Д.Г. Гаев

Основы прикладной криптографии. Часть 2: криптография в век компьютеров

Необходимость защищать свои тайны от посторонних глаз и ушей существовала всегда, поэтому криптография – это дисциплина, которая почти так же стара, как и само человечество. Во второй статье из цикла об основах прикладной криптографии рассказывается о криптографии современной, криптографии в век компьютеров (начало см.: Гаев Д.Г. Основы прикладной криптографии (часть 1) / Д.Г. Гаев // Культура в современном мире. –2014. – № 3).

В первой части статьи рассматривалась криптография докомпьютерной эпохи. Поговорим теперь о криптографии *современной*, история которой уже связана с компьютерными технологиями неразрывно. Появление компьютеров стало революцией во многих областях человеческой деятельности, и защита важной информации не исключение. Поэтому не удивительно, что создателем современной криптографии является одновременно и один из отцов современной информатики — великий Клод Элвуд Шеннон.

Можно сказать, что именно Шеннон поставил теоретическую криптографию на надежный математический фундамент. Изучая проблему устойчивости шифров (и их ключей) к взлому, он ввел важнейшие понятия *конфузии* и *диффузии*. Если не вдаваться в математические тонкости, их можно определить примерно так: под «конфузией» понимается искажающее воздействие самого ключа на шифруемые данные, а под «диффузией» — взаимно-искажающее влияние этих данных друг на друга. Хороший криптоалгоритм стремится максимизировать как конфузию, так и диффузию. Взаимно дополняя друг друга, они вызывают т. н. *«лавинный эффект»*, при котором даже минимальные изменения в исходных данных (или в секретном ключе) способны исказить выходной результат алгоритма до полной неузнаваемости. Таким образом, в результате шифрования тщательно маскируются любые статистические особенности кодируемого текста (такие, как часто повторяющиеся слова, относительная частота отдельных символов или биграмм и т. д.), т. е. именно то, что традиционно давало криптоаналитикам лазейки для взлома шифра. Работы Шеннона также дали криптографам *математический аппарат* для проверки стойкости того или иного шифра, основанный на анализе статистических особенностей шифротекста.

Современная теория шифров заставила по-новому взглянуть на проблему обеспечения их секретности. Это может показаться неожиданным, но секретность самого алгоритма шифрования необязательно следует признавать его достоинством. Более того, именно

несекретные (т. е. открыто опубликованные и широко обсуждаемые) криптоалгоритмы давно признаны более надежными. Дело в том, что любой широко известный и активно применяемый криптоалгоритм неоднократно подвергался серьезнейшей «проверке на разрыв» командами экспертов-криптологов со всего мира. Профессионалы накопили богатый опыт разнообразных криптографических атак на шифры: и только алгоритм, который их уверенно выдержал, можно считать надежным и пригодным для серьезного использования. В отношении же любого «секретного» шифра (как правило, созданного для частного или ограниченного применения) такой уверенности в принципе быть не может. Более того, шифр, который искренне казался своим создателям исключительно надежным, иногда может быть взломан настоящими профессионалами за очень короткое время. Вот почему для обеспечения секретности желательнее использовать хорошо известный (и сертифицированный настоящими профессионалами) алгоритм, чем алгоритм неизвестный практически никому, но изобретенный дилетантами.

Поэтому в настоящее время единственным реальным секретом шифра обычно является его *ключ:* держать в тайне все остальное зачастую просто нет необходимости. Впрочем, еще до Шеннона этот принцип получил определенную известность в виде т. н. «*второго критерия Керкгоффса*». Хотя шесть «критериев» устойчивости криптосистемы были предложены автором фундаментальной работы по военным шифрам Огюстом Кергхоффсом еще в XIX веке, их стоит привести здесь целиком:

1. *Система должна быть физически (если не математически) невскрываемой;*
2. *Нужно, чтобы не требовалось сохранение системы в тайне: попадание системы в руки врага не должно причинять неудобств;*
3. *Хранение и передача ключа должны быть осуществимы без помощи бумажных записей; корреспонденты должны располагать возможностью менять ключ по своему усмотрению;*
4. *Система должна быть пригодной для сообщения через телеграф;*
5. *Система должна быть легко переносимой, работа с ней не должна требовать участия нескольких лиц одновременно;*

Наконец, от системы требуется, учитывая возможные обстоятельства её применения, чтобы она была проста в использовании, не требовала значительного умственного напряжения или соблюдения большого количества правил.

(Пусть упомянутый здесь «телеграф» сейчас и заменен Интернетом, но в остальном всё процитированное звучит достаточно актуально и по сей день.)

Симметричные криптоалгоритмы

Основу современной компьютерной и интернет-криптографии составляют *симметричные* криптоалгоритмы, т. е. алгоритмы, в которых для шифрования и дешифрования сообщения используется *один и тот же ключ*. В настоящее время подобных алгоритмов разработано довольно много (но, несмотря на это, постоянно придумываются новые). Рассказ о них мы начнем с рассмотрения некоторых их общих свойств.

Например, все алгоритмы, о которых пойдет речь ниже, являются *блочными*. Это означает, что исходное сообщение разбивается на блоки данных фиксированного размера (обычно измеряемого в битах). Используемый ключ шифрования обычно также приводится к некой фиксированной битовой разрядности. Сама процедура шифрования состоит в применении базовой *функции шифрования* (*криптофункции*) вместе с заданным ключом последовательно к каждому блоку исходных данных. Функция шифрования чаще всего состоит в последовательном применении определенного набора преобразований. Каждое из этих преобразований (называемое *циклом*, или *раундом*, шифрования) состоит в выполнении над данными определенных (более или менее тривиальных) арифметических и логических операций: сложения, вычитания, «исключающего ИЛИ», сдвигов и перестановок битов в определенной последовательности и т. д. Большинство этих операций в современных процессорах являются встроенными и очень эффективно реализованными, что весьма важно, так как в процессе шифрования каждого блока их приходится выполнять множество раз. Многократное применение циклов шифрования вызывает вышеупомянутый «*лавинный эффект*», т. е. сильное взаимное искажение данных как блока, так и ключа (тем самым доводя до предела «конфузию» и «диффузию» по Шеннону). Для некоторых алгоритмов шифрование блока состоит в чередовании нескольких *различных* видов раундов кодирования (в соответствии с определенным *расписанием*).

Также очевидно и то, что процесс шифрования должен быть *обратим*, т.е. для каждой из циклических операций шифрования должна быть определена *обратная операция*, уничтожающая ее результаты. Таким образом, если шифрование блока данных состоит в применении N “шифрующих” циклов, то его полная дешифровка обычно требует такое же количество “дешифрующих”. Кроме того, в силу взаимной симметрии «прямого» и «обратного» алгоритмов для шифрования блока данных (обычно) требуется примерно столько же времени и вычислительных ресурсов, как и для дешифрования.

Из выходных (т. е. зашифрованных) блоков конструируется зашифрованное сообщение. Для этого также могут применяться разные методы:

- *Электронная кодовая книга* (**ECB**, *Electronic Code Book*)

Самый простой метод последовательного шифрования: каждый блок исходных данных на выходе просто заменяется блоком шифротекста. Достоинства понятны: любой блок зашифрованных данных не зависит от других блоков (и даже безвозвратная потеря части этих блоков не мешает расшифровать оставшуюся часть сообщения). Недостатки не менее очевидны: это самый слабо защищенный из всех методов блочного шифрования. Например, одинаковые блоки исходных данных при идентичном ключе дают одинаковый результат, что достаточно легко может быть использовано для криптоатаки.

- *Сцепление блоков шифротекста* (**CBC**, *Cipher Block Chaining*)

Более устойчивый способ шифрования состоит в том, что каждый следующий блок шифротекста формируется на основе *предыдущего* зашифрованного блока. В CBC это делается путем применения к данным (*до шифрования*) и к результату шифрования предыдущего блока операции «исключающее ИЛИ». Таким образом, все блоки шифротекста логически образуют *последовательную цепочку*, и для дешифровки какого-либо блока требуется знание всех предшествующих блоков. С другой стороны, в отличие от ECB, даже значительные повторы в кодируемом сообщении никак не проявляются в результате.

- *Обратная связь по шифротексту (CFB, Cipher Feedback Mode)*

Отличается от CBC в основном тем, что криптофункция циклически применяется к *одному и тому же* блоку, к которому на каждом шаге «примешивается» (обычно также через «исключающее ИЛИ») содержимое очередного блока данных. Как и для CBC, утрата или искажение какого-либо блока сделают невозможным дешифровку оставшейся части сообщения, но любые повторы в исходном сообщении маскируются достаточно надежно.

- *Обратная связь вывода* (**OFB**, *Output Feedback Mode*)

В отличие от CFB влияние кодируемых данных на результат уменьшается до минимума (циклически шифруется один и тот же блок, а очередные блоки данных просто «смешиваются» с результатом путем применения «исключающего ИЛИ»). Криптографически этот способ, безусловно, слабее, чем CFB, но при этом он может быть реализован существенно быстрее в случае, когда ключ известен заранее (и остается неизменным), и меняются лишь шифруемые данные. Кроме того, как и в ECB, утрата промежуточных блоков

данных нефатальна для расшифровки остальной части сообщения.

Разные криптоалгоритмы также могут использовать разные способы формирования (например, из строки текста) ключа с требуемой разрядностью и по-разному дополнять исходные данные (padding) требуемым числом битов для формирования целого блока.

Рассмотрим теперь и сами блочные шифры, активно применяемые в компьютерной криптографии в настоящее время.

- **DES**

Криптоалгоритм DES (Data Encryption Standard) был разработан корпорацией IBM в 1977 г. по заказу ряда правительственных ведомств США. В течение нескольких десятилетий он был фактическим государственным стандартом шифрования в США (и приобрел известность и в остальном мире).

Блочный шифр, разрядность каждого блока данных — 64 бита, разрядность ключа — 56 битов. Выполняется 16 циклов шифрования для каждого блока. Дополнительно к блокам данных применяются фиксированные начальные и финальные перестановки битов. На каждом цикле шифрования большая часть операций задается набором предопределенных таблиц.

Поскольку алгоритм DES существует довольно давно, он *очень* пристально изучался криптоаналитиками со всего мира и признан относительно надежным. Его основная слабость связана с относительно небольшой длиной ключа (56 битов): это уже слишком короткий ключ по нынешним временам. Так, уже получили известность примеры успешного взлома шифров DES методом «лобовой атаки» за довольно короткое время (правда, с применением специализированных и дорогостоящих компьютеров).

> *Используя суперкомпьютер стоимостью 250 тыс. долл., сотрудники RSA Laboratory "взломали" утвержденный правительством США алгоритм шифрования данных (DES) менее чем за три дня. (Предыдущий рекорд по скорости взлома был установлен с помощью огромной сети, состоящей из десятков тысяч компьютеров, - 39 дней). На специально организованной по этому случаю пресс-конференции ученые с беспокойством говорили о том, что злоумышленники вряд ли упустят случай воспользоваться подобной уязвимостью.*
>
> *Эксперимент проходил в рамках исследования DES Challenge II, проводимого RSA Laboratory под руководством общественной организации Electronic Frontier Foundation (EFF), которая занимается проблемами информационной безопасности и личной тайны в Интернет.*
>
> *Суперкомпьютер, построенный в RSA Laboratory для расшифровки данных, закодированных методом DES по 56-разрядному ключу, получил название EFF DES Cracker.*
>
> *Как утверждали правительственные чиновники и некоторые специалисты, для взлома кода DES требуется суперкомпьютер стоимостью в несколько миллионов долларов*[1].

[1] [Электронный ресурс]. – Режим доступа: URL: http://www.osp.ru/cw/1998/28-29/30844/

Успешность некоторых современных атак на DES — это один из главных стимулов для создания альтернативных, более надежных систем шифрования.

- **Double DES / Triple DES**

Фактически, эти алгоритмы основаны на том же DES, но применяемом *дважды* с двумя разными ключами (всего 112 бит ключевой информации) или *трижды* с тремя ключами (168 бит). Понятно, что это делает их более стойкими к взлому и для этих алгоритмов вполне можно использовать уже имеющиеся программные или быстрые аппаратные, в том числе в виде специальных микросхем, реализации DES.

- **ГОСТ 28147-89**

Популярный отечественный криптографический стандарт. Предположительно, он был создан где-то в 1980-х и не без участия 8-го управления КГБ (сейчас это ФАПСИ — Федеральное агентство правительственной связи и информации). Долгое время вся информация об этом алгоритме имела гриф «Для служебного пользования», но в 1994 году он был полностью рассекречен.

Блочный шифр, разрядность ключа — 256 бит, разрядность блока — 64 бита, циклов преобразования — 32. По сравнению с DES, алгоритм является довольно простым и, вместе с тем, относительно эффективно реализуемым.

Единственным важным параметром «настройки» алгоритма является так называемый “узел замены” (или *S-блок*) — прямоугольная матрица (16 столбцов на 8 строк), содержащая числа от 0 до 15. Надежность алгоритма существенно зависит от выбранного «узла замены». Опубликованные и хорошо известные узлы замены (например, используемые Центральным Банком РФ), по-видимому, можно считать «надежными». С момента публикации алгоритм тщательно изучался многими криптологами, но сколько-нибудь перспективной криптоатаки не найдено. Все предлагаемые варианты «взлома» носят сугубо умозрительный характер, например: *в 2004 г. группа специалистов из Кореи предложила атаку, с помощью которой, используя дифференциальный криптоанализ на связанных ключах, можно получить с вероятностью 91,7% 12 бит секретного ключа. Для атаки требуется 2^{35} выбранных открытых текстов и 2^{36} операций шифрования*[2].

Однако, учитывая астрономическое количество необходимых для подобного взлома «открытых текстов» (2^{35} — это примерно тридцать миллиардов!) этот способ вряд ли реализуем на практике.

- **IDEA**

Название алгоритма интерпретируется как International Data Encryption Algorithm

[2] [Электронный ресурс]. – Режим доступа: URL: http://mind-control.wikia.com/wiki/%D0%93%D0%9E%D0%A1%D0%A2_28147%E2%80%9489

(«Международный алгоритм шифрования данных»). Из названия очевидно, что он предлагается в качестве открытого и международного криптографического стандарта (разработчики — сотрудники Швейцарской высшей технической школы в Цюрихе).

Блочный шифр, разрядность ключа — 128 битов, разрядность каждого блока — 64 бита. Для шифрования каждого блока используется всего 8 раундов шифрования (и в дополнение к этому одно финальное «выходное преобразование»). На всех этапах шифрования используются только самые простые арифметические операции (модульное сложение, модульное умножение, «исключающее ИЛИ»). За счет этого, алгоритм обеспечивает довольно высокую скорость работы (примерно в 2—2,5 раз быстрее DES). При этом надежность его также признана существенно лучшей, хотя были обнаружены некоторые классы криптографически более «слабых» ключей. Имеется довольно много аппаратных реализаций (например, алгоритм реализован в шифровальном устройстве VINCI).

- **Blowfish**

Алгоритм разработан и опубликован в 1993 году известным американским криптологом Брюсом Шнайером (Bruce Schneier).

Блочный шифр, разрядность блока данных — 64 бита, разрядность ключа — варьируется (от 32 до 448 бит). Шифрование каждого блока выполняется за 16 циклов. Важными дополнительными параметрами шифра являются так называемый *ключевой массив* P (содержит 18 32-битовых чисел) и четыре *таблицы замены* S (каждая содержит 256 чисел). Эти параметры алгоритма несекретны.

К основным достоинствам Blowfish можно причислить то, что это очень быстрый алгоритм (возможно, один из самых быстрых), и при этом достаточно надежный (при условии правильно выбранных массивов *P* и *S*). Кроме того (в отличие от IDEA), он полностью свободен от патентных ограничений. Поддерживается практически всеми известными криптографическими программами.

- **Twofish**

«Модернизированный» и усложненный вариант Blowfish. Размер блока — 128 бит, размер ключа — 128, 192 или 256 бит. Предполагается отсутствие «слабых» ключей. Как и Blowfish, свободен от патентных ограничений. К недостаткам можно отнести то, что он относительно медленнее, чем AES.

- **Camellia**

Шифр разработан в Японии (при участии таких известных корпораций, как «Мицубиси» и «Nippon Telegraph & Telephone Corp.»). Защищен патентом, но при этом может использоваться относительно свободно. В Японии сертифицирован и рекомендован для промышленного и государственного использования.

Размер блока — 128 бит, ключа — 128, 196 или 256 бит, циклов шифрования — 18 (128-битный ключ) или 24.

- **AES (Rijndael)**

Входной блок данных — 128 бит, разрядность ключа — можно выбирать из 128, 192 или 256 бит. В зависимости от выбранной длины ключа — применяется 10, 12 или 14 раундов шифрования.

Один из самых популярных криптоалгоритмов в настоящее время. В его пользу говорит и то, что он в настоящее время принят в качестве государственного стандарта шифрования в США вместо DES.

> *В июне 2003 года* Агентство национальной безопасности *США постановило, что шифр AES является достаточно надёжным, чтобы использовать его для защиты сведений, составляющих государственную тайну (англ. classified information). Вплоть до уровня SECRET было разрешено использовать ключи длиной 128 бит, для уровня TOP SECRET требовались ключи длиной 192 и 256 бит* [8].

Впрочем, парадоксальным образом этот же алгоритм так же стал источником самых серьезных проблем, с которыми АНБ пришлось столкнуться за свою историю! Когда основатель Wikileaks Джулиан Ассанж принял решение опубликовать знаменитый пакет конфиденциальных документов из США (в том числе свои «афганское» и «иракское» досье) вместе с ними был опубликован и зашифрованный файл «подстраховки». Предположительно, он содержал более полные (и, что важно, *избавленные от купюр*) версии многих секретных документов — и Ассанж грозился опубликовать ключ к шифру, если он будет арестован.

Этот файл был зашифрован алгоритмом AES с 256-битовым ключом.

Асимметричная криптография и Интернет

Как мы видим, типов симметричных шифров существует очень много. Однако у них всех имеется общая проблема — потребность в *едином ключе*, который известен обоим участникам обмена кодированной информацией. Традиционно, использование любых симметричных шифров предполагает и специальный «секретный» канал информации (вроде курьера фельдегерской службы) для передачи ключей к этим шифрам. Очевидно, что это непросто реализовать в Интернете, который обычно открыт для всех.

Напомним: основная сила Интернета заключается в его гибкости и *динамической адаптации* ко всем имеющимся каналам для передачи данных. Конечно, именно это во многом обеспечило победу Интернета над альтернативными сетевыми технологиями. Однако это же становится принципиальной слабостью в случае, когда информация, передаваемая через Интернет, вдобавок должна быть *защищена* от подсматривания или изменения. Ведь никто не может предсказать заранее, через какое количество промежуточных узлов пройдет

эта информация на пути к адресату (и сколько любопытствующих посторонних получат возможность с ней познакомиться в процессе).

Общение двух (или более) участников через Интернет неявно предполагает, что какие-либо иные, более защищенные каналы коммуникации у них отсутствуют. Послать дипкурьера с шифрами (в пристегнутом к запястью чемоданчике) для простого пользователя Интернета подобное вряд ли осуществимо! Использовать любой из вышеперечисленных алгоритмов шифрования, разумеется, вполне возможно, но ведь перед этим обеим сторонам придется договориться об используемом ключе. А посылать его через Сеть *в открытом виде* не имеет смысла: если канал связи «прослушивается» какой-либо «третьей стороной», то ключ тоже немедленно станет им известен. После чего, конечно, «третьей стороне» уже практически ничто не помешает читать весь поток зашифрованных данных с той же легкостью, что и обоим легитимным участникам.

Выходом из этой тупиковой ситуации является применение методов *асимметричной криптографии*. Все эти способы предполагают тот или иной способ обмена ключами, делающий их перехват трудным или же лишенным практического смысла. Если вся симметричная криптография основана на том, что большинство математических операций *легко обратимы*, то асимметричная основывается на прямо противоположных идеях., то есть на том, что некоторые математические вычисления *труднообратимы* (и «в одну сторону» могут быть вычислены существенно проще, чем в другую).

В самой идее ничего принципиально нового нет: уже в школе все узнают, что, к примеру, делить числа «столбиком» заметно труднее, чем их умножать. Кстати, подобная асимметрия между умножением и делением справедлива и для компьютеров (невзирая на то, конечно, что они выполняют умножение и деление настолько быстро, что человек вряд ли сумеет заметить разницу). Но вот для серьезных криптографических применений уже требуются математические операции, «обращение» которых имеет более высокую вычислительную сложность, и не в несколько раз, а *на много порядков*. Именно подобные операции и являются основой для *криптографии с открытым ключом*. Сложность «обращения» операции возведения в степень по модулю (*дискретному логарифмированию*) — это основа для алгоритмов Диффи-Хеллмана и Эль-Гамаля. А сложность разложения очень больших чисел на простые множители («*факторизации*») — это именно то, на чем основан алгоритм RSA.

Протокол Диффи-Хеллмана

Протокол Диффи-Хеллмана был одним из первых изобретений, позволяющих решить проблему безопасного обмена данных в Интернете. Сам по себе это *не криптографический протокол:* единственная задача, которую он решает, — безопасный *обмен ключами* через

небезопасную среду (Интернет или какие-либо другие незащищенные каналы связи). Точнее говоря, протокол позволяет двум участникам обмена получить *общий ключ* таким образом, что никакая «третья сторона» не способна его перехватить в процессе передачи. Понятно, что располагая общим ключом, обе стороны легко могут наладить засекреченный обмен информацией, используя для этого любой из хорошо известных протоколов симметричной криптографии (такой, как IDEA, Blowfish или AES).

Впервые описание этого протокола опубликовано еще в 1976 году. Свое название он получил в честь своих разработчиков — математиков-криптографов из Стэнфорда — Уитфилда Диффи и Мартина Хеллмана.

С точки зрения математики, основная идея здесь весьма проста и основана на том, что всем известно из школьной программы: результат операции многократного потенцирования (возведения в степень) *не зависит* от порядка, в котором они выполняются, то есть (для любых целых *a*, *b* и *c*) справедливо следующее тождество:

$$(a^b)^c = a^{(bc)} = a^{(cb)} = (a^c)^b$$

Это справедливо, когда все операции возведения в степень выполняются по некоторому модулю. Диффи и Хеллман предложили использовать следующую схему:

- оба участника (обозначим их просто **A** и **B**) выбирают два целых числа: основание степени (*G*) и модуль (*P*). (Оба этих параметра *несекретны* и могут передаваться через сеть открыто.);
- участник **A** придумывает *случайный ключ a* и передает участнику **B** число *U*, представляющее собой *G* в степени *a* по модулю *P*:

$$U = G^a \bmod P;$$

- участник **B** придумывает *случайный ключ b* и передает участнику **A** число *V*, представляющее собой *G* в степени *b* по модулю *P*:

$$V = G^b \bmod P;$$

- участник **A**, получив от **B** число *V*, возводит его в (известную ему, но не **B**!) степень *a* по модулю P, получая число *K*:

$$K = V^a \bmod P = G^{ba} \bmod P;$$

- Участник **B**, получив от **A** число *U*, возводит его в (известную ему, но не **A**!) степень *b* по модулю *P*, *также* получая число *K*:

$$K = U^b \bmod P = G^{ab} \bmod P;$$

- В результате и **A**, и **B** получили *одно и то же* (ввиду вышеприведенного тождества!) число *K*, которое и является требуемым ключом.

На чем основана *секретность* этого метода? На том, что, даже зная *G* и *P*, и успешно перехватив значение *U* (или *V*), невозможно вычислить значение *a* (или *b*). Точнее говоря, это, конечно, возможно, однако очень и очень трудоемко. Эта операция называется *дискретным логарифмированием*, и в настоящее время неизвестны способы, позволяющие выполнить её за разумное время, когда значение *P* достаточно велико (например, несколько сотен десятичных цифр).

Алгоритм Диффи-Хеллмана легко может быть обобщён и для большего числа участников. Из его серьезных недостатков нужно отметить то, что он надежно защищает от *подслушивания*, но не от прямого *подлога*. Другими словами, хотя простой перехват значений *U* или *V* ничего не даст «третьей стороне», но вот возможность вмешаться в процесс передачи данных и «подсунуть» участникам **A** и **B** «фиктивные» значения вместо подлинных уже позволяет «обмануть» этот протокол без особого труда. (Впрочем, его можно и усилить за счет дополнительных средств аутентификации, позволяющих успешно проверить подлинность полученных *U* и *V*.)

Криптографический протокол RSA

Наиболее распространенный из современных алгоритмов асимметричной криптографии — RSA — получил свое название по первым буквам фамилий трех своих главных создателей: *Рональда Ривеста* (Rivest), *Ади Шамира* (Shamir) и *Леонарда Адлемана* (Adleman). Первое описание алгоритма увидело свет всего через год после публикации схемы Диффи-Хеллмана и было основано на похожих идеях. Вместе с тем, важно отметить, что RSA уже является полноценным (и самодостаточным) криптоалгоритмом, т. е. не требующим дополнительных технологий для организации зашифрованной связи.

В реализации RSA большую роль тоже играет операция модульного возведения в степень. Работа этого алгоритма основана на свойстве *обратимости* этой операции: два целых числа *d* и *e* считаются *мультипликативно-обратными* по модулю N, если для них (и для любого *x*) будет справедливо следующее тождество:

$$(x^{d})^{e} \pmod{N} = (x^{e})^{d} \pmod{N} = x.$$

Таким образом, если возвести произвольное число *x* сперва в степень *d*, а затем в степень *e*, мы обязательно *вернемся* к предыдущему числу. Это обстоятельство предоставляет возможность использовать одно из этих чисел как ключ шифрования (*открытый*), а другое — как ключ дешифрования (*закрытый*). Для вычисления случайной пары таких чисел Ривест, Шамир и Адлеман предложили следующий алгоритм:

- выбираются два больших простых числа: *P* и *Q*;
- на их основе вычисляются два произведения: $N = P \cdot Q$ и $M = (P-1) \cdot (Q-1)$;
- выбирается *публичный ключ e* (обычно, это небольшое простое число);

- из значения *M* и значения *e* вычисляется *секретный ключ d*, удовлетворяющий условию:

$$d \cdot e \equiv 1 \pmod{M};$$

- пара чисел (*e, N*) образует *публичный ключ RSA*, который может передаваться по сети открыто.
- в то время как пара чисел (*d, N*) является *секретным ключом RSA* (известным только одной из сторон), соотношение, связывающее *d* и *e*, легко может быть переписано в следующей форме: d • e – M • v = 1. Это уравнение (так называемое *диофантово уравнение*) требует решения в целых числах (что возможно, например, с помощью *расширенного алгоритма Эвклида*). Тот факт, что числа *e* и *d* действительно являются *мультипликативно-обратными* по модулю N, следует из *малой теоремы Ферма* и *«китайской теоремы об остатках»*[3].

На чем же основана криптографическая защищенность RSA? Прежде всего, на том обстоятельстве, что даже зная (несекретные) значения *N* и *e*, но, не зная *M*, *P* или *Q*, *не существует* простого способа вычислить *d*. Формально говоря, основную сложность здесь представляет нахождение *P* или *Q* (очевидно, что, узнав одно из этих чисел, путем простого деления легко вычислить и второе). Проблема в том, что для этого необходимо разложить число *N* на множители. Эта задача (называемая *факторизацией*) для больших чисел считается очень трудноразрешимой. Один из самых быстрых из известных современных методов — метод *решета числового поля* (*GNFS*). Но даже с его использованием разложение чисел, используемых в современных версиях RSA, может потребовать много лет работы быстродействующих компьютеров.

Создатели самого алгоритма RSA также были настроены относительно него достаточно оптимистично (может, чересчур). В 1977 году в колонке журнала Scientific American, посвященной математическим играм (которую вел Мартин Гарднер, хорошо известный своими прекрасными книгами по популярной математике), сам Рональд Ривест предложил всем читателям взломать зашифрованное RSA сообщение со следующими несекретными параметрами: значение e = 9007, а значение N равно:

114 3816 257 5788 886 7669 235 7799 761 4661 201 0218 296 7212
423 6256 256 1842 935 7069 352 4573 389 7830 597 1235 639 5870
505 8989 075 1475 992 9002 687 9543 541

(последнее число также получило известность как RSA-129). Ну, а само зашифрованное сообщение (т. е. криптограмма) выглядело так:

[3] Элементарное доказательство справедливости этого имеется, например, в следующей статье [Электронный ресурс]. – Режим доступа: URL: https://ru.wikipedia.org/wiki/RSA.)

9686	9613	7546	2206
1477	1409	2225	4355
8829	0575	9991	1245
7431	9874	6951	2093
0816	2982	2514	5708
3569	3147	6622	8839
8962	8013	3919	9055
1829	9451	5781	5154

Ривест утверждал, что желающим прочитать его шифровку потребуется разложить число RSA-129 на множители, для чего, по его оценкам, придется потратить примерно *сорок квадриллионов* лет! Увы, он недооценивал прогресс в грядущем развитии вычислительной техники. Шифр оставался неразгаданным до 1993 года, когда взломом серьезно занялась группа примерно из 600 добровольцев из разных стран (использующих более 1600 компьютеров, работающих параллельно и координирующих вычисления через Интернет). И примерно *за восемь месяцев* совместного труда удалось разложить RSA-129 на множители — и наконец-таки взломать код Ривеста! Выяснилось, что зашифрованная им фраза была такова: «*THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE*» («*Волшебные слова — брезгливый стервятник*»). Кстати, за взлом шифра участники получили давно обещанную награду в 100 долларов, которую пожертвовали в фонд свободного программного обеспечения.

Помимо прочего, этот взлом подтвердил давно известную истину: не существует *принципиально* «невзламываемых» шифров — вопрос лишь в том, сколько ресурсов может быть затрачено на то, чтобы их взломать. (Понятно, что далеко не всякий шифр сумеет привлечь к себе интерес 600 добровольцев, готовых потратить на его взлом полгода свой жизни.) К тому же, сейчас активно используются существенно более длинные ключи RSA. Например, за успешное разложение на множители числа RSA-1024

135066410865995223349603216278805969938881475605667027524485143851
526510604859533833940287150571909441798207282164471551373680419703
964191743046496589274256239341020864383202110372958725762358509643
110564073501508187510676594629205563685529475213500852879416377328
5339061097505443349998111500569772368909275 63

обещан приз в 100 тыс. долларов!

Относительно недавно выяснилось, что еще до публикации Ривестом, Шамиром и Альдеманом своей работы аналогичная схема шифрования была предложена одним

британским криптографом из GCHQ (британского аналога американского АНБ). К сожалению, его изобретение было немедленно засекречено, ‘то и отдало всю славу создателей открытой криптографии Ривесту с его коллегами.

Безопасные протоколы в Интернете

Только теперь, после обстоятельного рассмотрения идей и методов современной криптографии, мы, наконец, можем рассмотреть то, как реализуется криптографическая защита в сети Интернет. Ее особенностью является то, что в ней *совместно* используются различные методы симметричной и асимметричной криптографии, реализуя, таким образом, *гибридную* схему защиты данных.

Как мы помним, использовать в Интернете исключительно симметричную криптографию бессмысленно, так как она (сама по себе) не реализует надежной защиты используемых ключей. Использовать только асимметричную (такую, как RSA) в принципе возможно. Но основным препятствием для этого является *скорость работы*: при тех ключах RSA, которые сейчас принято считать надежными (как минимум 1024 бита, но рекомендуется 2048), шифровка и дешифровка данных выполняются весьма медленно, т. к. требуют возведения последовательности небольших целых чисел в очень высокие степени. (Из-за особенностей реализации RSA ключ дешифровки обычно намного больше, чем ключ шифрования, поэтому дешифровка будет выполняться еще медленнее!) Основная идея интернет-криптографии состоит в том, что RSA используется только один раз исключительно для *шифрования секретного ключа*, а уж дальше этот ключ используется для защиты самих передаваемых данных.

В современном криптографическом интернет-протоколе TLS (Transport Layer Security) эта идея реализована примерно так:

- сперва клиент и сервер *устанавливают соединение* (в процессе также происходит верификация «подлинности» самого сервера с использованием цифровых сертификатов);
- «надежный» сервер генерирует пару ключей RSA. *Открытый* ключ передается клиенту, *закрытый* ключ остается известен только серверу;
- клиент генерирует ключ для симметричного шифрования (*«сеансовый» ключ*) и передает его серверу, *зашифровав* с помощью открытого RSA-ключа;
- сервер дешифрует полученный от клиента сеансовый ключ, используя известный ему закрытый RSA-ключ;
- после этого сервер и клиент могут без проблем обмениваться закодированной информацией с применением сеансового ключа и любого из широко известных симметричных криптографических протоколов (например, Triple DES, Camellia или AES).

Как легко заметить, сам процесс шифровки-дешифровки передаваемых данных

выполняется достаточно быстро, т. к. для него используются только быстрые симметричные алгоритмы. Асимметричный алгоритм (обычно RSA, хотя возможен и Диффи-Хеллмана) используется только при передаче сеансового ключа и гарантирует, что потенциальный злоумышленник, если и сможет перехватить ключ, не сумеет расшифровать его. Понятно, что для каждого сеанса используются уникальные пары RSA-ключей и свой сеансовый ключ (по окончании сеанса они уничтожаются).

Понятие «транспортный протокол» означает, что этот протокол решает базовую задачу передачи данных между клиентом и сервером, т. е. является фундаментом для любых специфичных прикладных протоколов. Например, для HTTP (работа с World Wide Web), большинства протоколов электронной почты и т. д. Фактически, любой прикладной протокол Интернет может быть достаточно легко адаптирован под TLS, т. е. сделан криптографически безопасным.

Наиболее популярная реализация этого протокола (OpenSSL) доступна для Microsoft Windows и многих других операционных систем. Для нее есть и столь же свободные бесплатные альтернативы, например GnuTLS.

Проблема взлома шифров и квантовые компьютеры

Хотя все рассмотренные выше схемы выглядят достаточно защищенными, но и возможности спецслужб (особенно таких мощных, как Агентство национальной безопасности) тоже не следует недооценивать! Возможно ли, что у них есть свой секретный ключ (или «отмычка») к интернет-шифрам? Хотя достоверно этого, конечно, не знает никто (кроме тех, кому положено), но некоторый свет на это сумел пролить Эдвард Сноуден:

Экс-специалист ЦРУ Эдвард Сноуден, некогда работавший техническим специалистом в спецслужбах Пентагона, продолжает «сливать» секретную, и крайне важную информацию о работе спецслужб родного государства. Очередной кусочек секретной информации Сноуден выдал в 2014 году. На этот раз беглый разведчик Сноуден заявил, что американские национальные агентства, как оберегающие страну, так и ведущие разведку на чужих территориях, работают над созданием квантового компьютера.

Как пояснил Эдвард Сноуден, перед мощью квантового компьютера не сможет устоять ни один из существующих алгоритмов шифрования. Обладая высокой скоростью работы, он с легкостью взломает любые методы шифрования[4].

Здесь требуются некоторые пояснения. Утверждение «не сможет устоять ни один из существующих алгоритмов шифрования» выглядит излишне громким, но появление *работающих* квантовых компьютеров действительно станет для интернет-криптографии

[4] статье [Электронный ресурс]. – Режим доступа: URL: http://hronomir.ru/specsluzhby-ssha-razrabatyvayut-kvantovyj-kompyuter-zayavil-snouden/

весьма серьезным вызовом.

Квантовые компьютеры (далее КВМ) — вычислительные машины, существующие пока исключительно гипотетически. Основные принципы их работы радикально отличаются от традиционных электронных компьютеров (ЭВМ): они основаны не на микроэлектронике, а на *квантовой механике*. В их работе главную роль будет играть *случайность*.

Начиная с самых первых легендарных ЭВМ, любой электронный компьютер представляет собой *детерминированную систему*. Это означает, что в любой момент времени он (вместе с процессором, оперативной памятью и всеми периферийными устройствами) находится в неком четко определенном состоянии. Собственно, и процесс работы компьютера можно формально описать как последовательность переходов из одного состояния в другое, происходящих в точном соответствии с заложенной в него программой. Этот процесс также *детерминирован* (предсказуем): одна и та же программа (с одними и теми же входными данными) всегда выдаст один результат. На этой предсказуемости компьютеров и основана вся классическая кибернетика.

Но с гипотетическим квантовым компьютером все обстоит принципиально иначе. Его внутренние элементы находятся не в каком-либо четко определенном состоянии, а в *нескольких состояниях одновременно*. Если минимальная единица информации электронного компьютера — один бит — всегда находится в одном из двух состояний («0» или «1»), то его квантовый аналог («*кубит*») всегда находится в обоих состояниях сразу. Они различаются лишь относительной *вероятностью*, с которой, прочитав один кубит, мы получим в результате ноль или единицу. То же самое относится к любой группе кубитов: все их возможные состояния различаются лишь относительной вероятностью, находясь в состоянии «квантовой запутанности». Можно сказать, что в квантовом компьютере все возможные вычисления носят не детерминированный, а *вероятностный* характер.

Чем же КВМ могут быть лучше ЭВМ? Секрет потенциальной мощности квантового компьютера заключается в том, что в силу фундаментальных принципов квантовой механики все изменения группы взаимосвязанных кубитов происходят не последовательно, а *одновременно*. Таким образом, если удастся связать достаточно много кубитов, появится возможность создания вычислительной системы, мощность которой будет расти *экспоненциально* от их числа, несмотря на то что сами вычисления при этом останутся весьма «нечеткими». Именно это обстоятельство будет принципиально отличать КВМ от самой быстродействующий ЭВМ.

Известно, что далеко не все алгоритмы могут быть адаптированы для квантовых устройств. Тем не менее, уже опубликован алгоритм, позволяющий разлагать на простые множители произвольное целое число (*алгоритм Шора*), причем за время, зависящее от его

длины линейно. Имеются аналогичные алгоритмы и для дискретного логарифмирования. На данный момент все это, конечно, существует сугубо теоретически: практического смысла в этих алгоритмах нет ввиду отсутствия реализованного «в железе» квантового компьютера с достаточным количеством кубитов. (Более того, неизвестно точно, удастся ли создать такие компьютеры вообще!) Но если все-таки работающие КВМ вдруг появятся, многие виды криптографии с открытым ключом окажутся под угрозой взлома.

Однако хорошей новостью можно считать то, что, раз АНБ так старательно трудится над проблемой квантового компьютера, значит, лучших средств для взлома шифров на данный момент у них *просто нет*. А пока их не существует, криптографические интернет-соединения можно считать защищенными даже от американцев (не говоря уж о любых шпионских организациях с менее развитыми возможностями).

Список литературы:

1. [Электронный ресурс]. – Режим доступа: URL: http://intsystem.org/1120/asymmetric-encryption-how-it-work/ (дата обращения: 1.10.2014).
2. [Электронный ресурс]. – Режим доступа: URL: http://book.kbsu.ru/theory/chapter3/shannon.html (дата обращения: 1.10.2014).
3. [Электронный ресурс]. – Режим доступа: URL: https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BD%D1%86%D0%B8%D0%BF_%D0%9A%D0%B5%D1%80%D0%BA%D0%B3%D0%BE%D1%84%D1%84%D1%81%D0%B0 (дата обращения: 1.10.2014).
4. [Электронный ресурс]. – Режим доступа: URL: https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%94%D0%B8%D1%84%D1%84%D0%B8_%E2%80%94_%D0%A5%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD%D0%B0 (дата обращения: 1.10.2014).
5. [Электронный ресурс]. – Режим доступа: URL: http://lenta.ru/articles/2005/11/12/rsa (дата обращения: 1.10.2014).
6. [Электронный ресурс]. – Режим доступа: URL: https://www.schneier.com/blog (дата обращения: 1.10.2014).
7. [Электронный ресурс]. – Режим доступа: URL: http://www.quickiwiki.com/ru/RSA (дата обращения: 1.10.2014).
8. [Электронный ресурс]. – Режим доступа: URL: https://ru.wikipedia.org/wiki/RSA-%D1%87%D0%B8%D1%81%D0%BB%D0%B0 (дата обращения: 1.10.2014).
9. [Электронный ресурс]. – Режим доступа: URL: http://hronomir.ru/specsluzhby-ssha-razrabatyvayut-kvantovyj-kompyuter-zayavil-snouden/ (дата обращения: 1.10.2014).

Сведения об использованных иллюстрациях:

Иллюстрация содержания [Электронный ресурс]. – Режим доступа: URL: http://www.bezpeka.com/ru/news/2010/09/02/Researchers-discover-flaw-in-quantum-cryptography.htm (дата обращения: 20.08.2014).

Ил. 1. [Электронный ресурс]. – Режим доступа: URL: http://www.masters.donntu.edu.ua/2013/frt/gryshko/ind/index.htm (дата обращения: 20.08.2014).