1.ОПРЕДЕЛЕНИЕ КРИПТОГРАФИИ 2

2.КРИПТОГРАФИЯ — СИСТЕМЫ ШИФРОВАНИЯ 3

3.ДРУГИЕ КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ 8

4.КРИПТОГРАФИЯ — ИСТОРИЯ 9

4.1. РАННЯЯ ИСТОРИЯ КРИПТОГРАФИИ 9

4.2. КРИПТОГРАФИЯ— НАШЕ ВРЕМЯ 13

5. КРИПТОГРАФИЯ: АЛГОРИТМЫ ШИФРОВАНИЯ 16

5.1. СИММЕТРИЧНЫЕ АЛГОРИТМЫ 16

5.2. АСИММЕТРИЧНЫЕ АЛГОРИТМЫ 19

5.3. АЛГОРИТМЫ ХЭШ-ФУНКЦИЙ 21

6.RC4 23

6.1.История 24

6.2.Описание алгоритма 25

6.3.Безопасность 28

6.4.Модификации RC4 31

6.5.Реализация 33

7.ССЫЛКИ И ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА 34

8. ПРИЛОЖЕНИЕ 35

9. ПРИМЕР ВЫПОЛНЕНИЯ КОДА 36

1.ОПРЕДЕЛЕНИЕ КРИПТОГРАФИИ

Криптография или криптология, наука и искусство передачи сообщений в таком виде, чтобы их нельзя было прочитать без специального секретного ключа. В отечественном словоупотреблении термин «криптология» объединяет в себе «криптографию», т.е. шифрование сообщений, и «криптоанализ», т.е. несанкционированное расшифровывание сообщений (также широко употребляются термины «дешифровка / взламывание / ломка / взлом(ка) / вскрытие / раскрытие / раскалывание»). Слово «криптограф» происходит от древнегреческих слов kryptos ‘секрет’ и graphos ‘писание’. Исходное сообщение называется в криптографии открытым текстом, или клером. Засекреченное (зашифрованное) сообщение называется шифротекстом, или шифрограммой, или криптограммой. Процедура шифрования обычно включает в себя использование определенного алгоритма и ключа. Алгоритм - это определенный способ засекречивания сообщения, например компьютерная программа или, скажем, список инструкций. Ключ же конкретизирует процедуру засекречивания.

2.КРИПТОГРАФИЯ — СИСТЕМЫ ШИФРОВАНИЯ

Пусть требуется зашифровать следующее сообщение (открытый текст):

DEAR DAD SEND MORE MONEY AS SOON AS POSSIBLE TOM

«Дорогой папа. Как можно скорее пришли еще денег. Том.»

Один из способов шифрования простая замена, при которой каждая буква открытого текста заменяется на какую-то букву алфавита (возможно, на ту же самую). Для этого отправитель сообщения должен знать, на какую букву в шифротексте следует заменить каждую букву открытого текста. Часто это делается путем сведения нужных соответствий букв в виде двух алфавитов, например так, как показано ниже в таблице:

Алфавит Открытый ABCDEFGHIJKLMNOPQRSTUVWXYZ Шифровальный BLUESTOCKINGADFHJMPQRVWXYZ

Шифрограмма получается путем замены каждой буквы открытого текста на записанную непосредственно под ней букву шифровального алфавита.

ESBM EBE HGSBPS PSDE AFMS AFDSY BP PFFD BP HFPPKLGS QFA

Две алфавитные последовательности, используемые в процессе шифрования, называются, соответственно, открытым и шифровальным компонентом системы. Чтобы получатель шифрограммы мог восстановить открытый текст и прочитать сообщение, ему необходимо иметь копию вышеприведенной таблицы. Дешифровщик повторяет в обратном порядке все действия шифровальщика, раскрывая тем самым содержание сообщения.

В вышеприведенном примере использовался алгоритм побуквенной замены. Этот метод называется простой, или моноалфавитной заменой. Ключ к данному шифру состоит из таблицы, содержащей открытый и шифровальный алфавиты, в которой указывается, на какую букву в шифротексте следует

заменить букву открытого текста. В такой криптографической системе предполагается, что алгоритм шифрования общеизвестен, тогда как ключ доступен только отправителю и получателю соответствующих сообщений.

В открытом алфавите буквы расположены в их обычной последовательности; такой алфавит называется прямой стандартной последовательностью. Шифровальный же алфавит в нашем примере состоит из 26 букв латинского алфавита, определенным образом переставленных с использованием ключевого слова BLUESTOCKING (букв. «синий чулок»). После ключевого слова (ключевой фразы) ключ далее записывается с опущением всех тех букв, что уже появились в этом слове (или в первом слове этой фразы), а затем вписываются остающиеся буквы алфавита в обычном порядке, опять же с опущением всех ранее появившихся букв. Так, если в качестве ключевой мы используем фразу UNITED STATES OF AMERICA, то шифровальный алфавит будет выглядеть следующим образом:

U N I T E D S A O F M R C B G H J K L P Q V W X Y Z

С помощью ключевого слова (фразы) при шифровании можно перемешать любую алфавитную последовательность. Использование ключевых слов облегчает восстановление открытого и шифровального компонента системы, поскольку при этом необходимо запомнить только соответствующее ключевое слово (фразу). Нет необходимости записывать (или разгадывать) какие бы то ни было таблицы: если помнить ключевое слово, то алфавитную последовательность всегда можно восстановить по памяти.

В вышеприведенной шифрограмме между словами сохранены пробелы, однако шифровку можно сделать более защищенной (или, как говорят криптографы, устойчивой, или стойкой ко взлому; шифр считается тем более стойким, чем дольше он не поддается вскрытию) путем удаления межсловных пробелов из окончательного шифротекста. Согласно установившейся практике, шифротекст принято делить на группы из пяти букв каждая. (Когда-то

телеграфные компании при взимании платы каждую группу из пяти букв считали, как одно слово открытого текста.) Если убрать пробелы между словами, то нашу шифрограмму можно было бы записать так:

ESBME BEMGS BPSPS DEAFM SAFDS YBPPF FDBPH FPPKL GSQFA

Замена - одно из основных криптографических преобразований. Другим важнейшим шифровальным алгоритмом является перестановка. В шифре перестановки все буквы открытого текста остаются без изменений, но переставляются согласно заранее оговоренному правилу. Здесь также может использоваться ключ, управляющий процедурой шифрования. Так, используя в качестве ключа слово PANAMA, мы могли бы зашифровать вышеупомянутый открытый текст следующим образом:

Ключ P A N A M A Числовая последовательность6 1 5 2 4 3Блок перестановки D E A R D A D P L E A S E S E N D M O N E Y A S S O O N A S P O S S I B L E T O M Шифрограмма EPSNO OEREN YNSOA SMSSB DADAA IMALE EOSTD DEOSP L

В этом примере ключевое слово использовано для получения шифровальной числовой последовательности путем нумерации букв ключевого слова (относительно друг друга) в порядке их следования слева направо в стандартном алфавите. Далее под числовой последовательностью в строках, равных по длине ключевому слову, записан открытый текст. В процессе шифрования текст выписывается уже по отдельным столбцам в порядке, определяемом данной числовой последовательностью. Этот метод перестановки называется перестановкой столбцов, но можно избрать и другие «маршруты» перестановки, например выписывать шифротекст следуя по диагонали (слева направо или справа налево, или же чередуя левое и правое направления) или по спирали и т.п. Кроме того, буквы шифротекста могут записываться в виде различных геометрических фигур или любыми другими способами. Один из них состоит в двойном шифровании путем повторной

перестановки столбцов. При этом и в первом, и во втором блоках перестановки может быть использовано одно и то же ключевое слово, хотя лучше использовать разные ключевые слова. Такой шифр, называющийся двойной перестановкой, получил широкое распространение в 20 в.

Третьим основным алгоритмом шифрования является дробление. При этом каждой букве открытого текста сопоставляется более одного символа шифротекста, после чего символы перемешиваются (переставляются) в определенном порядке. Ниже приведена система, демонстрирующая процедуру дробления с использованием знаменитого шифра Bifid, авторство которого приписывается французскому криптографу Феликсу Мари Деластеллю.

Ключевая фраза UNITED STATES OF AMERICA 1 2 3 4 5 1 U N I T E Полибианский квадрат 2 D S A O F 3 M R C B G 4 H K L P Q 5 V W X Y Z Открытый текст DEARDADPLEASESENDMOREMONEY Соответствующая строка2123222441221211232313211 5Соответствующий столбец 1532131435325252114251425 4 Шифротекст DASOHSNUAAIDEERITGRWWUKVKY

Сначала составляется шифровальная таблица размером 5г5 (т.н. полибианский квадрат), куда построчно вписывается шифровальный алфавит с ключевой фразой UNITED STATES OF AMERICA; причем ради того, чтобы общее число букв алфавита не превышало 25, буква J опускается (поскольку эта буква, с одной стороны, малоупотребительна в английских текстах, а с другой - вполне может быть заменена буквой I, без какого-либо урона для смысла). В данной таблице букве A, например, соответствуют координаты 23, букве B - 34 и т.д.

Далее, в процессе шифрования под каждой буквой открытого текста в столбик записываются ее табличные координаты - номер строки и, ниже, номер столбца, а затем получившаяся цифровая последовательность переводится с помощью той же таблицы обратно в буквенную форму, но на этот раз она

читается уже в строчку, т.е. 21 - буква D, 23 - буква A, 22 - буква S и т.д. При таком шифровании координата строки и координата столбца каждой буквы оказываются разъединенными, что характерно именно для раздробляющего шифра.

Замена, перестановка и дробление представляют собой основные криптографические алгоритмы. Эти три базовых преобразования, зачастую в сочетании друг с другом, используются в большинстве систем шифрования для создания очень сложных шифровальных алгоритмов, особенно когда шифрование производится компьютером.

Код отличается от шифра тем, что подходит к преобразованию открытого текста скорее с лингвистической точки зрения, нежели с чисто формальной, как в вышеописанных примерах. (В повседневном употреблении термин «код» обыкновенно путают с термином «шифр».) Кодирование, как правило, включает в себя применение пространной таблицы или кодового словаря, где перечислены числовые соответствия (эквиваленты) не только для отдельных букв, но и для целых слов и наиболее употребительных фраз и предложений. Например, такая кодовая группа, как AABRT, вполне может соответствовать открытому тексту «как можно быстрее». Раньше в дипломатических и военных криптографических системах часто использовались весьма пространные коды. Однако с появлением компьютеров наибольшее распространение получили шифры.

3.ДРУГИЕ КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ

Криптография включает в себя и системы тайнописи, отличные от вышеописанных. Для сокрытия исходного сообщения могут использоваться симпатические (невидимые) чернила. Секретные сообщения могут содержаться в нотной партитуре в виде особой записи отдельных нот, а также мелодических и гармонических секвенций. В фотографии для сокрытия сообщения применяется метод микроскопических точек. Сообщение может быть также спрятано в рисунке, чертеже или картине.

4.КРИПТОГРАФИЯ — ИСТОРИЯ

4.1. РАННЯЯ ИСТОРИЯ КРИПТОГРАФИИ

Как только возникло письмо, появились и способы его шифрования. В древних цивилизациях мы находим два вида письма: иератическое, или священное письмо, использовавшееся священнослужителями для тайного общения друг с другом, и демотическое письмо, употреблявшееся всеми остальными. И у греков, и у римлян, и у других сопоставимых с ними по историческому значению народов были свои системы тайного письма. Изобретение первой системы скорописи, которая изначально замышлялась как секретное письмо, приписывается Туллиусу Тиро, вольноотпущенному рабу Цицерона (106-43 до н.э.).

Но только в период позднего Средневековья криптография превратилась в важный фактор дипломатических, торговых и военных сношений. По мере формирования европейских наций-государств возрастала нужда в средствах секретной коммуникации, особенно в сфере дипломатии. Со временем, когда отношения между новыми политическими образованиями установились, каждое государство учредило свою «тайную палату», в чьи функции входили перехват, досмотр и вскрытие шифров и кодов соседних государств. Зачастую это было сопряжено также с подделкой соответствующих документов и печатей. Так родилась наука криптоанализа, более известная под названием «взламывание шифров».

Первая книга по криптографии, называвшаяся Polygraphiae Libri Sex, была написана в 1499 Иоганном Трисемусом, аббатом бенедиктинского монастыря Св. Мартина в Шпанхейме (Германия). Хотя труд Трисемуса был

весь пронизан кабалистическими и оккультными аллюзиями, в этой рукописи были описаны многие шифры - как широко бытовавшие в то время, так и изобретенные самим автором. Эта работа была чрезвычайно популярна, первое латинское издание появилось в 1518, а в 1541 книга была переиздана на французском языке. Вскоре последовал и ее перевод на немецкий язык. Книга вызвала гнев многих монарших дворов Европы, полагавших, что Трисемус выдал в ней слишком много секретов, и автор едва избежал сожжения на костре.

В 1550 итальянец Джироламо Кардано опубликовал книгу De subtilitate libri xxi. Кардано изобрел шифр, называемый решеткой, или трафаретом, в котором секретное послание оказывалось сокрыто внутри более длинного и совершенно невинно выглядевшего открытого текста. Секретное сообщение можно было обнаружить, наложив на открытый текст лист пергаментной бумаги с прорезями (трафарет). Слова, появлявшиеся в прорезях, и составляли секретное послание.

Примерно в то же время о шифрах писал Джамбаттиста делла Порта. Труд Порта был скопирован и слегка переделан французским дипломатом Блезом де Виженером, труд которого Traité des chiffres приобрел широкую известность.

Для периода примерно с 16 по 19 вв. типичным был т.н. номенклатор - шифр, представлявший собой сочетание шифра замены и небольшого кода. Номенклатор обычно содержал кодовые эквиваленты букв алфавита и наиболее употребительных слогов, слов и словосочетаний, а также ряд специальных символов. Чаще всего в нем встречались специально созданные для этой цели символы, но нередко также использовалась астрологическая и оккультная символика.

Номенклатор был разработан как система шифрования, наилучшим образом приспособленная к наиболее употребительным в то время методам

криптоанализа, которые, как правило, включали подсчет частоты появления в тексте каждого шифрового символа и поиск в тексте слов и выражений, содержащих характерные для данного языка сочетания букв. Метод частотного анализа букв основан на том, что в любом языке одни буквы встречаются чаще, чем другие. В английском языке, например, чаще других встречается буква E. Другие наиболее часто встречающиеся буквы - T, A, O, N, R и S. (Целиком эту группу букв легко запомнить в виде слова «SENORITA».) А буквы J, K, X и Z встречаются в английском редко. Поскольку в ходе операции замены частота буквы не меняется, ключ к разгадке истинного значения того или иного шифросимвола заключается иногда в подсчете частоты его появления в шифротексте. Равным образом операция простой замены не вносит изменений и в сочетания букв (буквенные модели). Например, слово ATTACK, будучи зашифрованным, может принять вид XJJXBU. Несмотря на то, что значения букв при этом изменились, буквенная модель слова (а именно две одинаковых буквы, обрамленные слева и справа одной и той же буквой, после чего следуют две другие различные буквы) осталась неизменной.

Чтобы усложнить частотный анализ букв, каждую букву клера стали заменять двумя и более шифросимволами. Такая операция обычно разрушала и буквенные модели, от наличия которых в значительной мере зависел успех криптоанализа. Рассмотрим в качестве иллюстрации, т.н. «малый шифр», использовавшийся в наполеоновской армии для связи между войсками и генеральным штабом.

A-15 AR-25 AL-39M-114 MA-107B-37 BU-3 BO-35 BI-29N-115 NE-94 NI-116C-6 CA-32 CE-20O-90 OT-153D-23 DE-52P-137 PO-152E-53 ES-82 ET-50 EN-68Q-173 QUE-136F-55 FA-69 FE-58 FO-71R-169 RA-146 RE-126 RI-148G-81 GA-51S-167 SA-171 SE-177 SI-134H-85 HI-77SO-168 SU-174I-119 IS-122T-176 TI-145 TO-157J-87 JAI-123U-138K-?V-164 VE-132 VI-161 VO-175L-96 LU-103 LE-117W, X, Y-?LA-106Z-166

Приведенная выше таблица замены была дешифрована французским криптографом Этьеном Базери. В имевшихся в его распоряжении шифровках некоторые буквы клера (K, W, X и Y) не встречались, поэтому он не смог определить их шифровых эквивалентов.

«Малый шифр» содержит числовые эквиваленты для всех букв алфавита, а также для часто встречающихся биграмм (двухбуквенных сочетаний) и некоторых триграмм (трехбуквенных сочетаний). С помощью этого шифра, который специалисты называют силлабарием (слоговым алфавитом), слово NAPOLEON «Наполеон», например, может быть зашифровано по-разному:

NAPOLEON115151379096539011 5 или NAPOLEON1151515211790115

Использование подобных приемов сильно усложняет задачу криптоаналитика. Хотя наполеоновский шифр относительно невелик, в эпоху номенклаторов нередки были также криптосистемы, содержавшие сотни соответствий.

4.2. КРИПТОГРАФИЯ— НАШЕ ВРЕМЯ

С 1990-х годов страны начали в открытую формировать свои стандарты криптографических протоколов. США, к примеру, приняло в качестве стандарта для криптографии с закрытым ключом шифр Rijndael, более известный, как AES, в Европе приняли шифр NESSIE, в Японии — CRYPTREC.

Сами же алгоритмы сильно усложнились с точки зрения математических конструкций — чтобы затруднить криптоанализ (как линейный, так и на основе дифференциальных вычислений). Принципы выбора алгоритмов постепенно начали усложняться из-за новых требований, которые часто не связаны с математикой напрямую: например, из-за требований к устойчивости к атакам через сторонние каналы и посредством социальной инженерии.

«Также появляются совершенно новые направления в криптографии: на стыке математики и квантовых вычислений. Хотя у людей пока нет мощных квантовых компьютеров, уже разрабатываются способы взлома современных систем — в их числе алгоритм Шора. Но, используя технологии квантовых компьютеров, можно построить более надежные и совершенно новые механизмы передачи информации. Их разработка ведется с 1980-х годов», — рассказывает эксперт.

Квантовый алгоритм разложения числа на простые множители, который разработал Питер Шор. Алгоритм важен для задач квантовой криптографии: разложение натуральных чисел на множители используется при считывании банковских пластиковых карт и для других конфиденциальных операций. Нахождение способа дешифровки может обезоружить системы защиты

Компьютеризация и появление доступного интернета вывели криптографию в массы. Криптографические методы стали широко использовать бизнес и частные лица в электронных коммерческих операциях,

телекоммуникациях и других сферах. Это привело к появлению новой валюты, которую не контролируют государства, — биткойн. И технологии, без которой не смогли бы проходить транзакции в криптовалютах, — блокчейн.

Сейчас криптография используется повсеместно. Кроме очевидного направления — передачи информации — это цифровое телевидение, сотовая связь, wi-fi, считывание билетов в транспорте, защита электронной почты от спама, банковские операции. Одно из новейших направлений — маркировка товаров для борьбы с подделками.

Именно оно является одной из самых важных задач для развития российской экономики. Ведь в 2015 году доля нелегального товарооборота достигала 9% рынка, то есть, фактически каждый десятый товар был подделкой. В абсолютных цифрах их стоимость достигала 2,5 триллионов рублей. Среди самых подделываемых товаров оказались одежда, обувь и алкоголь. На рынке минеральных вод доля нелегальных товаров достигла 20-25% в среднем и до 60% по отдельным позициям, подделывается 25% молочной продукции, каждый пятый флакон духов и каждая десятая пачка сигарет.

В 2018 году выявлено в 1,6 раза больше контрафакта, чем в 2017-м. Для того, чтобы обелить рынок было принято решение о создании единой системы цифровой маркировки и прослеживания товаров «Честный ЗНАК», которую реализуют Ростех и «Центр развития перспективных технологий».

Задача такой системы лежит как раз в плоскости криптографии. На упаковках товаров печатают Data Matrix-коды, которые состоят из двух частей: кода идентификации и криптохвоста. Коды генерирует система цифровой маркировки, в них зашифрована уникальная информация о товаре. Код идентификации определяет позицию товара в системе и едином каталоге, криптохвосты дополнительно шифруют каждый код на производстве. Из-за того, что система маркировки не хранит коды целиком, подделать их

невозможно. Серию уникальных кодов нельзя и предсказать, а сам цифровой код не повторяется минимально в течение пяти лет с момента выпуска товарной позиции.

Одно из главных применений маркировки товаров. Покупатель может проверить, что товар в его руках — не подделка. Через приложение на смартфоне он считывает Data Matrix-код: если товар определился в системе, значит его произвели легально / ©Первый канал

Сейчас маркировка применяется в нескольких отраслях в добровольном режиме, обязательной же маркировке подлежат духи, шины, обувь, табак, фототехника и пять групп изделий легкой промышленности. С 1 января 2020 года стартует маркировка лекарств, в марте — молочной продукции.

Система маркировки позволяет отследить движение товара на протяжении всего жизненного цикла: выход с конвейера, отгрузка поставщику, передача в розничный магазин, включая проверку кода в магазине, когда товар попадает на полку. Это исключает возможность вбросить подделку в число легальных товаров / ©Честный ЗНАК

«Развитие компьютерных технологий революционно изменило принципы построения криптосистем. Криптография ушла далеко вперед от примитивных шифров к сложным алгоритмам шифрования. А появление мобильной связи, интернета и смартфонов вывело криптографию в гражданское поле.

5. КРИПТОГРАФИЯ: АЛГОРИТМЫ ШИФРОВАНИЯ

На сегодняшний день существует масса алгоритмов шифрования, имеющих значительную стойкость перед криптоанализом (криптографическую стойкость). Принято деление алгоритмов шифрования на три группы:

Симметричные алгоритмы

Ассиметричные алгоритмы

Алгоритмы хэш-функций

5.1. СИММЕТРИЧНЫЕ АЛГОРИТМЫ

Симметричное шифрование предусматривает использование одного и того же ключа и для зашифрования, и для расшифрования. К симметричным алгоритмам применяются два основных требования: полная утрата всех статистических закономерностей в объекте шифрования и отсутствие линейности.

Принято разделять симметричные системы на блочные и поточные.

Поточные шифры относятся к шифрам замены, преобразующим посимвольно открытый текст в шифрованный. Поточный алгоритм зашифровывает ?-й символ $x_ь$ из алфавита X открытого текста в ?-й символ $y_($ из алфавита У шифрованного текста с использованием меняющихся от такта к такту шифрующих отображений $ф_г$: X —> У, ? = 1,2,... .

Традиционно шифры замены строились по принципу поточного шифрования, и в качестве шифруемых символов использовались буквы или биграммы. В электронных поточных системах в качестве шифруемых символов

чаще всего фигурируют биты или байты. При использовании шифрования в высокоскоростных системах передачи данных наиболее высокой скоростью шифрования характеризуются аппаратно реализованные поточные шифры.

В отличие от поточных *блочные шифры* с использованием (как правило) неизменного отображения обрабатывают блоки информации, размер блока чаще всего составляет 64 или 128 битов. Блочный принцип присущ традиционным шифрам перестановки. Например, в маршрутных перестановках блок информации состоит из mn букв, записанных в таблицу размера m x n. Однако сформировалось понятие блочного шифра лишь в начале 1970-х гг. при разработке технологии защиты компьютерной информации.

Необходимость разработки нового класса блочных шифров была вызвана в основном двумя обстоятельствами: функциональной несовместимостью аппаратуры поточного шифрования и вычислительных машин, а также тем, что программы шифрования компьютерной информации, использующие посимвольную обработку данных, не могли достичь необходимых скоростей. Это стимулировало разработку нового класса шифров с блочным принципом обработки информации и с более высокой производительностью шифрования.

С учетом разрядности современных компьютерных процессоров наиболее удобны для программной реализации симметричные блочные шифры с размером n обрабатываемых блоков данных от 64 до 256 битов, где n кратно 32.

Обычно при симметричном шифровании используется сложная и многоступенчатая комбинация подстановок и перестановок исходных данных, причем ступеней (проходов) может быть множество, при этом каждой из них должен соответствовать «ключ прохода»

Операция подстановки выполняет первое требование, предъявляемое к симметричному шифру, избавляясь от любых статистических данных путем перемешивания битов сообщения по определенному заданному закону.

Перестановка необходима для выполнения второго требования – придания алгоритму нелинейности. Достигается это за счет замены определенной части сообщения заданного объема на стандартное значение путем обращения к исходному массиву.

Симметричные системы имеют как свои преимущества, так и недостатки перед асимметричными.

К преимуществам симметричных шифров относят высокую скорость шифрования, меньшую необходимую длину ключа при аналогичной стойкости, большую изученность и простоту реализации. Недостатками симметричных алгоритмов считают в первую очередь сложность обмена ключами ввиду большой вероятности нарушения секретности ключа при обмене, который необходим, и сложность управления ключами в большой сети.

Примеры симметричных шифров

ГОСТ 28147-89 — отечественный стандарт шифрования

3DES (Triple-DES, тройной DES)

RC6 (Шифр Ривеста)

Twofish

SEED — корейский стандарт шифрования

Camellia – японский стандарт шифрования

CAST (по инициалам разработчиков Carlisle Adams и Stafford Tavares)

IDEA

XTEA — наиболее простой в реализации алгоритм

AES – американский стандарт шифрования

DES – стандарт шифрования данных в США до AES

5.2. АСИММЕТРИЧНЫЕ АЛГОРИТМЫ

Ассиметричные системы также называют криптосистемами с открытым ключом. Это такой способ шифрования данных, при котором открытый ключ передается по открытому каналу (не скрывается) и используется для проверки электронной подписи и для шифрования данных. Для дешифровки же и создания электронной подписи используется второй ключ, секретный.

Само устройство асимметричных криптосистем использует идею односторонних функций $f(x)$, в которых несложно найти x, зная значение самой функции но почти невозможно найти саму $f(x)$, зная только значение x. Примером такой функции может служить телефонный справочник большого города, в котором легко найти номер человека, зная его фамилию и инициалы, но крайне сложно, зная номер, вычислить владельца.

Принцип работы асимметричных систем

Допустим, имеются два абонента: A и B, и абонент B хочет отправить шифрованное сообщение абоненту A. Он зашифровывает сообщение с помощью открытого ключа и передает его уже зашифрованным по открытому каналу связи. Получив сообщение, абонент A подвергает его расшифрованию с помощью секретного ключа и читает.

Здесь необходимо сделать уточнение. При получении сообщения абонент А должен аутентифицировать свою личность перед абонентом В для того, чтобы недоброжелатель не смог выдать себя за абонента А и подменить его открытый ключ своим.

Примеры асимметричных шрифтов

RSA (Rivest-Shamir-Adleman, Ривест — Шамир — Адлеман)

DSA (Digital Signature Algorithm)

Elgamal (Шифросистема Эль-Гамаля)

Diffie-Hellman (Обмен ключами Диффи — Хелмана)

ECC (Elliptic Curve Cryptography, криптография эллиптической кривой)

ГОСТ Р 34.10-2001

Rabin

Luc

McEliece

5.3. АЛГОРИТМЫ ХЭШ-ФУНКЦИЙ

Хешированием (от англ. hash) называется преобразование исходного информационного массива произвольной длины в битовую строку фиксированной длины.

Алгоритмов хеш-функций немало, а различаются они своими характеристиками – криптостойкостью, разрядностью, вычислительной сложностью и т.д.

Нас интересуют криптографически стойкие хеш-функции. К таким обычно предъявляют два требования:

Для заданного сообщения C практически невозможно подобрать другое сообщение C’ с таким же хешем

Практически невозможно подобрать пар сообщений (CC’), имеющих одинаковый хеш.

Требования называются стойкостью к коллизиям первого рода и второго рода соответственно.
Для таких функций остается важным и другое требование: при незначительном изменении аргумента должно происходить значительное изменение самой функции. Таким образом, значение хеша не должно давать информации даже об отдельных битах аргумента.

Примеры хеш-алгоритмов:

- Adler-32
- CRC
- SHA-1
- SHA-2 (SHA-224, SHA-256, SHA-384, SHA-512)
- HAVAL

- MD2
- MD4
- MD5
- N-Hash
- RIPEMD-160
- RIPEMD-256
- RIPEMD-320
- Skein
- Snefru
- Tiger (TTH)
- Whirlpool
- ГОСТ Р34.11-94 (ГОСТ 34.311-95)
- IP Internet Checksum (RFC 1071)

6.RC4

RC4 (от англ. Rivest cipher 4 или Ron's code), также известен как ARC4 или ARCFOUR (alleged RC4) — потоковый шифр, широко применяющийся в различных системах защиты информации в компьютерных сетях (например, в протоколах SSL и TLS, алгоритмах обеспечения безопасности беспроводных сетей WEP и WPA).

Шифр разработан компанией RSA Security, и для его использования требуется лицензия.

Алгоритм RC4, как и любой потоковый шифр, строится на основе генератора псевдослучайных битов. На вход генератора записывается ключ, а на выходе читаются псевдослучайные биты. Длина ключа может составлять от 40 до 2048 бит. Генерируемые биты имеют равномерное распределение.

Основные преимущества шифра:

- высокая скорость работы;
- переменный размер ключа.

RC4 довольно уязвим, если:

- используются не случайные или связанные ключи;
- один ключевой поток используется дважды.

Эти факторы, а также способ использования могут сделать криптосистему небезопасной (например, WEP).

6.1.История

Потоковый шифр RC4 был создан Рональдом Ривестом, сотрудником компании RSA Security, в 1987 году. Сокращение «RC4» официально обозначает «Rivest cipher 4» или «шифр Ривеста» («4» — номер версии; см. RC2, RC5, RC6; RC1 никогда не публиковался; RC3 разрабатывался, но в нём была найдена уязвимость), но его часто считают сокращением от «Ron's code» («код Рона»).

В течение семи лет шифр являлся коммерческой тайной, и точное описание алгоритма предоставлялось только после подписания соглашения о неразглашении, но в сентябре 1994 года его описание было анонимно отправлено в список рассылки (англ. mailing list) «Cypherpunks». Вскоре описание RC4 было опубликовано в группе новостей usenet «sci.crypt». Оттуда исходный код попал на множество сайтов в сети Интернет. Опубликованный алгоритм на выходе выдавал шифротексты, совпадающие с шифротекстами, выдаваемыми подлинным RC4. Обладатели легальных копий исходного кода RC4 подтвердили идентичность алгоритмов при различиях в обозначениях и структуре программы.

Поскольку данный алгоритм известен, он более не является коммерческой тайной. Однако, название «RC4» является торговой маркой компании RSA Security[en]. Чтобы избежать возможных претензий со стороны владельца торговой марки, шифр иногда называют «ARCFOUR» или «ARC4», имея в виду англ. alleged RC4 — «предполагаемый» RC4 (поскольку «RSA Security» официально не опубликовала алгоритм).

Алгоритм шифрования RC4 применяется в некоторых широко распространённых стандартах и протоколах шифрования (например, WEP, WPA, SSL и TLS).

RC4 стал популярен благодаря:

- простоте его аппаратной и программной реализации;
- высокой скорости работы алгоритма в обоих случаях.

В США длина ключа, рекомендуемая для использования внутри страны, равна 128 битам. Соглашение, заключённое между «SPA» (англ. software publishers association) и правительством США, разрешило экспортировать шифры RC4 с длиной ключа до 40 бит. 56-и битные ключи разрешено использовать заграничным отделениям американских компаний.

6.2.Описание алгоритма

Ядро алгоритма поточных шифров состоит из функции — генератора псевдослучайных битов (гаммы), который выдаёт поток битов ключа (ключевой поток, гамму, последовательность псевдослучайных битов).

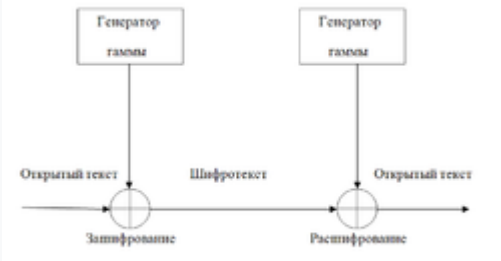

Рис.1 Режим гаммирования для поточных шифров

RC4 — фактически класс алгоритмов, определяемых размером блока (в дальнейшем S-блока). Параметр *n* является размером слова для алгоритма и определяет длину *S-блока*. Обычно, *n* = 8, но в целях анализа можно уменьшить его. Однако для повышения безопасности необходимо увеличить эту величину. В алгоритме нет противоречий на увеличение размера *S-блока* . При увеличении *n*, допустим, до 16 бит, элементов в *S-блоке* становится 65 536 и

соответственно время начальной итерации будет увеличено. Однако, скорость шифрования возрастёт.

Внутреннее состояние RC4 представляется в виде массива размером 2^n и двух счётчиков. Массив известен как *S-блок*, и далее будет обозначаться как S. Он всегда содержит перестановку 2^n возможных значений слова. Два счётчика обозначены через i и j.

Инициализация RC4 состоит из двух частей:

1. инициализация *S-блока*;
2. генерация псевдослучайного слова K.

Инициализация *S-блока*

Алгоритм также известен как «key-scheduling algorithm» или «KSA». Этот алгоритм использует ключ, подаваемый на вход пользователем, сохранённый в Key, и имеющий длину L байт. Инициализация начинается с заполнения массива S, далее этот массив перемешивается путём перестановок, определяемых ключом. Так как только одно действие выполняется над S, то должно выполняться утверждение, что S всегда содержит один набор значений, который был дан при первоначальной инициализации (*S[i] := i*).

```
for i from 0 to 255
   S[i] := i
endfor
j := 0
for i from 0 to 255
   j := ( j + S[i] + Key[ i mod L ] ) mod 256 // n = 8 ; 2^8 = 256
   поменять местами S[i] и S[j]
endfor
```

Генерация псевдослучайного слова *K*

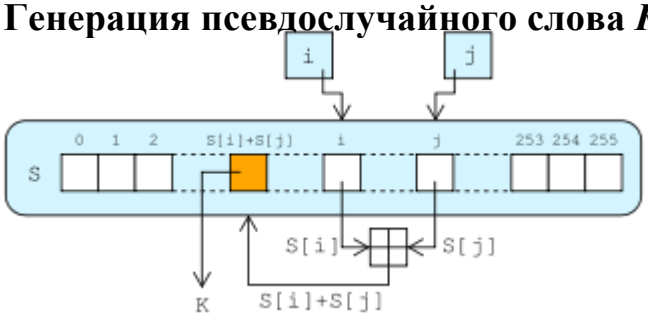

Рис.2 Генератор ключевого потока RC4

Эта часть алгоритма называется генератором псевдослучайной последовательности (англ. ***p**seudo-**r**andom **g**eneration **a**lgorithm*, *PRGA*). Генератор ключевого потока RC4 переставляет значения, хранящиеся в S. В одном цикле RC4 определяется одно *n*-битное слово K из ключевого потока. В дальнейшем ключевое слово будет сложено по модулю два с исходным текстом, которое пользователь хочет зашифровать, и получен зашифрованный текст.

```
i := 0
j := 0
while Цикл генерации:
   i := ( i + 1 ) mod 256
   j := ( j + S[i] ) mod 256
   поменять местами S[i] и S[j]
   t := ( S[i] + S[j] ) mod 256
   K := S[t]
   сгенерировано псевдослучайное слово K (для n = 8 будет сгенерирован один байт)
endwhile
```

6.3.Безопасность

В отличие от современных шифров (таких, как eSTREAM), RC4 не использует nonce (от англ. nonce — «number that can only be used once» — число, которое может быть использовано один раз) наряду с ключом. Это значит, что если один ключ должен использоваться в течение долгого времени для шифрования нескольких потоков, сама криптосистема, использующая RC4, должна комбинировать оказию и долгосрочный ключ для получения потокового ключа для RC4. Один из возможных выходов — генерировать **новый** ключ для RC4 с помощью хеш-функции от долгосрочного ключа и nonce. Однако многие приложения, использующие RC4, просто конкатенируют ключ и nonce. Из-за этого и слабого расписания ключей, используемого в RC4, приложение может стать уязвимым. Поэтому он был признан устаревшим многими софтверными компаниями, такими как Microsoft. Например, в .NET Framework от Microsoft отсутствует реализация RC4.

Здесь будут рассмотрены некоторые атаки на шифр и методы защиты от них.

Исследования Руза и восстановление ключа из перестановки

В 1995 году Андрю Руз (англ. *Andrew Roos*) экспериментально пронаблюдал, что первый байт ключевого потока коррелирован с первыми тремя байтами ключа, а первые несколько байт перестановки после алгоритма расписания ключей (англ. *KSA*) коррелированы с некоторой линейной комбинацией байт ключа. Эти смещения не были доказаны до 2007 года, когда Пол, Рафи и Мэйтрэ доказали коррелированность ключа и ключевого потока. Также Пол и Мэйтрэ доказали коррелированность перестановки и ключа. Последняя работа также использует коррелированность ключа и перестановки для того, чтобы создать первый алгоритм полного восстановления ключа из

последней перестановки после KSA, не делая предположений о ключе и векторе инициализации. Этот алгоритм имеет постоянную вероятность успеха в зависимости от времени, которая соответствует квадратному корню из сложности полного перебора. Позднее было сделано много работ о восстановлении ключа из внутреннего состояния RC4.

Атака Флурера, Мантина и Шамира (ФМШ)

В 2001 году Флурер, Мантин и Шамир опубликовали работу об уязвимости ключевого расписания RC4. Они показали, что первые байты ключевого потока среди всех возможных ключей неслучайны. Из этих байтов можно с высокой вероятностью получить информацию об используемом шифром ключе. И если долговременный ключ и nonce просто склеиваются для создания ключа шифра RC4, то этот долговременный ключ может быть получен с помощью анализа достаточно большого количества сообщений, зашифрованных с использованием данного ключа. Эта уязвимость и некоторые связанные с ней эффекты были использованы при взломе шифрования WEP в беспроводных сетях стандарта IEEE 802.11. Это показало необходимость скорейшей замены WEP, что повлекло за собой разработку нового стандарта безопасности беспроводных сетей WPA.

Криптосистему можно сделать невосприимчивой к этой атаке, если отбрасывать начало ключевого потока. Таким образом, модифицированный алгоритм называется «RC4-drop[n]», где n — количество байтов из начала ключевого потока, которые следует отбросить. Рекомендовано использовать $n = 768$, консервативная оценка составляет $n = 3072$

Атака базируется на слабости инициализационного вектора. Зная первое псевдослучайное слово K и m байтов входного ключа Key, используя

слабость в алгоритме генерации псевдо-случайного слова K, можно получить $m + 1$ байт входного ключа. Повторяя шаги добывается полный ключ. При атаке на WEP, для $n = 8$ IV имеет вид (B; 255; N), где B — от 3 до 8, а N любое число . Для определения около 60 вариантов *N* потребуется перехватить примерно 4 миллиона пакетов.

Атака Кляйна

В 2005 году Андреас Кляйн представил анализ шифра RC4, в котором он указал на сильную коррелированность ключа и ключевого потока RC4. Кляйн проанализировал атаки на первом раунде (подобные атаке ФМШ), на втором раунде и возможные их улучшения. Он также предложил некоторые изменения алгоритма для усиления стойкости шифра. В частности, он утверждает, что если поменять направление цикла на обратное в алгоритме ключевого расписания, то можно сделать шифр более стойким к атакам типа ФМШ.

Комбинаторная проблема

В 2001 году Ади Шамир и Ицхак Мантин первыми поставили комбинаторную проблему, связанную с количеством всевозможных входных и выходных данных шифра RC4. Если из всевозможных 256 элементов внутреннего состояния шифра известно x элементов из состояния ($x \leq 256$), то, если предположить, что остальные элементы нулевые, максимальное количество элементов, которые могут быть получены детерминированным алгоритмом за следующие 256 раундов, также равно x. В 2004 году это предположение было доказано Сорадюти Полом и Бартом Пренелем

Атака Ванхофа и Писсенса (2015)

Летом 2015 года Мэти Ванхоф (Mathy Vanhoef) и Франк Писсенс (Frank Piessens) из университета Левена в Бельгии продемонстрировали реальную атаку на протокол TLS, использующий RC4 для шифрования передаваемых данных. Идея взлома базируется на принципе MITM. Встроившись в канал передачи данных, атакующая сторона генерирует серверу большое количество запросов, вынуждая его в ответ возвращать куки, зашифрованные одним и тем же ключом. Имея в распоряжении около 9x227 ~ 230 пар {открытый текст, шифротекст}, атакующая сторона получила возможность на основе статистических методов Флюрер-Макгрю и ABSAB с вероятностью 94 % восстановить ключ и, следовательно, зашифрованные куки. Практические временные затраты составили около 52 часов, верхняя же оценка потребного времени на момент демонстрации составила около 72 часов.

6.4.Модификации RC4

Ранее рассматривались атаки, основанные на коррелируемости первых байт шифрованного текста и ключа. Подобные слабости алгоритма могут быть решены отбрасыванием начальной части шифрованного текста[16]. Надёжным считается отбрасывание первых 256, 512, 768 и 1024 байт. Исследования начала шифротекста были проведены для показания ненадёжности определённого числа первых байтов, что может привести к получению злоумышленником ключа шифрования. Были предложены несколько модификаций RC4 выполняющие поставленную задачу усиления безопасности при использовании алгоритма: RC4A, VMPC, RC4+.

RC4A

В 2004 году свет увидела работа Souradyuti Paul и Bart Preneel, в которой предлагалась модификация RC4A.

Для RC4A используется два *S-блока* вместо одного, как в RC4, обозначим S_1 и S_2. Для них соответствующе используются два счётчика j_1, j_2. Счётчик i, как и для RC4, используется в единственном числе для всего алгоритма. Принцип выполнения алгоритма остается прежним, но имеется ряд отличий:

S_1 является параметром для S_2.

За одну итерацию, то есть за одно увеличение индекса i, генерируется два байта шифротекста.

Алгоритм :

```
i := 0
j₁ := 0
j₂ := 0
while Цикл генерации:
   i := i + 1
   j₁ := ( j₁ + S₁[i] ) mod 256
   поменять местами S₁[i] и S₁[j₁]
   I₂ := ( S₁[i] + S₁[j₁] ) mod 256
   output := S₂[I₂]
   j₂ = ( j₂ + S₂[i] ) mod 256
   поменять местами S₂[i] и S₂[j₂]
   I₁ = ( S₂[i] + S₂[j₂] ) mod 256
   output := S₁[I₁]
endwhile
```

Скорость шифрования данного алгоритма может быть увеличена за счёт распараллеливания.

RC4+

В 2008 году была разработана и предложена модификация RC4+. Авторы Subhamoy Maitra и Goutam Paul модифицировали инициализацию *S-блока*(KSA+), использовав 3-уровневое скремблирование. Также модификации был подвергнут алгоритм генерации псевдослучайного слова (PRGA+).

Алгоритм:

Все арифметические операции выполняются по mod 256. Символами «<<» и «>>» обозначены битовые сдвиги влево и вправо соответственно. Символ «⊕» обозначает операцию «исключающее ИЛИ»

```
while Цикл генерации:
   i := i + 1
   a := S[i]
   j := j + a
   b := S[j]
   S[i] := b     (поменяли местами S[i] и S[j])
   S[j] := a
   c := S[ i<<5 ⊕ j>>3 ] + S[ j<<5 ⊕ i>>3 ]
   output ( S[a+b] + S[c⊕0xAA] ) ⊕ S[ j+b ]
endwhile
```

6.5.Реализация

Работа многих поточных шифров основана на линейных регистрах сдвига с обратной связью (англ. LFSR). Это позволяет достичь высокой эффективности реализаций шифра в виде интегральной схемы (аппаратная реализация), но затрудняет программную реализацию таких шифров. Поскольку шифр RC4 не использует LFSR и основан на байтовых операциях, его удобно реализовывать программно. Типичная реализация выполняет от 8 до 16 машинных команд на каждый байт текста, поэтому программная реализация шифра должна работать быстро.

7.ССЫЛКИ И ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

- https://studme.org/294609/informatika/potochnye_shifry
- https://gitjournal.tech/kriptografija-chto-jeto-takoe-i-zony-primenenija/
- https://www.krugosvet.ru/enc/lingvistika/kriptografiya

8. Приложение

main.c Открыть в приложении «Xcode»

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define swap(i, j) typeof(i) swp = i; i = j; j = swp;


void RC4(const unsigned char *data, long dataLen, const unsigned char *key, long keyLen, unsigned char *result)
{
    unsigned char T[256];
    unsigned char S[256];
    int j = 0, t, i;
    for (i= 0; i < 256; i++) {
        S[i] = i;
        T[i] = key[i % keyLen];
    }
    for (i= 0; i < 256; i++) {
        j = (j + S[i] + T[i]) % 256;
        swap(S[i], S[j])
    }
    j = 0;
    for (int x = 0; x < dataLen; x++) {
        i = (i + 1) % 256;
        j = (j + S[i]) % 256;
        swap(S[i], S[j])
        t = (S[i] + S[j]) % 256;
        result[x] = data[x] ^ S[t];
    }
}

unsigned char *readFile(FILE *f, long *fsize) {
    unsigned char *data;
    fseek(f, 0, 2);
    long x = ftell(f);
    *fsize = x;
    fseek(f, 0, 0);
    data = (unsigned char *) calloc(*fsize + 10, sizeof(unsigned char));
    memset (data, 0, *fsize + 10);
    fread(data, 1, *fsize, f);
    fclose(f);
    return data;
}

void Encryptor(const char *sourceFile, const char *keyFile, const char *resultFile) {
    unsigned char *data;
    unsigned char *Cypher;
    long fsize, ksize;
    FILE *f = fopen(sourceFile, "rb");
    data = readFile(f, &fsize);
    Cypher = (unsigned char *) calloc(fsize + 10, sizeof(unsigned char));
    memset (Cypher, 0, fsize + 10);
    unsigned char *key;
    f = fopen(keyFile, "rb");
    key = readFile(f, &ksize);
    if(ksize > 256)
    {
        printf("Ключ не может быть больше 256 байт");
        exit(0);
    }
    printf("\n Ключ: %s:\n%s\n", keyFile, key);
    RC4(data, fsize - 1, key, ksize, Cypher);
    printf("\n\n Шифрованный текст тут: %s: \n",resultFile);
    printf("\n\n");
    f = fopen(resultFile, "w");
    fwrite(Cypher, 1, fsize, f);
    fclose(f);
    free(data);
    free(Cypher);
    free(key);
}

int main(int argc, const char *argv[]) {;
    const char *sourceFile = argv[1];
    const char *keyFile = argv[2];
    const char *resultFile = argv[3];

    if (argc < 3) {
        printf("Для шифрования укажите файл с исходным текстом. Так же нужен путь к файлу,где записан ключ и
будет зашифрованный текст. Пример ввода аргументов:\n");
        printf("\t ./Encrypt ~/Desktop/plainText.txt ~/Desktop/key.txt ~/Desktop/Cipher.txt\n");
        exit(0);
    }
    Encryptor(sourceFile, keyFile, resultFile);
    printf("Завершено! Зашифрованный текст можете посмотреть в файле который вы указали при запуске программы");
    exit(1);
}
```

9. Пример выполнения кода.

В качестве исходного шифруемого текста возьмем один абзац из данной работы. ***Исходный текст:***

«В отличие от современных шифров (таких, как eSTREAM), RC4 не использует nonce (от англ. nonce — «number that can only be used once» — число, которое может быть использовано один раз) наряду с ключом. Это значит, что если один ключ должен использоваться в течение долгого времени для шифрования нескольких потоков, сама криптосистема, использующая RC4, должна комбинировать оказию и долгосрочный ключ для получения потокового ключа для RC4. Один из возможных выходов — генерировать **новый** ключ для RC4 с помощью хеш-функции от долгосрочного ключа и nonce. Однако многие приложения, использующие RC4, просто конкатенируют ключ и nonce. Из-за этого и слабого расписания ключей, используемого в RC4, приложение может стать уязвимым. Поэтому он был признан устаревшим многими софтверными компаниями, такими как Microsoft. Например, в .NET Framework от Microsoft отсутствует реализация RC4.»

Ключ:

«MIReaInSTitUtEOFCybeRNETICS100501cOMPUTERsECUrItYCcSo0520mIrEA InStItUTEOfCYbERNETiCS100501cOMPutERsecuriTYCCso0520mIReAinStituTe oFCybeRNeTicS100501COmpUtersECUrItyCCso0520»

Зашифрованный текст:

```
05 EC 01 07 6D 49 38 62 84 6E 5B 5D 89 8B 4C D9 1A E4 35 97 11 CB B8 37
AB 40 58 36 FE 0E DD E2 17 B6 C6 D5 8B 29 F1 90 21 8E AE F1 D6 81 AC 82
F1 AF C4 A8 AE 9E 2C BE 28 22 EF C9 E1 17 14 00 B8 34 03 73 67 BE 80 92
15 8A 8E 42 A3 12 0A BC 03 1C 54 05 4E EC 22 AA BC C1 D1 94 53 8B 14 CA
42 EC 0B 4F 45 2D 46 21 75 9E 1F 7E 3E FE 13 C0 60 9D D5 5D CE 30 B9 FF
76 6F 70 0C 6A 39 E7 97 BF 61 C6 BA E3 B1 26 ED 8F F3 1F F1 7B 5A 07 B5
6A CA FC 7F F2 CA 8F 10 84 64 B7 1C 0E F0 7E 58 72 D8 4A 4D 3F D1 67 1B
38 46 46 E0 15 4E 1E D2 63 E8 63 80 BD 89 34 B5 11 19 FF 1D 5B 9B A5 40
DA FF 3F FB F7 4A 48 27 F4 02 D0 EE 53 90 A0 44 1F C6 42 36 4F C6 4D B3
FF 59 A9 4A B6 98 B1 4F 0B 09 DC 20 8B 96 49 1E 78 71 41 76 CB 59 89 6B
8B C2 B4 B4 38 A4 BF BA AD 67 0E 62 23 2E DC 67 BA B8 17 A1 09 62 CA C1
85 FA 21 D2 3B D0 90 29 E9 FB F8 00 7B FF 0C AC D3 E0 90 93 60 16 BB 3E
```

```
89 46 B0 97 AA 1B 0A E3 B5 19 F2 51 5D 89 86 EA 7A 45 45 38 0B 07 53 86
77 7D 15 12 85 95 ED FC 94 18 E4 E3 DC 76 1F 76 22 4F 83 B8 76 5A AD 07
EF FC 4B 5E 73 A7 4B E3 B2 AD 6D FA A2 38 BC 38 70 2A 07 12 50 4A 58 56
5F 4E B4 CD 5E CE B4 CD 86 CB A4 39 6E FD 80 C4 72 D1 EC AC 51 99 72 40
B8 3D 54 BF D5 74 A8 32 5A 28 F1 83 37 23 B8 57 F7 EE 9B F0 98 C6 9F 3D
0A A0 63 17 1C 25 70 F0 51 B7 19 10 BE F4 B4 81 03 9D 59 61 54 F4 6A A1
74 6F 33 C1 32 36 CC 5E 62 6A 7B 18 05 7C 1F B7 77 7C E8 59 9B AE 99 BA
57 12 E9 1D 93 E6 0B 81 BD A1 28 00 01 69 37 C4 B4 D6 84 4B 52 A4 3C BB
B2 97 8F 6A E2 F1 6D C5 1A 5A 37 E3 9C 00 45 C9 46 FF 9E EB 7C 59 93 75
7B 81 BF 01 10 2F 49 AF 2D 37 3C 8B 8B DF 73 73 FC 00 66 FC DD 57 CE D8
0A FB 69 86 58 58 BB 27 58 5C 46 F2 E7 2A E0 EF DC 21 68 50 B0 3C DC 27
63 5F 6F 55 7C 3A 7A B3 1D 08 A9 05 9D CB B4 E2 F7 B2 04 64 EB BA BF 66
71 BD 9E F2 45 C1 D9 75 34 67 DA 24 A0 CD 17 55 24 41 D2 64 FC FB 86 03
F1 7F 85 AC 99 EB 67 2F 52 C6 C8 0F F1 A5 B7 3C 8C B8 7C 13 20 4D 1B 36
0E 69 A9 2A CE 09 4F 29 9F 15 04 2A 0D 5C 16 41 97 CA 9C 31 CE 1C 9F 98
C0 44 BB FB E6 57 21 62 FB CC 92 D7 E5 00 09 B8 8E 6C 37 87 33 E9 0B F8
90 E4 7B B9 57 5A C3 A2 4A 2F D5 86 E7 22 50 00 91 07 F1 CC B7 60 F5 3F
E0 89 1D 06 52 79 C8 27 F5 6B FC 02 B5 48 16 BF 06 B0 54 16 E0 E6 F5 1F
E4 90 94 D1 8A 12 27 26 7C 55 E0 83 3A 0C 84 A6 0A 61 86 F8 9A 66 9D 72
E1 CB A9 F8 69 DD EF FB 49 9E D3 C2 32 33 36 7F D7 D2 0E 59 AC CC D9 47
97 B9 17 B6 BA 16 4A 19 DE 2D A6 8A AB 86 AE 10 B6 29 AC 08 09 03 23 29
92 38 D6 6D 7D 84 B7 E8 B0 66 53 B9 33 54 2D 94 3F E2 49 F7 FC 01 B3 B0
D5 AE E9 07 29 AC CB 2D 54 AB 6A 20 F3 78 DC B1 B8 30 71 E8 76 C8 CE DC
69 56 5C 0A 81 7A 19 E4 85 27 BA A6 2F 54 D0 F3 79 89 CF ED 6A 80 B6 3F
6D 31 82 8D 31 35 ED 61 F8 43 43 C1 0C 00 A4 CF 1A 25 8E 4F D4 BD B4 28
E7 FD 33 CB 0E 32 83 2A BF 2B 39 D7 9C C5 D9 F2 A1 32 1C 56 EE 1B B2 AE
22 74 AA 3B C6 4C 5E A7 FA 14 3D A8 28 3A 7E B3 6A 06 30 02 A0 EB 48 72
82 3D 41 11 12 F1 08 58 61 E9 2E 6A F6 D7 27 46 69 C4 64 A6 C7 5B D9 7D
92 0F 24 57 4E BA E5 DE D8 82 05 00 41 E9 DB 18 FC 97 9B DF 2B 2F 5C 39
0E E6 96 6E 75 D1 11 94 56 F1 04 F0 D0 08 14 26 55 A9 2B FA 98 B3 D3 ED
06 85 39 3C 66 28 09 DD F8 93 33 39 BB E7 DD 83 52 86 E3 9B 60 5F A8 E8
D3 CA 8D F4 A7 7A 12 04 D8 5A 7A 7E 90 AA C0 85 7F 26 9A 6A 4C D2 87 EF
F2 32 49 DD C9 FC EE 2A 4F D9 73 01 7C 8B EF 77 9F 2E 83 68 9A A5 B4 2E
E5 8F D2 EC 96 30 F5 3E C4 62 AE DA 97 5F CE 0A 07 C8 6B 43 9B 92 30 BE
86 86 6C E4 72 E9 76 7D 3A AE 74 44 E2 32 85 D3 F1 99 7C 68 D3 11 FE 18
E0 D4 9C 83 2A 68 3D 3E D7 9A D8 6D 72 4E 02 19 B8 EC 09 16 3A F4 37 B2
A3 8F 10 3D F8 FC BF 98 64 92 F4 AB 81 EE 1C 51 81 04 3C 0D 3E 79 8F BE
7B C1 41 59 E7 0B 67 66 E1 50 FC 65 04 64 FC 29 79 AA A5 D2 9C 24 CF 65
74 78 A3 D8 F0 CA B7 59 9B CF 7A 0D 3D 37 0E E5 BA 72 AF CB C1 87 6C 03
BE 30 DC 19 7B EC F3 A1 B1 72 CA 86 68 5B CF 2A 1E 9B E6 C4 28 71 41 BA
0C 76 7B EA CA A4 34 C3 12 A9 41 B7 ED 61 BE F0 2B 47 1F 19 4D 51 2A 7F
8C 28 FC EF 8C F0 DC 8C 57 46 95 F8 A7 16 55 54 EC 0D C1 55 EF B7 9A 4D
E9 A6 69 D8 26 90 21 08 C1 73 11 C2 DC B8 31 D1 82 29 4C 33 E4 64 9E E8
5B 1B FC 24 32 3A 54 CA C0 64 E3 BC 6A 39 78 5B BF F7 32 A1 08 FD 17 67
2A A7 51 B8 AC 33 8B 41 DF B9 55 16 B3 35 07 0E 42 CF 59 04 FB 87 1C 28
32 94 CC 13 38 A3 F8 86 E7 90 96 FC 07 CE 32 34 CA D4 56 0D 85 2C 88 19
84 7B F6 3D 85 36 95 ED 78 7F BE 55 5F 79 08 6C CE 65 94 0A EE 8E 39 6B
E6 E5 11 5B E8 F5 D8 B7 A6 A8 AF 87 91 7B A6 42 B6 0B 0F A5 ED 80 A6 3A
E0 35 1F 24 D6 5D F9 90 EB 75 DB 0B 30 50 28 BB 92 62 0D 18 47 BD A6 10
0A 62 99 D6 8A E8 3E FB 7E D4 92 E9 54 0B 12 3E 9A 74 21 F4 99 74 9C 98
FA DA A7 22 A4 D1 7D F8 1F CC 7D 3A 87 3E 7F 22 E1 F4 E7 0E 13
```