**0Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа информационных технологий и робототехники

Отделение - ОИТ

Направление – Информатика и вычислительная техника

**Отчёт по лабораторной работе № 1**

**«Шифрование»**

по дисциплине: Защита информации

Выполнил:

студент гр. 8В7Б Мальцев М.Ю.

Проверил:

Доцент ОИТ Ботыгин И.А.

Томск 2020 г.

Оглавление

Оглавление

[Оглавление 2](#_Toc55337119)

[Результат работы 3](#_Toc55337120)

[1. Введение 3](#_Toc55337121)

[2. История возникновения криптографии 4](#_Toc55337122)

[3. Шифры, их виды и свойства 7](#_Toc55337123)

[3.1 Симметричные криптографические системы 7](#_Toc55337124)

[3.2 Асимметричные криптографические системы 10](#_Toc55337125)

[4. Хеширование 12](#_Toc55337126)

[5. Электронная цифровая подпись 13](#_Toc55337127)

[6. Пример работы шифрования на собственной программе 14](#_Toc55337128)

[Вывод 17](#_Toc55337129)

[Приложение А. Исходный код приложения 18](#_Toc55337130)

Результат работы

1. Введение

Проблема защиты информации путем ее преобразования, исключающего ее прочтение посторонним лицом волновала человеческий ум с давних времен. Криптографические методы защиты информации в автоматизированных системах могут применяться как для защиты информации, обрабатываемой в ЭВМ или хранящейся в различного типа ЗУ, так и для закрытия информации, передаваемой между различными элементами системы по линиям связи.

Криптографическое преобразование как метод предупреждения несанкционированного доступа к информации имеет многовековую историю. В настоящее время разработано большое количество различных методов шифрования, созданы теоретические и практические основы их применения. Подавляющие число этих методов может быть успешно использовано и для закрытия информации.

Почему проблема использования криптографических методов в информационных системах (ИС) стала в настоящий момент особо актуальна?

С одной стороны, расширилось использование компьютерных сетей, в частности глобальной сети Интернет, по которым передаются большие объемы информации государственного, военного, коммерческого и частного характера, не допускающего возможность доступа к ней посторонних лиц.

С другой стороны, появление новых мощных компьютеров, технологий сетевых и нейронных вычислений сделало возможным дискредитацию криптографических систем еще недавно считавшихся практически не раскрываемыми.

Криптография занимается поиском и исследованием математических методов преобразования информации.

Криптогра́фия — наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним), целостности данных (невозможности незаметного изменения информации), аутентификации (проверки подлинности авторства или иных свойств объекта), а также невозможности отказа от авторства.

Изначально криптография изучала методы шифрования информации — обратимого преобразования открытого (исходного) текста на основе секретного алгоритма или ключа в шифрованный текст (шифротекст). Традиционная криптография образует раздел симметричных криптосистем, в которых зашифрование и расшифрование проводится с использованием одного и того же секретного ключа. Помимо этого раздела современная криптография включает в себя асимметричные криптосистемы, системы электронной цифровой подписи (ЭЦП), хеш-функции, управление ключами, получение скрытой информации, квантовую криптографию.

1. История возникновения криптографии

Большинство современных исследователей связывают появление криптографии с появлением письменности, указывая, что эти процессы произошли почти одновременно.

Методы секретной переписки были изобретены независимо в различных государствах древнего Востока, таких как Египет, Китай и Шумер, хотя сегодня очень трудно судить об уровне развития криптологии в этих обществах. Клинопись, рисуночное и иероглифическое письмо само по себе было крайне сложно и требовало длительного обучения, так что вопрос о шифровании сообщений часто попросту не поднимался, так как круг грамотных лиц был весьма ограничен.

Всю сложность данного вопроса иллюстрирует один пример: найдено множество глиняных табличек с клинописными знаками, записанными в несколько слоев (первоначальная запись замазывалась глиной и поверх нее наносилась новая).

Однако с развитием фонетического письма и значительным упрощением письменности, криптология получает значительный стимул к развитию. Развитию этой области знаний способствовали и развитие торговли, военного дела и дипломатической деятельности, которые создавали необходимый спрос на «продукцию» криптографов.

Наибольшее развитие в это время криптография получает в полисах Древней Греции, а позже в Риме. Основные криптографические системы, многие из которых используются вплоть до наших дней были разработаны в Древней Греции и получили широкое практическое применение в Риме. В Древней Греции использовались как шифры замены, так и шифры перестановки. Так наиболее распространенным и получившим широкую известность в античном мире шифром замены является т.н. шифр Цезаря, описанный Светонием. Для того чтобы зашифровать сообщение, каждую его букву заменяли на другую букву латинского алфавита, но со сдвигом влево или вправо. Цезарь в своих посланиях к сенату заменял все буквы на три отстоящие слева, Август применял тот же шифр, но со сдвигом в четыре знака.

Наибольших успехов в криптографии в античный период добилась Спарта, где активно использовались все известные виды шифров и были созданы первые дошедшие до нас шифровальные устройство. Первым таким прибором, реализующим шифр перестановки была т.н. «сциталла» (ококо VI-V вв. до н.э.). На цилиндр определенного диаметра по спирали наматывался ремень, на который наносили буквы вдоль оси цилиндра. В результате в развернутом виде все буквы смешивались, а если намотать ремень на цилиндр того же диаметра, то сообщение вновь становилось понятным.

В то же время стойкость данного шифра была невелика, а позже Архимед предложил устройство (т.н. антисциталла), с помощью которого расшифровка подобного сообщения без нужного цилиндра была весьма простой и быстрой. Ремень наматывали на коническое «копье» и сдвигали вверх и вниз до тех пор, пока не находили нужный диаметр и текст сообщения становился понятным.

Существовали и другие способы «механизации» криптографического дела, связанные прежде всего с именем древнегреческого полководца Энея Тактики. Он создал т.н. «диск Энея», получивший в Древней Греции широкое применение. В небольшом диске высверливались отверстия, соответствующие буквам алфавита, через которые продевалась нить, в соответствии с буквами шифруемого текста. Для расшифровки нить вытягивали, получая обратную последовательность букв. Этот крайне примитивный на первый взгляд способ шифрования, был весьма эффективен, так как противнику, перехватившему сообщение было неизвестно, какая буква соответствует каждому отверстию. Кроме того, если возникала опасность перехвата сообщения, нить можно было легко порвать, тем самым уничтожив его.

Существовала также и «линейка Энея», использовавшая тот же принцип, что и диск. Значительным вкладом Энея стал и изобретенный им т.н. «книжный шифр», активно использовавшийся вплоть до ХХ века. В своем трактате «Об обороне укрепленных мест», Эней предлагал прокалывать малозаметные дырки над буквами текста какой-либо книги. Сложив вместе отмеченные буквы адресат получал исходное сообщение. Римляне усовершенствовали диск Энея, создав первую многодисковую шифрующую систему. На общую ось одевали два диска с хаотичным расположением букв. Каждой букве первого диска соответствовала буква второго, что и составляло шифр.

Значительным шагом вперед, по сравнению с предыдущими системами шифрования представлял шифр, предложенный Полибием (ок. II в. до н.э.). Механизм его состоял в следующем: в квадрат определенных размеров (в соответствии с количеством букв алфавита – для латинского 5Х5, для русского 5Х6, при этом некоторые буквы редуцируются) вписываются буквы алфавита. Каждая клетка квадрата имеет двузначные координаты, на которые и заменяется при шифровании. Первоначально буквы записывались в естественном порядке, что значительно снижало стойкость шифра. Позднее буквы стали располагать хаотично, но это требовало наличие записанного ключа, что также было небезопасно. Выход был найден в применении т.н. ключевого слова. Берется какое-либо слово, из него убираются повторяющиеся буквы, а оставшиеся записываются в первые клетки квадрата. Пустые клетки заполняются буквами алфавита в естественном порядке. Полибианский квадрат стал одной из наиболее широко распространенных криптографических систем, когда-либо употреблявшихся. Этому способствовала его достаточно высокая стойкость (во всяком случае до автоматизации дешифрующих систем) – так квадрат 5Х5 для латинского алфавита содержит 15511210043331000000000000 (расчет весьма приблизителен) возможных положений, что практически исключает его дешифрование без знания ключа.

Интересно отметить, что полибианский квадрат дожил до наших дней и лег в основу т.н. «тюремного шифра», используемого заключенными при перестукивании. В нем буквы расположены в естественном порядке, а число ударов обозначает координату каждой буквы. Но так как используется естественное расположение букв, т.е секретного ключа нет, «тюремный шифр» является скорее способом кодировки сообщения, чем способом ее засекречивания.

Эти криптографические системы активно применялись в Древней Греции и Риме и надолго определили характер криптографии. В условиях необходимости ручного расшифрования, полибианский квадрат был практически неуязвимым шифром, а сциталла и диск Энея, достаточно простые, тем не менее, позволяли оперативно шифровать и расшифровывать информацию, что делало их применимыми, скажем в полевых условиях для оперативной передачи приказов.

1. Шифры, их виды и свойства

В криптографии криптографические системы (или шифры) классифицируются следующим образом:

-симметричные криптосистемы

-асимметричные криптосистемы

* 1. Симметричные криптографические системы

Под симметричными криптографическими системами понимаются такие криптосистемы, в которых для шифрования и расшифрования используется один и тот же ключ, хранящийся в секрете. Все многообразие симметричных криптосистем основывается на следующих базовых классах:

I. Моно - и многоалфавитные подстановки.

Моноалфавитные подстановки - это наиболее простой вид преобразований, заключающийся в замене символов исходного текста на другие (того же алфавита) по более или менее сложному правилу. В случае моноалфавитных подстановок каждый символ исходного текста преобразуется в символ шифрованного текста по одному и тому же закону. При многоалфавитной подстановке закон преобразования меняется от символа к символу. Один и тот же шифр может рассматриваться и как моно - и как многоалфавитный в зависимости от определяемого алфавита.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Исходные символы шифруемого текста | а | б | в | г | д | е | ж | з | и | к | л | м | н | о | п | р | с | т | у | ь | … |
| Заменяющие символы | s | p | x | l | r | z | i | m | a | y | e | d | w | t | b | g | v | n | j | o | … |

Например, самой простой разновидностью является прямая (простая) замена, когда буквы шифруемого сообщения заменяются другими буквами того же самого или некоторого другого алфавита. Таблица замены может иметь следующий вид:

Используя эту таблицу, зашифруем слово двигатель. Получим следующее: rxflsnzeo

II. Перестановки - перестановка местами символов исходного текста по некоторому правилу. Шифры перестановок в настоящее время не используются в чистом виде, так как их криптостойкость недостаточна, но они входят в качестве элемента в очень многие современные криптосистемы.

Самая простая перестановка - написать исходный текст наоборот и одновременно разбить шифрограмму на пятерки букв. Например, из фразы

ПУСТЬ БУДЕТ ТАК, КАК МЫ ХОТЕЛИ

получится такой шифротекст:

ИЛЕТО ХЫМКА ККАТТ ЕДУБЪ ТСУП

III. Блочные шифры - семейство обратимых преобразований блоков (частей фиксированной длины) исходного текста. Фактически блочный шифр - это система подстановки на алфавите блоков. Она может быть моно - или многоалфавитной в зависимости от режима блочного шифра. Иначе говоря, при блочном шифровании информация разбивается на блоки фиксированной длины и шифруется поблочно. Блочные шифры бывают двух основных видов: шифры перестановки (transposition, permutation, P-блоки) и шифры замены (подстановки, substitution, S-блоки) 1. В настоящее время блочные шифры наиболее распространены на практике.

Американский стандарт криптографического закрытия данных DES (Data Encryption Standard), принятый в 1978 г., является типичным представителем семейства блочных шифров и одним из наиболее распространенных криптографических стандартов на шифрование данных, применяемых в США. Этот шифр допускает эффективную аппаратную и программную реализацию, причем возможно достижение скоростей шифрования до нескольких мегабайт в секунду. Первоначально метод, лежащий в основе данного стандарта, был разработан фирмой IBM для своих целей. Он был проверен Агентством Национальной Безопасности США, которое не обнаружило в нем статистических или математических изъянов.

DES имеет блоки по 64 бит и основан на 16-кратной перестановке данных, также для шифрования использует ключ в 56 бит. Существует несколько режимов DES: Electronic Code Book (ECB) и Cipher Block Chaining (CBC).56 бит - это 8 семибитовых символов, т.е. пароль не может быть больше чем восемь букв. Если вдобавок использовать только буквы и цифры, то количество возможных вариантов будет существенно меньше максимально возможных 256. Однако, данный алгоритм, являясь первым опытом стандарта шифрования, имеет ряд недостатков. За время, прошедшее после создания DES, компьютерная техника развилась настолько быстро, что оказалось возможным осуществлять исчерпывающий перебор ключей и тем самым раскрывать шифр.

IV. Гаммирование - преобразование исходного текста, при котором символы исходного текста складываются с символами псевдослучайной последовательности (гамме), вырабатываемой по некоторому правилу. В качестве гаммы может быть использована любая последовательность случайных символов. Процедуру наложения гаммы на исходный текст можно осуществить двумя способами. При первом способе символы исходного текста и гаммы заменяются цифровыми эквивалентами, которые затем складываются по модулю k, где k - число символов в алфавите. При втором методе символы исходного текста и гаммы представляются в виде двоичного кода, затем соответствующие разряды складываются по модулю 2. Вместо сложения по модулю 2 при гаммировании можно использовать и другие логические операции.

Таким образом, симметричными криптографическими системами являются криптосистемы, в которых для шифрования и расшифрования используется один и тот же ключ. Достаточно эффективным средством повышения стойкости шифрования является комбинированное использование нескольких различных способов шифрования. Основным недостатком симметричного шифрования является то, что секретный ключ должен быть известен и отправителю, и получателю.

* 1. Асимметричные криптографические системы

Эти системы характеризуются тем, что для шифрования и для расшифрования используются разные ключи, связанные между собой некоторой зависимостью. Применение таких шифров стало возможным благодаря К. Шеннону, предложившему строить шифр таким способом, чтобы его раскрытие было эквивалентно решению математической задачи, требующей выполнения объемов вычислений, превосходящих возможности современных ЭВМ (например, операции с большими простыми числами и их произведениями). Один из ключей (например, ключ шифрования) может быть сделан общедоступным, и в этом случае проблема получения общего секретного ключа для связи отпадает. Если сделать общедоступным ключ расшифрования, то на базе полученной системы можно построить систему аутентификации передаваемых сообщений. Поскольку в большинстве случаев один ключ из пары делается общедоступным, такие системы получили также название криптосистем с открытым ключом. Первый ключ не является секретным и может быть опубликован для использования всеми пользователями системы, которые зашифровывают данные. Расшифрование данных с помощью известного ключа невозможно. Для расшифрования данных получатель зашифрованной информации использует второй ключ, который является секретным. Разумеется, ключ расшифрования не может быть определен из ключа зашифрования.

В настоящее время наиболее развитым методом криптографической защиты информации с известным ключом является RSA, названный так по начальным буквам фамилий его изобретателей (Rivest, Shamir и Adleman) и представляющий собой криптосистему, стойкость которой основана на сложности решения задачи разложения числа на простые сомножители. Простыми называются такие числа, которые не имеют делителей, кроме самих себя и единицы. А взаимно простыми называются числа, не имеющие общих делителей, кроме 1.

Криптосистема RSA широко применяется в Интернете. Когда пользователь подсоединяется к защищенному серверу, то здесь применяется шифрование открытым ключом с использованием идей алгоритма RSA. Криптостойкость RSA основывается на том предположении, что исключительно трудно, если вообще реально, определить закрытый ключ из открытого. Для этого требовалось решить задачу о существовании делителей огромного целого числа. До сих пор ее аналитическими методами никто не решил, и алгоритм RSA можно взломать лишь путем полного перебора.

Таким образом, асимметричные криптографические системы - это системы, в которых для шифрования и для расшифрования используются разные ключи. Один из ключей даже может быть сделан общедоступным. При этом расшифрование данных с помощью известного ключа невозможно.

1. Хеширование

Хеширование (иногда хэширование, англ. hashing) - преобразование входного массива данных произвольной длины в выходную строку фиксированной длины. Такие преобразования также называются хеш-функциями или функциями свёртки, входной массив – прообразом, а результаты преобразования - хешем, хеш-кодом, хеш-образом, цифровым отпечатком или дайджестом сообщения (англ. message digest).

Хеш-функция – легко вычислимая функция, преобразующая исходное сообщения произвольной длины (прообраз) в сообщение фиксированное длины (хеш-образ), для которой не существует эффективного алгоритма поиска коллизий.

Коллизией для функции h называется пара значений x, y, x ≠ y, такая, что h(x) = h(y). Т.о. хеш-функция должна обладать следующими свойствами:

- для данного значения h(x) невозможно найти значение аргумента x. Такие хеш-функции называют стойкими в смысле обращения или стойкими в сильном смысле;

- для данного аргумента x невозможно найти другой аргумент y такой, что h(x) = h(y). Такие хеш-функции называют стойкими в смысле вычисления коллизий или стойкими в слабом смысле.

В случае, когда значение хеш-функции зависит не только от прообраза, но и закрытого ключа, то это значение называют кодом проверки подлинности сообщений (Message Authentication Code, MAC), кодом проверки подлинности данных (Data Authentication Code, DAC) или имитовставкой.

На практике хеш-функции используют в следующих целях:

- для ускорения поиска данных в БД;

- для проверки целостности и подлинности сообщений;

- для создания сжатого образа, применяемого в процедурах ЭЦП;

- для защиты пароля в процедурах аутентификации.

1. Электронная цифровая подпись

Электронная подпись (ЭП), Электронная цифровая подпись (ЭЦП), Цифровая подпись (ЦП) позволяет подтвердить авторство электронного документа (будь то реальное лицо или, например, аккаунт в криптовалютной системе). Подпись связана как с автором, так и с самим документом с помощью криптографических методов, и не может быть подделана с помощью обычного копирования.

Широко применяемая в настоящее время технология электронной подписи основана на асимметричном шифровании с открытым ключом и опирается на следующие принципы:

Можно сгенерировать пару очень больших чисел (открытый ключ и закрытый ключ) так, чтобы, зная открытый ключ, нельзя было вычислить закрытый ключ за разумный срок. Механизм генерации ключей строго определён и является общеизвестным. При этом каждому открытому ключу соответствует определённый закрытый ключ. Если, например, Иван Иванов публикует свой открытый ключ, то можно быть уверенным, что соответствующий закрытый ключ есть только у него.

Имеются надёжные методы шифрования, позволяющие зашифровать сообщение закрытым ключом так, чтобы расшифровать его можно было только открытым ключом. Механизм шифрования является общеизвестным.

Если электронный документ поддается расшифровке с помощью открытого ключа, то можно быть уверенным, что он был зашифрован с помощью уникального закрытого ключа. Если документ расшифрован с помощью открытого ключа Ивана Иванова, то это подтверждает его авторство: зашифровать данный документ мог только Иванов, т.к. он является единственным обладателем закрытого ключа.

Однако шифровать весь документ было бы неудобно, поэтому шифруется только его хеш — небольшой объём данных, жёстко привязанный к документу с помощью математических преобразований и идентифицирующий его. Шифрованный хеш и является электронной подписью.

1. Пример работы шифрования на собственной программе

Теперь перейдем к практическому примеру работы шифрования. Для этого был использован язык программирования Kotlin с библиотеками Cipher и KeyGenerator.

Для реализации поставленной задачи был создан класс EncoderExample, получающий в конструктор алгоритм шифрования.

**class** **EncoderExample**(**private** **val** algorithm: String) { … }

Данный класс имеет два поля, за процесс шифрования – keyGenerator и secretKey. Сначала инициализируется KeyGenerator на основе полученного алгоритма, после чего в классе сохраняется секретный ключ данного генератора ключей, чтобы в дальнейшем иметь возможность дешифровать.

Далее, был создан метод encode(inputStr), принимающий любую строку и возвращающий зашифрованную.

// Метод шифрования строки

**fun** **encode**(inputStr: String): String {

// Инициализация шифра

**val** cipher: Cipher = Cipher.getInstance(algorithm)

cipher.init(Cipher.ENCRYPT\_MODE, secretKey) // Иниц. шифрования с ключом

// Возвращаем зашифрованную строку

**return** Base64.getEncoder()

.encodeToString(cipher.doFinal(inputStr.toByteArray(StandardCharsets.UTF\_8)))

}

Данный метод работает следующим образом: сначала берется экземпляр шифра с указанным алгоритмом шифрования, указывается решим шифра (в данном случае, шифрование). Далее, изначальная строка конвертируется в байтовый массив, который шифруется по указанному алгоритму и возвращается в виде строки.

Следующий реализованный метод – encode(encodeString), принимающий зашифрованную строку и возвращающий расшифрованную строку.

// Метод дешифрования строки

**fun** **decode**(encodedString: String): String {

// Инициализация шифра

**val** cipher: Cipher = Cipher.getInstance(algorithm)

// Включаем режим расшифрования с тем же ключом

cipher.init(Cipher.DECRYPT\_MODE, secretKey) // Иниц. дешифрования с ключом

// Возвращаем дешифрованную строку

**return**

**String**(

cipher.doFinal(

Base64.getDecoder()

.decode(encodedString.toByteArray(StandardCharsets.UTF\_8))

)

)

}

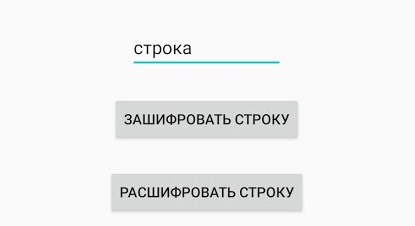
Приведенный метод работает так же, как и метод encode, за исключением указания режима decrypt у класса cipher.

В данном примере использовался алгоритм для симметричного шифрования под названием DES (Data Encryption Standard), разработанный фирмой IBM и утвержденный правительством США.

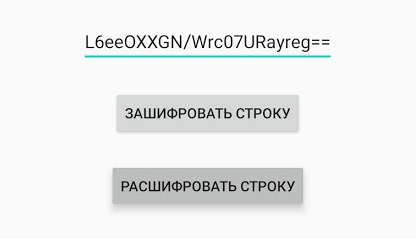
// Инициализация класса, шифрующего по алгоритму DES

**val** encoder = EncoderExample("DES")

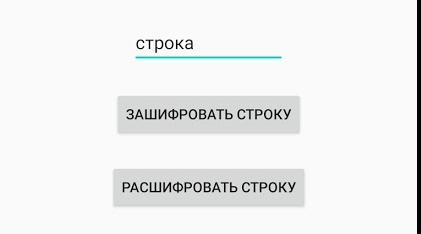
Интерфейсом, выполняющим данный пример, стало android приложение, представленное на рисунке ниже.



1. – Изначальная строка в поле ввода



1. – Зашифрованная строка



1. – Расшифрованная строка

По нажатию первой или второй кнопки вызывается метод encode или decode соответственно. Входная строка берется с поля ввода, результат выводится туда же.

Как видно из приведенных выше скриншотов, программа корректно выполняет шифрование и дешифрование с одинаковым ключом.

Вывод

В процессе выполнения лабораторной работы было проведено исследование одной из сфер безопасности – шифрование. Была изучена ее история развития, приведены примеры и типы шифрования. Теоретические знания подкреплены практическим примером шифрования и дешифрования строки, реализованным на языке программирования Kotlin с графическим интерфейсом операционной системы Android.

Приложение А. Исходный код приложения

EncoderExample:

**class** **EncoderExample**(**private** **val** algorithm: String) {

**private** **val** keyGenerator: KeyGenerator = KeyGenerator.getInstance(algorithm) // Генератор ключа

**private** **val** secretKey = keyGenerator.generateKey() // Генерация ключа шифра

// Метод шифрования строки

**fun** **encode**(inputStr: String): String {

// Инициализация шифра

**val** cipher: Cipher = Cipher.getInstance(algorithm)

cipher.init(Cipher.ENCRYPT\_MODE, secretKey) // Иниц. шифрования с ключом

// Возвращаем зашифрованную строку

**return** Base64.getEncoder()

.encodeToString(cipher.doFinal(inputStr.toByteArray(StandardCharsets.UTF\_8)))

}

// Метод дешифрования строки

**fun** **decode**(encodedString: String): String {

// Инициализация шифра

**val** cipher: Cipher = Cipher.getInstance(algorithm)

// Включаем режим расшифрования с тем же ключом

cipher.init(Cipher.DECRYPT\_MODE, secretKey) // Иниц. дешифрования с ключом

// Возвращаем дешифрованную строку

**return** **String**(

cipher.doFinal(

Base64.getDecoder().decode(encodedString.toByteArray(StandardCharsets.UTF\_8))

)

)

}

}

Android View:

// Инициализация класса, шифрующего по алгоритму DES

**val** encoder = EncoderExample("DES")

// Шифрование по нажатию кнопки

encodeButton.setOnClickListener {

**val** inputStr = inputView.text.toString()

**try** {

inputView.setText(encoder.encode(inputStr))

}

**catch** (e: Exception) {

Toast.makeText(context, "Ошибка при попытке шифрования строки",

Toast.LENGTH\_SHORT).show()

}

}

// Дешифрование по нажатию кнопки

decodeButton.setOnClickListener {

**val** inputStr = inputView.text.toString()

**try** {

inputView.setText(encoder.decode(inputStr))

}

**catch** (e: Exception) {

Toast.makeText(context, "Ошибка при попытке дешифрования строки",

Toast.LENGTH\_SHORT).show()

}

}