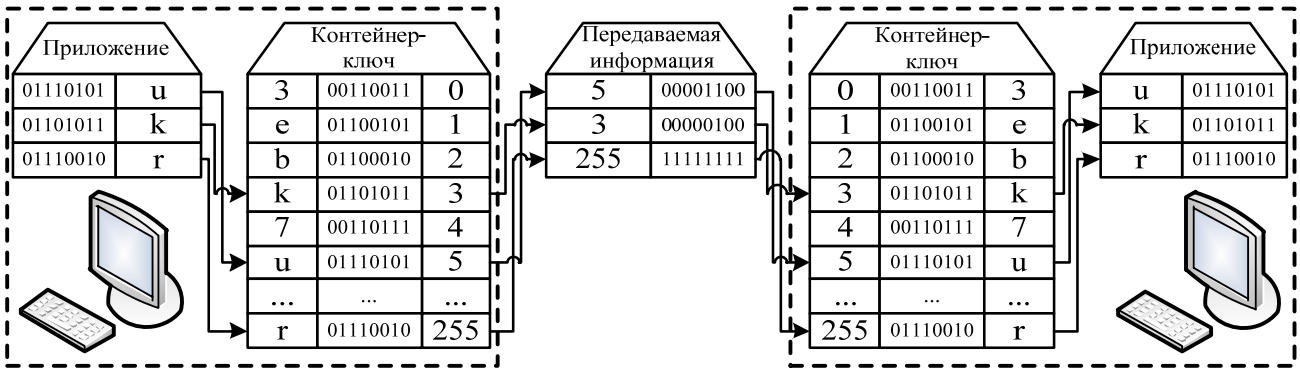

Рис. 1. Общий алгоритм косвенного шифрования

Аппаратное решение на базе стандартного микропроцессора

Программные средства обеспечения информационной безопасности являются потенциально уязвимыми, поскольку весь процесс кодирования (шифрования/дешифрования) данных выполняется во внутренней памяти ЭВМ, к которой может получить доступ любое запущенное на ЭВМ приложение. А это означает, что существует возможность проводить разноуровневые атаки на любое ПО, в том числе и на направленное обеспечивать безопасность обрабатываемой информации. Таким образом, построить высокоуровневую защиту исключительно программными средствами практически невозможно [5].

Для ограничения доступа к средствам, выполняющим криптографические преобразования, необходимо перенести их из ЭВМ на закрытую аппаратную подсистему. В результате чего злоумышленник не сможет получить непосредственный доступ к процессам кодирования (шифрования/дешифрования).

Авторами был разработан прототип программно-аппаратного комплекса защиты [6], состоящего из двух подсистем:

- программной – обеспечивает гибкую интеграцию защитного комплекса в информационную систему;
- аппаратной – взаимодействует с программной подсистемой, выполняет кодирование (шифрование/дешифрование) полезной информации.

Внутри аппаратного устройства реализован алгоритм косвенного шифрования, который принадлежит к классу нераскрываемых шифров. Разработанный комплекс обеспечивает повышенный уровень защиты за счёт того, что используемый контейнер-ключ содержится исключительно внутри аппаратного устройства и никогда не покидает его пределы (рис. 2).

Главной особенностью данного устройства является использование стандартных общедоступных компонентов, что позволяет применять комплекс защиты в различных отраслях за счет его простой модификации под нужные задачи. Основной частью аппаратного устройства является «Блок ПЗУ», в котором размещается используемый ключ-контейнер. Этот блок представляет собой флэш-память большого объема. Следует отметить, что современные микросхемы флэш-памяти обладают объемом порядка сотни байт, при этом имея миниатюрный размер. Для создания комплекса защиты на базе указанного прототипа необходимо выполнить инициализацию устройства защиты. Для этого с помощью специализированного генератора истинно случайных чисел генерируется массив чисел, объем которого равен объему части блока ПЗУ, необходимый для хранения ключа-контейнера.

Сгенерированный файл записывается в одно или несколько устройств и при необходимости сохраняется во внешнем файле. При выполнении операций кодирования используется описанный ключ-контейнер, находящийся в ПЗУ. После использования всего ключа-контейнера необходимо повторно инициализировать устройство новым ключом-контейнером.

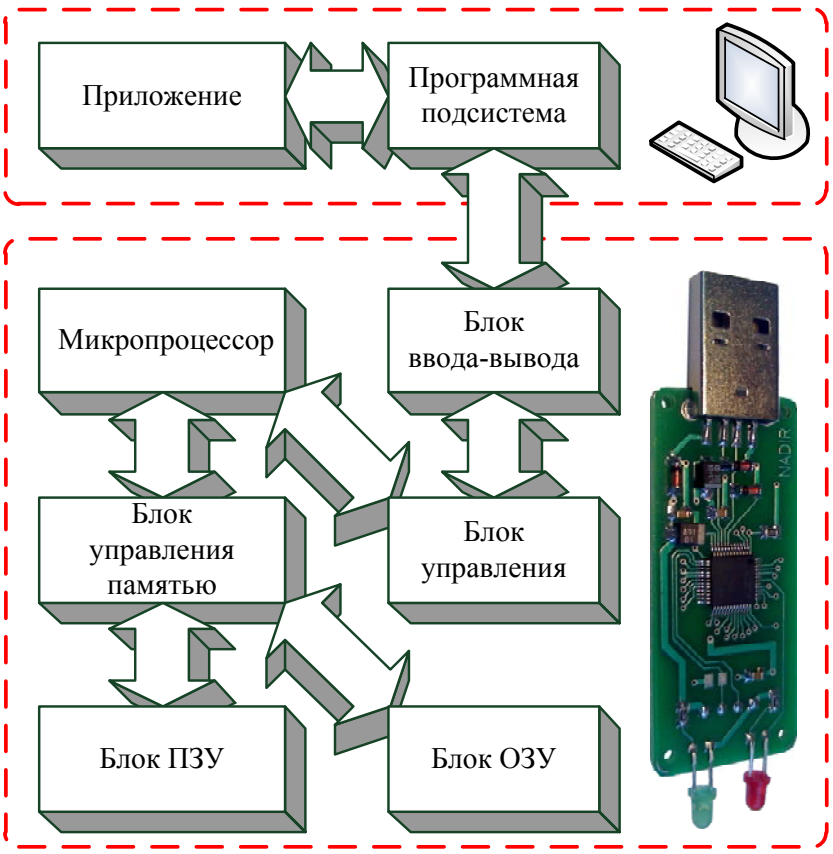

Рис. 2. Принципиальная схема существующего решения

На базе построенного прототипа было разработано ряд приложений, обеспечивающих защиту информации в типовых задачах:

- защита веб-трафика (http/https);
- построение систем защищенного документооборота;
- контроль доступа к информационным ресурсам;
- контроль доступа к охраняемым объектам;
- защита почтовых сообщений.

В дополнение к типовым задачам защиты благодаря наличию готовых программных решений авторам представляется возможным использование разработанного комплекса защиты в любых задачах, связанных с обеспечением повышенной криптографической стойкости различных массивов данных.

Разработка и построение аппаратного устройства защиты на базе процессоров цифровой обработки сигналов (ПЦОС)

Комплекс защиты на базе стандартного микропроцессора не подходит для применения в задачах криптографической защиты сверхбольших массивов данных, организации шифрования «на лету» потоковой информации (речь, аудит, видео) и других подобных задачах, ввиду использования флэш-памяти ограниченного объема и ограниченной производительности классических микроконтроллеров. Эти трудности могут быть решены путем использования дополнительных модулей флэш-памяти или более вместительных накопителей, что естественно приведёт к значительному увеличению накладных расходов.

Поэтому была разработана новая модификация аппаратного устройства комплекса защиты с применением ПЦОС и новой элементной базы.

Новое устройство предполагает наличие нескольких отличительных возможностей:

- шифрование/дешифрование неограниченного объема данных;
- выполнение криптографических преобразований в реальном масштабе времени;
- гибкий выбор скорости шифрования, криптостойкости зашифрованных данных;
- работа в потоковом режиме шифрования.

Принципиальная схема создаваемого устройства показана на рис. 3. Это устройство имеет схожую архитектуру, что и первый прототип, но здесь дополнительно организован ПЦОС с соответствующими схемами развязки и коммутации.

Для обеспечения возможности шифрования неограниченного объема данных используемый ключ-контейнер генерируется внутри устройства и потом сохраняется в ПЗУ устройства. В качестве генераторов случайных чисел, необходимых для создания ключа-контейнера, предполагается использование криптостойких алгоритмов генерации псевдослучайных чисел (ГПСЧ). Эти алгоритмы позволяют, во-первых, генерировать поток чисел, неотличимых от случайных чисел, и, во-вторых, использовать комбинацию алгоритма и зерно инициализации ГПСЧ как ключ шифрования. Этим условиям отвечают ГПСЧ, которые проходят все тесты, описанные в стандарте FIPS 140-3 [7].

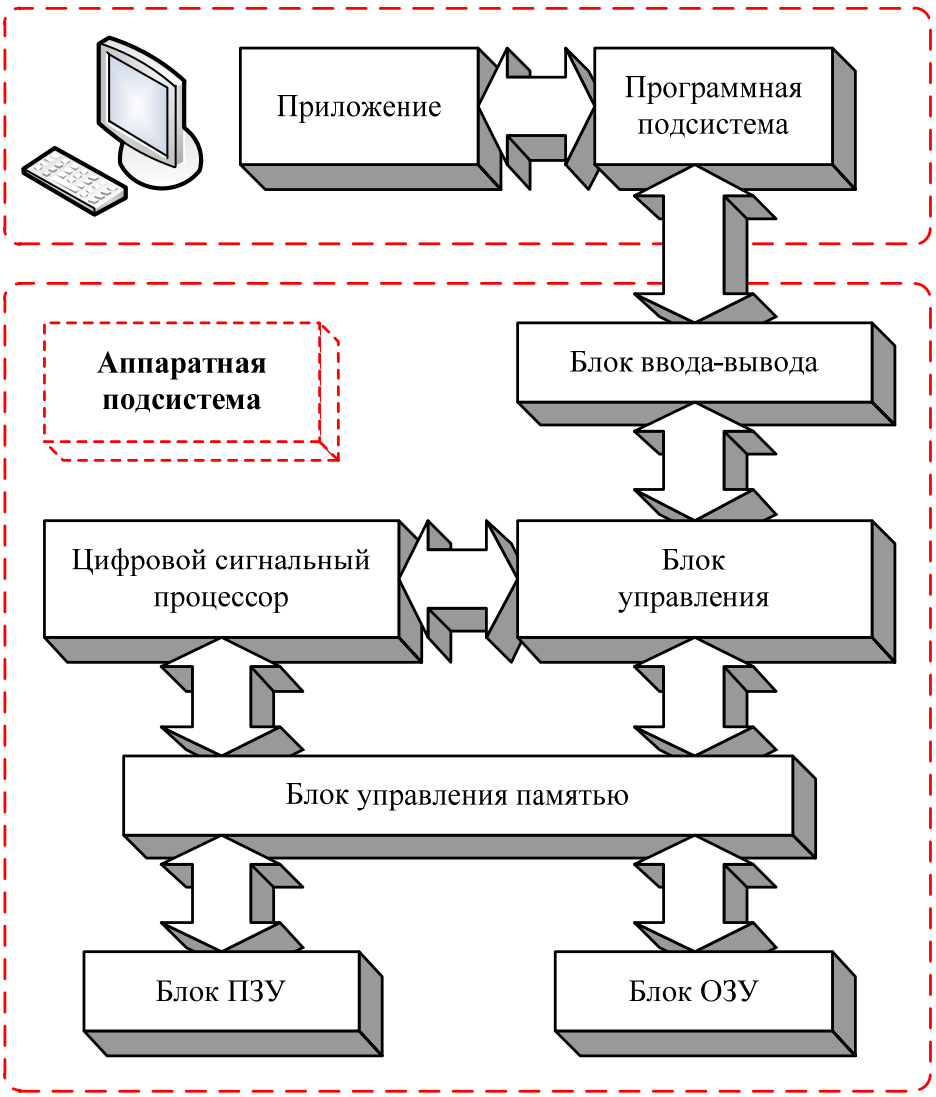

Рис. 3. Принципиальная схема устройства на базе ПЦОС

Алгоритмы псевдослучайной генерации чисел. Для создания контейнера-ключа предлагается использовать несколько генераторов псевдослучайных чисел, при этом преследуются такие цели:

- возможность динамического выбора генератора псевдослучайных чисел в зависимости от внешних условий;
- усложнение попыток подбора генерируемых значений путем случайного выбора генератора.

Основной сферой применения разрабатываемого устройства предполагается защита пересылаемой информации в распределенные телекоммуникационные сети, которые обладают неравномерными свойствами с точки зрения производительности, задержек и т. п. Исходя из таких особенностей современных сетей, используемые распределенные приложения могут динамически изменять различные параметры соединения для подстройки под доступные возможности системы и каналов связи. Соответственно и требования к системе защиты могут динамически изменяться. Например, при использовании системы удаленного доступа в сети государственного органа власти, которая задействует промежуточные недоверенные сети, следует выбрать более криптостойкий алгоритм ГПСЧ, но с меньшей производительностью. Другие доступные алгоритмы с большей производительностью могут использоваться для шифрования мультимедийного контента в реальном времени между различными подсетями в корпоративной сети.

Для усложнения взлома канала передачи предлагается выбирать используемый ГПСЧ в момент инициализации процесса шифрования передаваемой информации и регулярно его менять в процессе шифрования. Это позволит получить следующие преимущества:

- заранее неизвестно, какой будет использоваться алгоритм ГПСЧ, и соответственно невозможно предсказать, какой набор атак нужно применить;
- взлом одного из предыдущих сеансов не позволяет, каким-либо образом получить дополнительную информацию для компрометации последующих сеансов (предсказания «зерна» инициализации ГПСЧ, раскрытие ранее сгенерированных последовательностей и т. п.).

В качестве указанных алгоритмов ГПСЧ используются алгоритмы, которые легко реализуются на основе ПЦОС. К таким алгоритмам относятся генератор Blum-Micali, Blum Blum Shub, ANSI X9.31 и многие другие математические алгоритмы [8, 9]. В некоторых современных ПЦОС могут использоваться ГПСЧ в виде отдельных аппаратных реализаций, обладающие значительной производительностью [10].

Для функционирования устройства предполагается реализация нескольких этапов:

- инициализация устройства;
- инициализация сеанса связи;
- шифрование передаваемых данных.

Инициализация устройства. Процесс инициализации устройства запускается при поступлении команды извне о том, что будет проводиться шифрование сеанса связи. На этом этапе выбирается схема проведения защиты канала из следующего набора доступных вариантов.

1. Случайный выбор ГПСЧ из набора доступных реализаций.
2. Выбор используемого ГПСЧ исходя из требуемых параметров связи.
3. Использование заданного ГПСЧ.

После выбора генератора используемого для шифрования сеанса связи, производится генерация начального зерна инициализации этого ГПСЧ в рабочую область ПЦОС загружается соответствующая программа, реализующая выбранный ГПСЧ (в случае программной реализации), и запускается на выполнение. Выходящие псевдослучайные последовательности используются для формирования ключа-контейнера. Полученное начальное зерно инициализации генератора и номер используемого алгоритма ГПСЧ представляют собой ключ шифрования, который нужно передать другому участнику защищенного сеанса связи.

Инициализация сеанса связи. После генерации ключа выполняется его согласование с устройством, принимающим участие в сеансе связи. Для успешного

создания зашифрованного канала связи необходимо согласовать следующий набор параметров:

- используемый ГПСЧ;
- стартовое зерно генерации;
- используемый режим шифрования.

Для согласования указанных параметров предполагается использовать специальный протокол согласования на базе алгоритма «Диффа – Хеллмана» [11]. Авторами выбран этот алгоритм, так как он хорошо зарекомендовал себя в подобных задачах и достаточно глубоко исследован на наличие различных слабостей и недостатков. Разработанная схема позволяет согласовать ключ шифрования между двумя оконечными устройствами, используя незащищенный от прослушивания канал связи (рис. 4).

Весь алгоритм согласования реализуется внутри аппаратной части комплекса защиты, поэтому с программной составляющей никакие параметры, относящиеся к алгоритму шифрования, не отправляются, кроме запроса на инициализацию сеанса связи, содержащего указание адресата и требований, необходимых для устанавливаемого сеанса связи (тип трафика, допустимые задержки, необходимая скорость шифрования и т. п.). Следует отметить, что после инициализации устройства в его памяти уже хранятся выбранные начальные параметры (выбранный ГПСЧ, параметры связи и т. п.), необходимые для создания канала связи.

Дальше в работе рассматривается режим функционирования, предусматривающий использование заданного ГПСЧ между клиентским и удаленным устройством. Зачастую такой режим функционирования применяется, если необходимо обеспечить соответствие системы защиты информации регламентирующим требованиям государственных органов власти. В таком случае необходимо использовать только стандартизированные ГПСЧ (например, ANSI X9.31) и поэтому скрывать тип используемого ГПСЧ от злоумышленника нецелесообразно, так как эта информация априори доступна любому пользователю системы защиты, но в любом случае в открытом виде она не передается.

После поступления запроса на установление сеанса связи с помощью внутреннего генератора случайных чисел, реализованного в виде аппаратной схемы, или программно-аппаратной реализации генерируются два больших простых числа n и q и проводится их проверка на соответствие требованию

$$1<<q<n.$$

Далее с помощью того же генератора случайных чисел генерируется большое случайное число x, а также число

$$A = q^x \ mod \ n.$$

Полученные значения n, q, A используются при формировании пакета для передачи на удаленное устройство. К полученному пакету добавляются дополнительные параметры (выбранный режим функционирования системы защиты, конкретный ГПСЧ и т. п.), после чего он пересылается на удаленное устройство.

После получения удаленным устройством пакета с запросом на установление сеанса связи из его содержимого извлекаются запрашиваемые параметры связи и проверяется их выполнимость на данном устройстве (реализован ли алгоритм ГПСЧ, доступность необходимого режима функционирования и т. п.). Если эти параметры подходят, то инициализация сеанса продолжается, в противном случае отправляется уведомление на клиентское устройство, что параметры связи не подходят и сеанс связи прекращается.

Если процесс установления сеанса связи между двумя устройствами продолжается, то из полученного пакета извлекаются значения A, q, n. После чего генератор случайных чисел генерирует случайное большое число y и вычисляет новое значение

$$B = q^y \ mod \ n.$$

В соответствии с ранее сгенерированным числом y вычисляется значение K_y по формуле

$$K_y = A^y \ mod \ n.$$

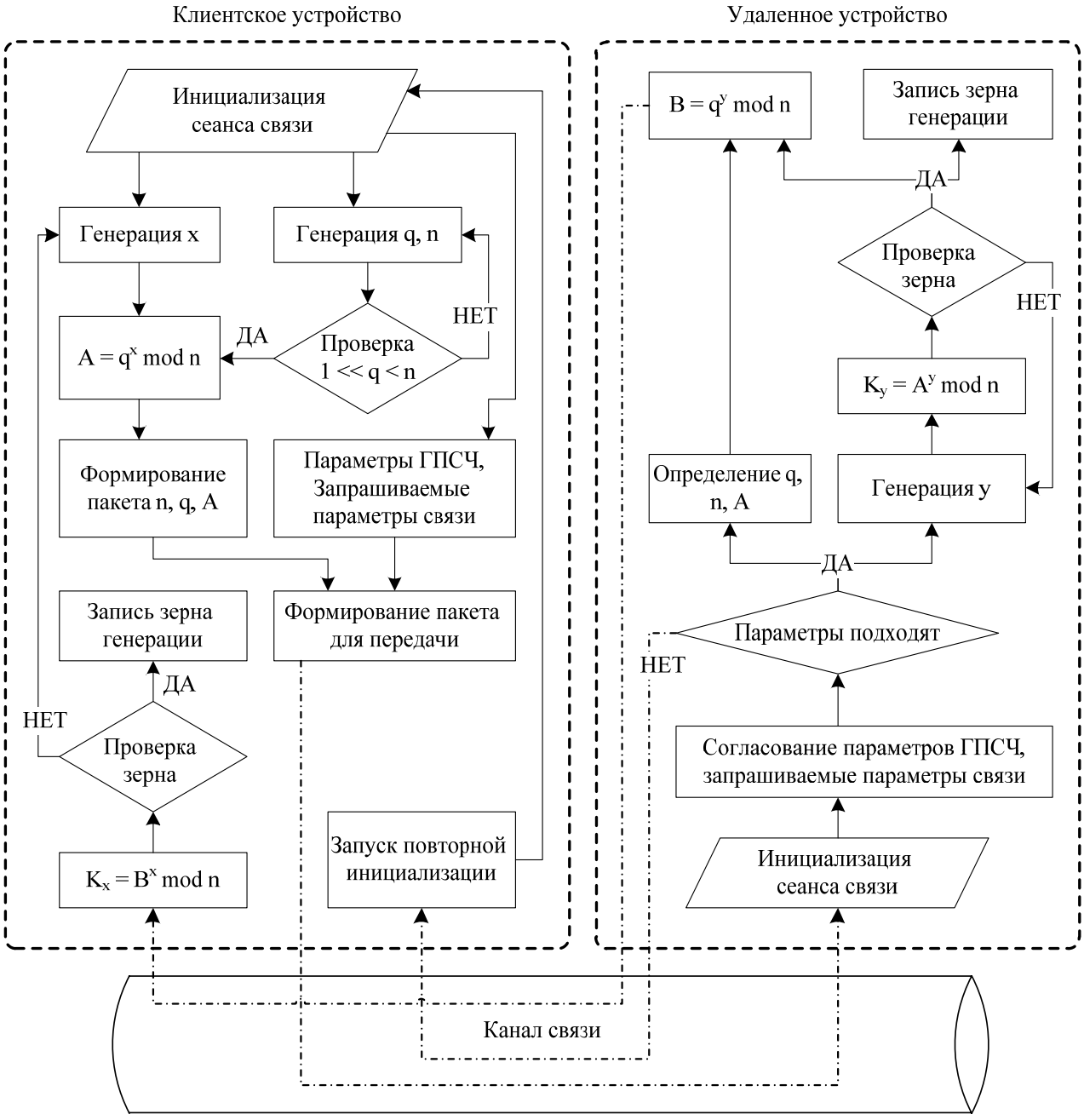

Рис. 4. Алгоритм согласования параметров шифрования

Вычисленное значение K_y проверяется в качестве начального зерна инициализации выбранного ГПСЧ. Если полученное зерно генерации не удовлетворяет выдвигаемым требованиям для заданного ГПСЧ, то проводится цикл генераций случайного большого числа y, вычисление значения K_y и повторная проверка на соответствие заданным параметрам. Условием окончания указанного цикла является нахождение такого зерна генерации, которое будет удовлетворять необходимым условиям. Найденное зерно генерации сохраняется в памяти устройства и передается в подпрограмму, реализующую функционирование ГПСЧ.

Следующим этапом установления сеанса связи является пересылка полученного числа B обратно клиентскому устройству, которое вычисляется

$$K_x = B^x \ mod \ n.$$

Описанный алгоритм гарантирует, что числа K_y и K_x равны между собой и могут быть использованы в качестве секретного ключа для шифрования который представляет собой начальное зерно инициализации ГПСЧ. Фактически указанные пакеты данных пересылаются в открытом виде и поэтому злоумышленнику доступны значения параметров q, n, A, B, которые он может извлечь из перехваченных сообщений. Криптографическая стойкость данного алгоритма основана на проблеме дискретного логарифмирования, при котором невозможно вычислить [12]

$$K=q^{xy} \bmod n.$$

по известным значениям n, q, $A=q^x \bmod n$ и $B=q^y \bmod n$.

После согласования начального зерна инициализации ГПСЧ, а также дополнительных параметров связи в обоих устройствах сеанс связи считается установленным и далее происходит шифрование и обмен полезными данными между оконечными устройствами в двунаправленном режиме.

Шифрование передаваемых данных. Предлагаемая система защиты (рис. 5) работает в потоковом режиме, то есть зашифрованные данные, которые передаются по каналу связи, расшифровываются в оконечном устройстве и дальше обрабатываются другими приложениями или аппаратными устройствами. Поэтому реализуемый режим функционирования системы защиты не предполагает сохранение переданных данных в зашифрованном виде.

Функционирующий защитный комплекс состоит из двух подсистем:

- программная подсистема;
- аппаратная подсистема.

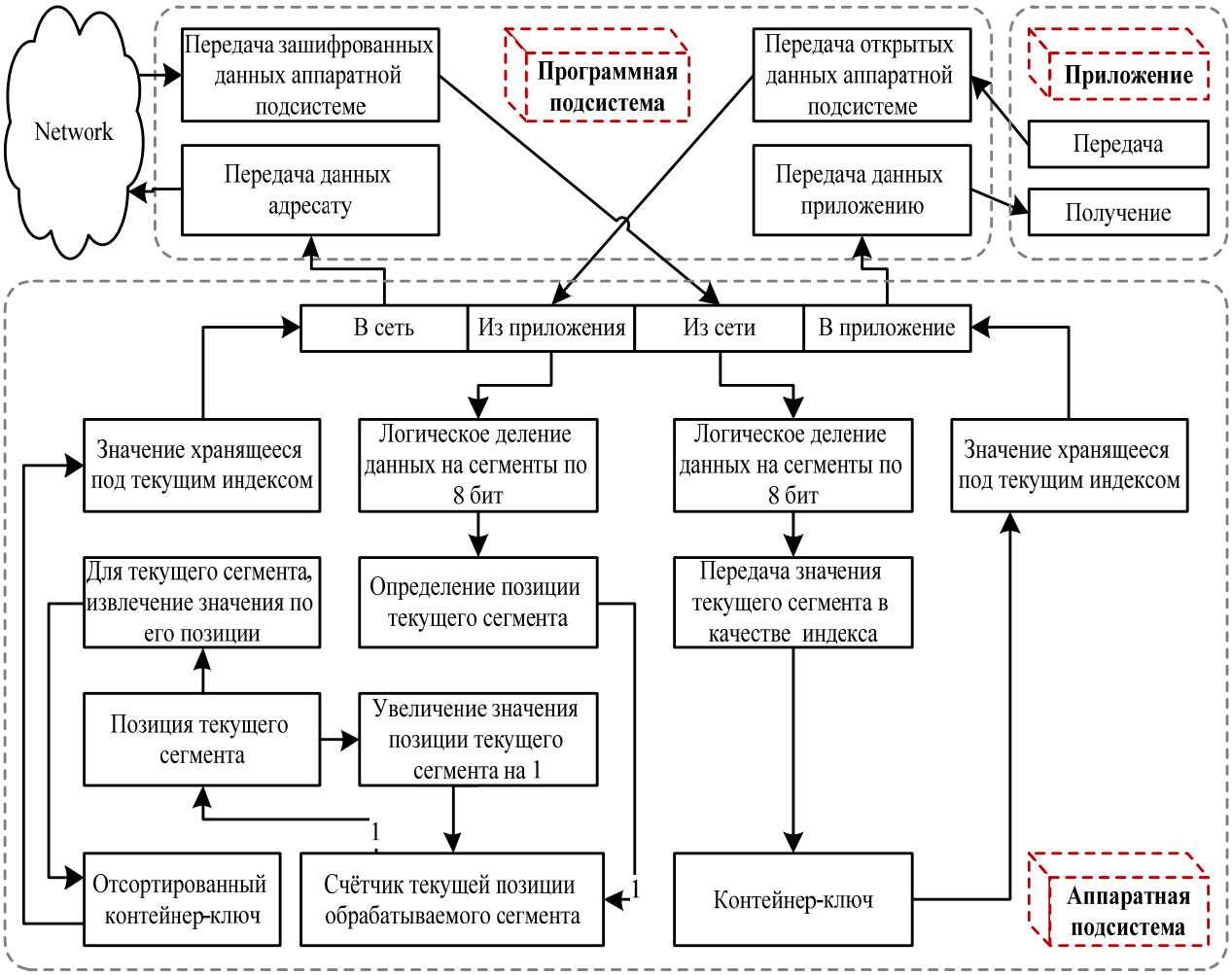

Рис. 5. Общий алгоритм работы защитного комплекса

Программная подсистема. Программная подсистема предоставляет необходимый набор API-функций для сторонних приложений, а также реализует функционал, описанный далее указанным псевдокодом.

//Мониторинг сетевой активности приложение: ожидание передачи данных приложением;

//ожидание поступления данных приложению

while*((из приложения не поступают данные) || (приложению не поступают данные))*

{

Анализ сетевой активности приложения;

Прослушивание транспортного порта, на котором должно ожидать подключения приложение.

}

//начало поступления данных

if*(данные из приложения) code = 1;*

//Вызов нового потока для шифрования и дешифрования

CreateThread(NULL, NULL, Thread, &code, NULL, &thID);

Thread(code)

{

if*(code == 1) //шифрование – данные из приложения*

{

Блокировать передачу данных в сеть непосредственно из приложения;

Передача управляющей команды аппаратной подсистеме – «шифровать»;

Направление открытых данных от приложения на аппаратную подсистему;

Приём зашифрованных данных от аппаратной подсистемы;

Формирование сетевых пакетов по образу приложения;

Передача зашифрованных данных адресату от имени приложения;

}

else *//дешифрование – данные из сети*

{

Блокировать передачу данных из сети непосредственно в приложение;

Передача управляющей команды аппаратной подсистеме – «дешифровать»;

Направление зашифрованных данных из сети на аппаратную подсистему;

Приём расшифрованных данных от аппаратной подсистемы;

Формирование сетевых пакетов по образу получаемых из сети;

Передача расшифрованных данных приложению;

}

}

Аппаратная подсистема. Для приложений аппаратная подсистема представляется в виде набора функций, которые вызываются при вызове заданных API-функций из программной подсистемы. Такая реализация позволяет гибко модифицировать различные подпрограммы без необходимости повторного перепрограммирования всего устройства. Кроме этого появляется возможность добавлять в аппаратную часть новые реализации алгоритмов ГПСЧ, протоколов согласования, специализированных функций обработки различного контента и т. п.

В предлагаемой в данной работе реализации аппаратного устройства используется алгоритм шифрования/дешифрова-ния, описываемый следующим псевдокодом:

A[n] *//контейнер-ключ*

B[256][n/256] *//отсортированный контейнер-ключ*

C[256] *//Счётчик текущего положения обрабатываемого элемента*

while*(данные не поступают)*

ожидать поступления данных

//начало поступления данных

if*(данные из приложения) code = 1;*

//Вызов нового потока для шифрования и дешифрования

CrateThread(NULL,NULL,Thread,&code,NULL, &thID);

```
Thread(code)
{
if(code == 1) //шифрование – данные из приложения
{
        ifstream encrypting ("открытые данные из приложения");
        ifstream encrypted ("зашифрованные данные в сеть");
        while(encrypting.good())
//пока поступают данные
        //Передача значения encrypting в качестве индекса массива A[]
        encrypted = A[encrypting];
        encrypting.close();
        encrypted.close();
}
else //дешифрование – данных из сети
{
        ifstream decrypting ("зашифрованные данные из сети");
        ifstream decrypted ("открытые данные в приложение");
        while(decrypting.good())
//пока поступают данные
        {
                //Определение позиции текущего сегмента
                count = C[decrypting];
                //определение количество элементов в строке decrypting массива B[]
                cell=sizeof(B[decrypting][])/
sizeof(B[decrypting][0]);    //Увеличение значения позиции текущего сегмента на 1
                if(C[decrypting]<cell)
                        C[decrypting]++;
                else
                        C[decrypting]=0;
                //Для текущего сегмента, извлечение значения по его позиции
                decrypted=B[decrypting][count];
        }
        decrypting.close();
        decrypted.close();
}
}
```

Заключение

Создание программно-аппаратных комплексов защиты на базе использования не раскрываемых шифров является перспективным направлением исследований в области информационной безопасности. Ввиду того, что данный класс шифров обладает доказанной криптостойкостью и соответственно не может быть взломан, это позволяет использовать гарантируемую криптостойкость, исчисляемую сотнями и тысячами лет. Особым преимуществом описанного комплекса защиты является работа в потоковом режиме, что позволяет применять его в системах аудио-видео трансляции, а также в различных распределенных системах с повышенными требованиями к параметрам используемых каналов связи. В настоящее время разрабатываемый аппаратно-программный комплекс проходит экспериментальное исследование в настольных компьютерах, локально-корпоративной сети компьютеров, а также в глобальной сети Интернет.

1. *Gilbert S. Vernam*. Cipher Printing Telegraph Systems For Secret Wire and Radio Telegraphic Communications, // J. of the IEEE. – 1926. – Vol. 55. – P. 109–115.
2. *Шеннон К.* Работы по теории информации и кибернетике. – М.: Изд-во иностр. литры, 1963. – 830 с.
3. *Зубов А.* Совершенные шифры. – М.: Гелиос АРВ, 2003. – 160 с.
4. *Алишов Н.И., Марченко В.А., Оруджева С.Г.* Косвенная стеганография как новый способ передачи секретной информации // Комп'ютерні засоби, мережі та системи. – 2009. – № 8. – С. 105–112.
5. *Stallings W*. Cryptography and network security: principles and practice. – New York: Prentice Hall, – 2006. – 680 p.
6. *Алишов Н.И., Алишов А.Н., Бойко А.В.* и др. Технология системной интеграции аппаратных и программных средств защиты информации // Проблеми програмування. – 2010. – № 4. – С. 75–89.
7. *FIPS*-140 –3. DRAFT Security Requirements for Cryptographic Modules (Revised Draft). – http://csrc.nist.gov/publications/ PubsDrafts.html# FIPS-140–3.

8. *Фергюсон Н., Шнайер Б.* Практическая Криптография. – М.: Вильямс, 2005. – 416 с.

9. *ANSI* X9.31-1998: Public Key Cryptography Using Reversible Algorithms for the Financial Services Industry (rDSA). – American National Standards Institute, 1998. – 66 p.
10. *Pseudorandom* Number Generator (ANSI X9.17/X9.31 PRNG). – http://www.xilinx.com/products/ipcenter/ANSIX917X931PRNG.htm
11. *Rescorla E.* "Diffie-Hellman Key Agreement Method": RFC2631. – East Palo Alto, RTFM Inc., 1999. – 12 p.
12. Dan Boneh The Decision Diffie-Hellman problem // Lecture Notes in Computer Science. – 1998. – Vol. 1423/1998. – P. 48–63.

Получено 19.04.2011

Об авторах:

Алишов Надир Исмаил-оглы,
доктор технических наук,
ведущий научный сотрудник,

Марченко Виталий Анатолиевич,
кандидат технических наук,
старший научный сотрудник,

Мищенко Александр Николаевич,
аспирант.

Место работы авторов:

Институт кибернетики имени В.М. Глушкова
НАН Украины.
03187, Киев-187,
проспект Академика Глушкова, 40.
Тел.: 526 3427.