Оглавление

[Криптографический алгоритм 2](#_Toc485170101)

[Симметричные криптосистемы 3](#_Toc485170102)

[Поточный шифр 4](#_Toc485170103)

[Синхронные поточные шифры. 5](#_Toc485170104)

[Криптографическая стойкость 7](#_Toc485170105)

[Абсолютно стойкая система 8](#_Toc485170106)

[Оценка криптостойкости систем шифрования 9](#_Toc485170107)

[Шифр Вернама 10](#_Toc485170108)

[Криптографическая стойкость шифра Вернама 11](#_Toc485170109)

[Недостатки шифра Вернама 12](#_Toc485170110)

[Шеннон 13](#_Toc485170111)

[Обоснование выбора языка 14](#_Toc485170112)

[Статистические тесты для поточных шифров 15](#_Toc485170113)

[Список литературы 16](#_Toc485170114)

Введение

Передача сообщения между людьми была всегда очень важной задачей. Но при перехвате сообщения его мог узнать третий человек. Для защиты сообщений с давних времен придумывались шифры. Но с развитием технологий, взлом шифров становится всё проще и безопасность сообщения становится под вопросом.

С появлением ЭВМ методы шифрования так же стали сложнее. Основной сложностью стала передача закрытого ключа от одного человека к другому. Каждый раз встречаться и передавать листочек с закрытым ключом неудобно. Если передавать закрытый ключ по каналу связи, то его может перехватить третье лицо.

Вернам предложил систему, которую невозможно взломать в реальных условиях. Но в системе встречается множество недостатков из-за которых на практике невозможно использовать шифр для передачи множества сообщений. Потому что если использовать один и тот же закрытый ключ, то третье лицо сможет выявить закономерность и восстановить закрытый ключ.

Шеннон предложил систему, которая поможет решить данные недостатки и позволит открыто передавать необходимые данные для генерации закрытого ключа. Но Шеннон не стал доказывать свою теорию. В 2016 году Рябко Б.Я. в своей работе [6] доказал данный метод шифрования.

В данной работе будет разработана реализация клиент-серверного приложения, благодаря которому клиенты смогут обмениваться сообщениями, зашифрованными методом Шеннона. Так же будут проведены статистические тесты, для выявления слабых точек реализации и указаны важные аспекты реализации данной криптосистемы.

Криптографический алгоритм

Криптография — наука, занимающаяся методами преобразования информации, не позволяющим противнику извлечь данные из перехватываемых им сообщений. Криптографический алгоритм — система преобразований входной информации в шифртекст с целью обеспечить секретность передаваемых данных. Неотъемлемой частью и главной особенностью любого алгоритма шифрования является использование ключа шифрования, при помощи которого осуществляется конкретное преобразование из совокупности возможных вариантов для данного шифра. [1] Основная идея шифрования заключается в том, что злоумышленник, которому удалось перехватить передаваемую информацию, имея зашифрованные данные и не имея к ним ключа, не будет иметь возможности ни прочесть, ни изменить перехваченную информацию.

Всё разнообразие существующих на настоящий момент шифров можно разделить по двум крупным классификациям. Первая классификация разделяет алгоритмы шифрования на две категории в зависимости от структуры используемых ключей:

* Симметричный метод, использующий один и тот же ключ, как для шифрования, так и для дешифрования
* Ассиметричный метод, использующий для тех же целей два различных ключа.

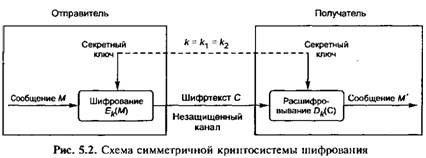
В зависимости от объема шифруемых данных, а также способа их обработки, шифры подразделяются на:

* Блоковый шифр, шифрующий блоки информации фиксированной длины, блок информации обрабатывается целиком
* Поточный шифр, шифрующий информацию по мере поступления, обрабатывает биты открытого и закрытого текста в режиме, приближенном к реальному времени.

В данной работе были реализованы исследования над двумя алгоритмами шифрования данных, которые относятся к поточным симметричным шифрам, поэтому особое внимание уделяется именно этой категории криптографических алгоритмов.

Симметричные криптосистемы

В симметричных криптосистемах шифрование и расшифровывание исходного сообщения происходит одним и тем же ключом. Любой, кто имеет доступ к ключу, может расшифровать исходное сообщение. Для такой системы все ключи должны держаться в секрете. Отсюда следует что ключ должен быть доступен только тем, кто должен получить сообщение. Схема симметричной криптосистемы шифрования:



Данную криптосистему можно охарактеризовать, как систему с высокой скоростью шифрования, обеспечивающую секретность, подлинность и целостность исходного сообщения. Секретность передачи данных зависит от надежности шифра и от того, насколько хорошо хранится закрытый ключ.

Закрытый ключ представляет из себя файл либо массив с данными, которые хранятся на персональном носителе, что позволяет иметь доступ к закрытому ключу только владельцу. Подлинность передаваемого сообщения достигается благодаря тому, что невозможно модифицировать передаваемое сообщение без знания закрытого ключа для шифрования. Модифицированное сообщение не может быть зашифровано без закрытого ключа.

Целостность передаваемых данных обеспечивается благодаря вставке контрольной суммы исходного сообщения. Получатель после расшифровывания сможет проверить целостность сообщения.

При подготовке к шифрованию необходимо передать закрытый ключ между отправителем и получателем. От надежности передачи закрытого ключа, зависит и безопасность передачи шифртекста. На каждую пару пользователей должна приходиться уникальная пара ключей.

Поточный шифр

Поточный шифр — это симметричный шифр, в котором каждый символ текста преобразуется в символ шифрованного текста в зависимости не только от используемого ключа, но и от его расположения в потоке открытого текста.[\*\*\*\*]

Простейший пример поточного шифра:

1. Генерируется гамма-последовательность K.
2. Исходный текст M.
3. К гамма-последовательности и исходному тексту применяется битовая операция XOR и получается шифртекст C.
4. Расшифровка происходит операцией XOR между гамма-последовательностью и шифртекстом получая исходный текст M.

Отсюда следует, что если гамма-последовательность не имеет периода и выбирается случайно, то взломать шифр невозможно.[\*\*\*]

Но поточный шифр имеет и свои минусы – из-за того, что закрытые ключи одинаковой длины с исходным сообщением в реальных системах не встречаются, на практике применяются ключи меньшей длины. С помощью него генерируется псевдослучайная гаммирующая последовательность. В будущем такое свойство гамма-последовательности, как псевдослучайность, может быть использовано при совершении атаки на шифр.

Если при генерации гамма-последовательности произойдет искажение одного любого бита, то при передаче по каналу связи и расшифровывании будет искажен всего лишь один бит, а остальная часть будет расшифрована правильно. Но если произойдет потеря 1 бита при передаче по каналу связи, то расшифровать сообщение дальше потерянного бита будет невозможно.

Практически во всех каналах передачи данных присутствуют помехи. Во избежание потери информации решают проблему синхронизации шифрования и расшифрования текста – применяют синхронные поточные шифры.

Синхронные поточные шифры.

Синхронные поточные шифры – такие шифры в которых поток ключей генерируется независимо от исходного текста и шифртекста.[\*\*\*]

Во время шифрования гамма-последовательность выдает биты для шифрования исходного сообщения, которые одинаковы с битами для расшифровывания. Потеря знака шифртекста приведет к нарушению синхронизации между двумя гамма-последовательностями и невозможность расшифровать сообщение. В этой ситуации отправитель и получатель должны синхронизироваться. Синхронизация происходит вставкой в передаваемый шифртекст специальных маркировок. Благодаря этому не правильное расшифровывание передаваемого шифртекста происходит лишь до того момента, пока на стороне получателя не будет принят один из маркеров. Для выполнения синхронизации ни одна часть закрытого ключа не должна повторяться.

Так как генерируется один и тот же поток ключей, как для шифрования, так и для дешифрования, следовательно, вывод ключевых элементов должен быть предопределен. Если данная система реализуется на конечном автомате, то рано или поздно последовательность придет к повторению, такая генерация потока называется периодической. Все генераторы ключевого потока являются периодическими, за исключением одноразовых блокнотов.

Плюсы синхронного поточного шифра:

1. Отсутствие распространения ошибок – только искаженные биты будут расшифрованы неправильно, все предшествующие и последующие биты не будут подвержены изменениям.
2. Защищает от подмены шифртекста данный факт будет заметен, так как действие подмены приведет к потере синхронизации.

Минусы синхронного поточного шифра:

1. Если злоумышленник знает исходный текст, то он сможет подменой битов изменить данные при расшифровке шифртекста.

Однако, синхронные потоковые шифры имею уязвимость к атакам на основе изменения отдельных бит данных шифротекста. Злоумышленник может изменить эти отдельные биты так, что шифротекст расшифруется так, как ему это будет выгодно.

Согласно Райнеру Рюппелю можно выделить четыре основных подхода к проектированию поточных шифров[\*\*\*\*]:

1. Cистемно-теоретический подход - основан на создании для криптоаналитика сложной, ранее неисследованной проблемы.
2. Cложно-теоретический подход - основан на сложной, но известной проблеме (например, факторизация чисел или дискретное логарифмирование).
3. Информационно-технический подход - основан на попытке утаить открытый текст от криптоаналитика – вне зависимости от того сколько времени потрачено на дешифрование, криптоаналитик не найдёт однозначного решения.
4. Рандомизированный подход - основан на создании объёмной задачи; криптограф тем самым пытается сделать решение задачи расшифрования физически невозможной.   
   Например: криптограф может зашифровать некоторую статью, а ключом будут указания на то, какие части статьи были использованы при шифровании. Криптоаналитику придётся перебрать все случайные комбинации частей статьи, прежде чем ему повезёт, и он определит ключ.

Соответственно этим подходам также были указаны и теоретические критерии Райнера Рюппеля для проектирования поточных систем:

1. Длинные периоды выходных последовательностей
2. Большая линейная сложность
3. Диффузия – рассеивание избыточности в подструктурах
4. Каждый бит потока ключей должен быть сложным преобразованием большинства битов ключа
5. Критерий нелинейности для логических функций.

Если атакующий обладает неограниченными вычислительными ресурсами и временем, то единственной реализуемой криптосистемой, защищенной от такого воздействия является одноразовый блокнот.

Криптографическая стойкость

Криптографическая стойкость — одно из центральных понятий криптографии и наиболее значимый параметр, характеризующий алгоритмы шифрования. Стойкостью шифра называют его способность противостоять возможным направленным на него атакам. Криптостойкость зашифрованной информации напрямую зависит от возможности попыток несанкционированного чтения данных.

Криптоанализ — наука, позволяющая найти относительную стойкость алгоритма, под данным понятием также понимается попытка найти уязвимость в алгоритме, выяснить ключ или вскрыть шифр. Абсолютно стойким алгоритмом считается такой алгоритм, на который от атакующего потребуются недостижимые на практике вычислительные ресурсы либо колоссальное количество времени для расшифровки перехваченного сообщения, такое, что по его истечению сообщение потеряет ценность. Часто можно только доказать уязвимость только алгоритма.

К основным видам атакам, направленных на вскрытие поточных шифров, можно отнести такие, как:

* 1. Силовые – основанные на полном переборе всех возможных вариантов. Сложность этого вида атаки зависит от таких параметров, как размер ключей, открытого текста и т.д.
  2. Статистические – направленные на изучение выходной последовательности данных, статистических свойств шифрующей системы. А также метод анализа сложности последовательности – способ направлен на поиск метода генерировать идентичную гамма-последовательность, но более простым методом.
  3. Аналитические – методы применяются при известном описании генератора, открытого и закрытого текста. Целью данной атаки является определить начальный ключ.

В зависимости от криптостойкости алгоритмов они делятся на абсолютно стойкие и достаточно стойкие системы. Шифры, исследуемые в данной работе, относятся к категории абсолютно стойких систем, поэтому более подробно рассмотрим свойства алгоритмов этой категории.

Абсолютно стойкая система

Абсолютно стойкой или обладающей теоретической стойкостью называется такая система, уязвимость которой невозможно доказать ни теоретически, ни практически, даже если у атакующего имеются бесконечно большие вычислительные ресурсы.

Доказательство существования данных систем шифрования, а также необходимые и достаточные условия для совершенной секретности, были определены Клодом Шенноном в работе “Теория связи в секретных системах”. [2]

Приведенные ниже параметры Шеннон определил, как необходимые для создания абсолютно стойких систем:

1. Закрытый ключ генерируется уникально для каждого передаваемого сообщения.

Вероятность появления каждого нового символа закрытого ключа равна и символы последовательно независимы и случайны.

Длина передаваемого сообщения и длина закрытого ключа должны совпадать или закрытый ключ должен быть длиннее передаваемого сообщения.

Передаваемое сообщение имеет некоторую избыточность, для того чтобы оценить правильность расшифровки шифртекста.

А также обозначил недостаток, имеющийся у совершенно секретных систем, который проявляется в том, что для передачи большого объема данных требуется посылать ключ шифрования эквивалентного с сообщением объема.

Шеннон доказал, что примером абсолютно стойкой системы является шифр Вернама, иначе называемый одноразовый блокнот. Верное использование данного шифра не позволит получить злоумышленнику какую-либо информацию из зашифрованной информации из-за того, что любой бит перехваченного сообщения можно лишь угадать с вероятностью 1/2.

На данный момент практическое применение криптосистем, удовлетворяющих требованиям абсолютной стойкости, ограничено из соображений стоимости и удобства пользования.

Оценка криптостойкости систем шифрования

Все современные шифры построены на принципах Кирхгофа, которые говорят о том, что секретность шифра обеспечивается не секретностью алгоритма, а секретностью ключа. [3] Анализ надежности криптосистемы всегда должен производиться исходя из того, что злоумышленник обладает всей возможной информацией о применяемом криптоалгоритме и неизвестен ему только ключ шифрования. Это свойства оценки криптостойкости имеет место быть по причине того, что даже если держать весь алгоритм шифрования в тайне, информация рано или поздно может быть скомпрометирована.

Стойкость криптосистемы зависит от таких параметров, как сложность алгоритма, длина ключа, методы реализации шифра и т.д. Процесс оценки криптостойкости систем условно можно разделить на три этапа:

1. Оценить сколько времени и вычислительных мощностей потребуется для полного перебора всех ключей. Сложность вычислительных алгоритмов можно оценивать числом элементарных операций, с учетом затраты на их выполнение. Это число должно выходить за пределы возможностей современных компьютерных систем.

Поиск уязвимостей алгоритма шифрования к известным атакам, таким как: линейный криптоанализ, дифференциальный криптоанализ и другие. Также, оценка статистической безопасности шифра, о которой говорит отсутствие зависимости между входной и выходной последовательностями и между ключами, используемыми в процессе шифрования, а также широко проявляемое диффузионное свойство. В этот этап входит проверка атаки на реализацию, которая происходит на конкретный программный-комплекс.

Длительность изучения алгоритма и реализаций на программных-комплексах, так как чем дольше происходит анализ, тем больше доказывается криптостойкость алгоритма.

Шифр считается достаточно надежным, если отсутствует возможность раскрыть его способом более эффективным, чем полным перебором по всему ключевому пространству.

Шифр Вернама

Шифр Вернама, изобретённый в 1917 году, является симметричной системой шифрования, а также разновидностью криптосистемы одноразовых блокнотов. Данная криптосистема была предложена для использования в системе телеграфных сообщений. Текст сообщения имел бинарный вид, а секретный ключ представлялся как случайный набор букв такого же алфавита, что и сообщение.

Для того, чтобы получить шифротекст, открытый текст объединяется с секретным ключом при помощи операции исключающее ИЛИ. К примеру, для шифрования двоичной последовательности (100110) с применением закрытого ключа (010010) будет получен шифртекст (110100). Для того, чтобы получить исходное сообщения (100110) из полученного шифртекста достаточно применить обратную операцию исключающее ИЛИ к принятому шифртексту (110100) и закрытому ключу (010010).

Для того, чтобы шифр имел полную криптографическую стойкость, закрытый ключ должен обладать как минимум тремя, приведенными ниже, свойствами:

1. Иметь случайное равномерное и равновероятное распределение двоичных символов

Размер закрытого ключа должен полностью совпадать с размером заданного сообщения

Закрытый ключ должен использоваться ровно один раз на сообщение.

Важной особенностью данного криптоалгоритма является то, что при применении шифра Вернама за перехваченным шифрованным текстом может скрываться абсолютно любой открытый текст. Аналогично, совершенно любому открытому тексту также можно подобрать гамму, которая будет порождать данный шифротекст.

Поскольку гамма является ключом, то при перехвате шифрованного сообщения невозможно отвергнуть ни одного открытого текста, имеющего такую же длину. [4]

Криптографическая стойкость шифра Вернама

Клод Шеннон в 1945 году доказал абсолютную стойкость шифра Вернама. При перехвате шифртекста третьим лицом, перехватчик не сможет узнать никакой информации об исходном сообщении или закрытом ключе. В криптографии шифр Вернама считается идеально безопасной системой. Для реализации такой системы необходимо обеспечить уникальное гаммирование исходного сообщения. Важным условием гаммирования являются:

1. При шифровании нового сообщения используется новая гамма-последовательность. Повторное использование гамма-последовательности невозможно из-за свойств операции “исключающее ИЛИ”.

Рассмотрим пример:

m’1 = m1 c

m’2 = m2 c

m’1m’2=(m1 с) (m2 с) = m1 m2

Результат не зависит от гамма-последовательности С и полностью зависит от исходного сообщения. Так как естественные языки имеют высокую избыточность, то результат поддается частотному анализу, соответственно можно подобрать исходные тексты и не важна гамма-последовательность.

Если гамма-последовательность будет сгенерирована на аппаратных генераторах случайных чисел, то гамма не будет считаться случайной и с помощью перебора, возможно подобрать начальное состояние генератора.

Длина гамма-последовательности не должна быть короче длины исходного сообщения, так как при выборе гамма-последовательности меньшей длины можно будет подобрать размер, проанализировать блоки шифртекста и подобрать биты гамма-последовательности.

Недостатки шифра Вернама

Несмотря на то, что шифр Вернама является абсолютно криптографически стойкой и самой безопасной системой, она имеет ряд существенных недостатков.

1. Главным необходимым условием криптографической стойкости шифра является случайность последовательности закрытого ключа. Полученная при помощи любого алгоритма последовательность считается не абсолютно случайной, а псевдослучайной.

Невозможность подтверждения целостности и подлинности переданного сообщения получателю. Получатель не может удостовериться, что сообщение пришло без искажений, соответствует исходному, а также подлинность отправителя, так как при перехвате и расшифровке шифртекста третьим лицом, он может заменить передаваемое сообщение своим сообщением точно такой же длины и зашифровать взломанным закрытым ключом.

Необходимость иметь для множества сообщений множество закрытых уникальных ключей. Это может привести к разрыву канала между двумя лицами до момента получения новых сгенерированных закрытых ключей. Также невозможно заранее знать размер передаваемого сообщения.

Закрытые ключи нельзя передавать по каналу связи, в виду его недостаточной защищенности – шифр будет защищенным ровно настолько, насколько защищен канал связи. При этом, принимая в учет тот факт, что ключ имеет ту же длину, что и сообщение, передать ключ по защищенному каналу не будет хоть немного проще, чем передать по каналу само сообщение.

Применение повторного ключа влечет за собой подбор закрытого ключа и последующее вскрытие шифра.

Шеннон

Шифры с бегущим ключом - это такие шифры, в которых исходное сообщение, ключи и шифртекст представлены последовательностями символов одного языка.

В криптосистему с бегущим ключом входит система Шеннона. Шеннон предложил два шифра с бегущим ключом, которые простые в реализации и надежные, но не исследовал их свойства. В первом шифре Шеннон предложил, что вместо одного текста на английском языке использовать D различных, прибавляя их к исходному сообщению. При достаточно большом количестве ключей повысится надежность шифра. Но Шеннон не привел никаких доказательств данного шифра. Во втором шифре Шеннон предложил брать каждую N-ую букву текста пропуская остальные буквы, тем самым выбор каждого следующего символа будет практически независим. Этот метод также должен был дать надежный шифр.

Обоснование выбора языка

Качество программного обеспечения. Основным преимуществом языка Python является удобочитаемость и высокое качество кода. Код читается намного легче, что ведет к многоразовому использованию и удобному дальнейшему обслуживанию. Единообразие оформления кода упрощает работу тех, кто не участвовал в проекте.

По сравнению с такими языками как C, C++, Java и C, Python повышает производительность разработчика. Объем кода написанного на языке Python в основном составляет 1/3 от такого же кода на языке C. Тем самым разработчик тратит меньше времени на отладку. Также программы на языке Python запускаются сразу, не проходя через этапы компиляции и связывания.

Большая часть кода на языке Python кроссплатформенная. Перенос программы из операционной системы Windows в Linux не требует от разработчика переписывания исходного кода. Python также предоставляет возможность создания переносимых интерфейсов, доступа к базам данных, веб-интерфейсов и многих других программ.

Стандартные библиотеки python предоставляют возможности работы со многими СУБД, выполнять парсинг данных, работа со множеством сетевых протоколов. Сообщество языка очень активно развивает множество библиотек для работ с математическими вычислениями, доступам к последовательному порту, разработка игровых движков, веб-программирование.

В языке отлично реализована интеграция с другими языками. Например в коде на языке Python мы можем вызывать стандартные функции библиотек С, С++ и так же вызываться в программах С, С++.

Python максимально оптимизирован для достижения высокой скорости разработки, у языка простой и понятный синтаксис, динамическая типизация, отсутствует этап компиляции.

Как вывод язык увеличивает производительность разработчика во много раз по сравнению с другими языками.

# Статистические тесты для поточных шифров

# Поточные шифры построены на имитации концепции одноразового блокнота. В отличие от одноразового блокнота, эти шифры не являются достаточно надежными, однако, их намного удобнее использовать на практике в виду использования в их алгоритме ключа для генерации псевдослучайной последовательности. Такая последовательность должна быть неотличима от совершенно случайной последовательности. Данное утверждение означает, что последовательность должна иметь одинаковое количество нулей и единиц, равновероятность распределения символов, а также отсутствие в последовательности статистических взаимосвязей.

# Для того, чтобы определить, насколько полученная последовательность близка к совершенно случайной последовательности, в криптоанализе используются такие методы исследования, как статистическое тестирование.

# Основным принципом, на котором строятся статистические тесты является проверка нулевой гипотезы Н0. Нулевая гипотеза утверждает, что тестируемая последовательность является совершенно случайной. Альтернативной гипотезой Н1 является гипотеза о том, что последовательность случайной не является. По результатам статистического тестирования нулевая гипотеза либо подтверждается, либо отвергается.

Генератор, который подлежит тестированию, производит двоичные последовательности фиксированной длины. Выполняется набор статистических тестов, каждый из которых оценивает последовательность по определенным критериям и вырабатывает значение p-value. На основе полученных значений может быть сделано заключение о качестве тестируемой последовательности. [5]

# В результате прохождения тестов может быть три варианта исходов:

# Анализ не показал отклонение от случайности

# Анализ ясно указал на отклонение от случайности

# Анализ является недостаточным для выводов

Наличие хороших статистических свойств является одним из наиболее важных критериев надежности алгоритма поточного шифрования.

Описание шифра

Для соединения двух клиентов выбирается набор из 128 файлов с ключами, которые передаются обоим клиентам.

Затем клиенты высчитывают по алгоритму Диффи-Хелмана закрытый ключ. Закрытый ключ обязательно должен быть 128-битным. Каждый бит ключа обозначает номер файла. Если бит закрытого ключа равен единице, то файл с ключом будет использован в шифровании, иначе файл не выбирается.

Первый клиент шифрует сообщение с выбранными файлами. Шифрование происходит так, что исходное сообщение выполняет операцию «исключающее ИЛИ» с выбранными файлами.

Затем первый клиент отправляет данный шифротекст второму клиенту.

Второй клиент принимает шифротекст и производит обратную операцию для расшифрование. Расшифрование происходит так, что исходное сообщение выполняет операцию «исключающее ИЛИ» с выбранными файлами.

1. Выбор текстов для шифрования

Клиент устанавливает соединение по протоколу TCP с сервером. Сервер в ответ из множества файлов выбирает 128 файлов, которые будут переданы клиенту.

def selection(self):

list = os.listdir("textCode/")

len\_list = len(list)

answer = []

while len(ans) != 128:

select = random.randint(0,lenList-1)

if (list[select] in answer) == False:

answer.append(list[select])

return answer

list — Содержит список из всех файлов в директории «textCode»

len\_list — Содержит количество файлов в директории

answer — Выходной список размерностью 128 записей.

Select — Номер случайно выбранного документа из списка list.

Сервер считывает 262Кб, что представляет достаточное количество для шифрования сообщения клиента, из каждого выбранного файла. Затем по открытому соединению передает его клиенту. После передачи каждого файла, сервер отправляет системный запрос «next», указывающий на передачу следующего файла.

def send\_files(self):

list = self.selection()

in\_len = 262144

for text in list:

file = open("textCode/" + text, "rb")

line = file.read(in\_len)

self.request.send(line.encode())

self.request.send(b"next")

file.close()

list — Результат функции selection. Содержит 128 имен выбранных файлов.

in\_len — Количество считываемых с файла данных и передаваемых.

line — Считанные данные из файла

Клиент принимает файлы от сервера, при системной команде «next» начинается запись в следующий файл. После того, как приняты 128 файлов функция завершается.

def downloadFiles(server):

data = ""

for i in range(0, 128):

file = open("text/"+str(i)+".txt", "wb)

while True:

data = server.recv(262144\*2)

if data == b"next“ or not data:

break

else:

file.write(data)

file.close()

data — Строка, в которую будут записаны принятые от сервера данные.

2. Закрытый ключ

Первый клиент высчитывает параметры закрытого ключа по алгоритму Диффи-Хеллмана и передает параметры второму клиенту, он высчитывает и отправляет свой открытый ключ для расчета закрытого ключа.

def DiffieHellman(self):

self.Q = random.randint((2\*\*128), (2\*\*129)-1)

while self.miller\_rabin(self.Q) == False or self.miller\_rabin(2 \* self.Q + 1) == False:

self.Q = random.randint((2\*\*128), (2\*\*129)-1)

self.P = 2 \* self.Q + 1

self.G = random.randint((2\*\*128), (2\*\*129)-1)

while (self.powmod(self.G, self.Q, self.P) == 1):

self.G = random.randint((2\*\*128), (2\*\*129)-1)

self.Xa = random.randint((2\*\*128), (2\*\*129)-1)

self.Ya = self.powmod(self.G, self.Xa, self.P)

self.request.send(str(self.P).encode())

self.request.send(str(self.G).encode())

self.request.send(str(self.Ya).encode())

self.Yb = int(self.request.recv(1024).decode())

self.Zb = self.powmod(self.Yb, self.Xa, self.P)

Функция второго клиента.

def generation\_key(server):

P = int(server.recv(1024).decode())

G = int(server.recv(1024).decode())

Ya = int(server.recv(1024).decode())

Xb = random.randint((2\*\*128), (2\*\*129)-1)

Yb = powmod(G, Xb, P)

server.send(str(Yb).encode())

return powmod(Ya, Xb, P)

3) Клиенты проходят по битам закрытого числа, для определения списка необходимых для шифрования текстов. Если бит равен единице то текст подходит, иначе не подходит.

def bits(n):

list\_bits = []

for i in range(0, 128):

a = bit(n, i)

list\_bits.append(a)

return list\_bits

def bit(num, pos):

return (num & (1 << pos)) >> pos

list\_bits — Выходной список, в котором хранятся значение 0 или 1, в зависимости от того, подходит нам файл или нет.

4) Первый клиент вводит сообщение и шифрует его. Исходный код шифруется с подходящими файлами.

def xorTest(text1, list):

for i in range(0, 128):

if list[i] == 1:

with open("text/{}.txt".format(i), "rb") as f:

text1 = xor(text1, bytearray(f.read()))

return text1

list — Список из 0 и 1 обозначающие подходит ли файл для шифрования.

def xor(text1, text2):

lenText2 = len(text2)

j = 0

for i in range(len(text1)):

text1[i] = text1[i] ^ text2[j]

j += 1

if j == lenText2-1:

j = 0

return text1

5) Первый клиент отправляет второму клиенту зашифрованное сообщение.

def sendMessage(server, message):

server.send(line.encode())

6) Второй пользователь принимает сообщение и производит операцию расшифровывания такую же как в пункте 4.

Исследование шифрования

Для исследования реализации алгоритма шифрования были выбраны тексты содержащие русский язык. Сервер хранит 300 текстов, которые будут служить файлами для шифрования клиентов. Сервер выбирает 128 файлов и передает клиентам. Клиенты шифруют исходные сообщения и записывают их в файлы. Анализ содержания файлов проводится статистическим тестом «Стопка книг». Для того, чтобы дать точные значения тестов были проведены 100 сессий. Входные данные для тестирования: алфавит будет состоять из 256 символов, верхняя часть 100 элементов, нижняя 156.

Шифрование с помощью текстов содержащих только русский алфавит показал плохие результаты. Причина таких результатов в избыточности содержимого данных текстов.

Во втором случае сервер хранить 300 файлов расширения djvu и jpg. Данный выбор позволит снизить избыточность файлов для шифрования, из которых будет формироваться шифротекст. Условия теста такие же как и в 1 случае, были проведены 100 сессий шифрования.

В данном случае, реализация шифра показала результаты:

В третьем случае сервер хранит 300 файлов заполненных псевдослучайными значениями. Каждый символ сгенерированных файлов будет иметь алфавит длинной 255 и включать в себя каждый символ алфавита.

Из приведенных результатов можно сделать вывод, что реализация шифра устойчива в случае, если использовать тексты (). Использование списка файлов с русским языком, привел к плохой защищенности шифротекста так как вероятность следующего бита шифротекста не была равна 1 / 2.

Вывод

Было проведена разработка и исследование характеристик реализации шифра Шеннона. Благодаря успешному подбору файлов шифрования реализация обеспечивает защищенный канал связи между клиентами и гарантию того, что перехваченное сообщение не будет расшифровано за реальное количество времени. Данная реализация позволяет общение клиентов в режиме передачи сообщений либо файлов. Разработанная архитектура клиент-серверного приложения может использоваться в разработке чатов, файловых хранилищ.

Обязательными условиями реализации были выявлены:

1) Закрытый ключ должен быть длины 128 бит.

2) Файлы ключи должны состоять из файлов ().

3) Рекомендуемый размер файлов для шифрования 128Кб, максимальный размер комплекта файлов будет составлять 16Мб.

4)

В дальнейшем планируется исследование возможностей использования шифра для трансляции аудио/видео потока от одного клиента второму клиенту либо множеству клиентов. Что позволит проводить защищенные конференции, звонки, осуществлять удаленное управление.

Список литературы

1. Э. Мэйволд. Безопасность сетей. — 2006. — 528 с

1. Клод Шеннон. Теория связи в секретных системах / пер. с англ. В. Ф.

Писаренко // Работы по теории информации и кибернетике / Под редакцией Р. Л. Добрушина и О. Б. Лупанова. — М. : Издательство иностранной литературы, 1963. — 829 с.

1. Проскурин В.Г. Защита программ и данных.

Издательство: М.: Академия, второе издание, 2012 г. 208 с.

1. Бабаш А. В., Гольев, Ю. И., Ларин Д. А. и др.

Криптографические идеи XIX века // Защита информации. Конфидент — СПб.: 2004. — вып. 3.

1. Е.В. Игоничкина Статистический анализ поточных шифров

//https://cyberleninka.ru/article/v/statisticheskiy-analiz-potochnyh-shifrov