УЗБЕКСКОЕ АГЕНТСТВО СВЯЗИ И ИНФОРМАТИЗАЦИИ

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

К защите допустить

Зав. кафедрой

_____

_____*201__ г.*

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему: "Анализ стойкости асимметричных криптосистем"

Выпускник _____ *Цуцилис А. П.*
подпись *Ф.И.О.*

Руководитель _____ *Абдуллаев Д.Г.*
подпись *Ф.И.О.*

Ташкент – 2010

УЗБЕКСКОЕ АГЕНТСТВО СВЯЗИ И ИНФОРМАТИЗАЦИИ

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Факультет: ИТ кафедра: Информационная безопасность

Направление: 5523500 – «Информационная безопасность»

У Т В Е Р Ж Д А Ю

Зав. кафедрой _____

« ___ »_____ 201___ г.

ЗАДАНИЕ

на выпускную квалификационную работу студента Цуцулис Алексея Павловича

на тему: «Анализ стойкости асимметричных криптосистем»

1. *Тема утверждена приказом по университету от « ___ » _____200__г. № _____*

2. *Срок сдачи законченной работы: 06.06.2010*

Исходные данные к работе: Microsoft Power Point, Microsoft Visual Studio, BigInteger(C# class), POP3(C# class).

4. *Содержание расчётно-пояснительной записки (перечень подлежащих к разработке вопросов): Аннотация, введение, обзорная часть, основная часть, практическая часть, БЖД, заключение, список литературы, приложения.*

5. *Перечень графического материала: презентация, выполненная в Microsoft Power Point.*

6. *Дата выдачи задания: 5.12.2009г.*

Руководитель _____
подпись

Задание принял _____

подпись

7. *Консультанты по отдельным разделам выпускной работы*

Наименование раздела	*Консультант*	*Подпись, дата*	
		Задание выдал	*Задание получил*
1. *Обзорная часть* 2. *Основная часть* 3. *БЖД*			

8. *График выполнения работы*

№	*Наименование раздела*	*Срок выполнения*	*Подпись руководителя (консультанта*
1.	*Подбор литературы и выбор темы*		
2.	*Утверждение темы*		
3.	*Оформление обзорной части*		
4.	*Основная часть*		
5.	*Программная часть*		
6.	*БЖД*		
7.	*Предварительная защита*		
8.	*Защита ВКР*		

*Выпускник*_____ *«____»_____200__ г.*

подпись

*Руководитель*_____ *«____»_____200__ г.*

подпись

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Защита информации – задача, на протяжении многих веков существовавшая только в определенных сферах. Однако к концу 20 века, в связи с бурным развитием информационных технологий, проблема информационной безопасности начала волновать все большее число людей. Сегодня уже невозможно найти человека, не сталкивавшегося в той или иной степени с компьютерными технологиями, а, следовательно, и с цифровыми данными. Говоря о цифровом буме, невозможно обойти стороной возникновение глобальной сети Интернет. С одной стороны, Интернет – мощнейший инструмент, помогающий экономить время. С другой, это и неограниченные возможности для разного рода злоумышленников. Естественно, все это приводит нас к тому, что задача защиты информации как никогда более актуальна.

Современный человек, работающий с электронными данными, стремится, прежде всего, к обеспечению следующих трех свойств:

- Целостность (т.е. сохранность данных)
- Конфиденциальность (т.е. сохранение данных в секрете от посторонних лиц)
- Достоверность (т.е. уверенность в том, что данные не были скомпрометированы или сфабрикованы).

Достичь выполнения всех трех вышеперечисленных принципов помогает, пожалуй, самый часто применяемый инструмент информационной безопасности – шифрование.

Вопросами шифрования и шифров занимается наука криптография. Сейчас уже невозможно даже представить, что всего 40 лет назад криптография была «заповедной зоной», доступ в которую имелся только у специализированных организаций. Сегодня мы используем криптографию повсеместно. Хранение паролей, поиск по базам данных, электронные цифровые подписи, системы сертификации вот лишь краткий список областей, так или иначе, применяющих криптографические средства.

Так что же изменилось за эти 40 лет? Как такая закрытая и специфическая область знаний, как наука о шифрах получила такое широкое применение? А дело все в том, что в 1976 году была опубликована революционная работа Диффи и Хеллмана «Новые направления в криптографии», в которой была изложена идея криптографии с открытым ключом (асимметричной криптографии). Идея послужила рождению т.н. гражданской криптографии, той самой, какой мы ее знаем теперь.

Сегодня криптография с открытым ключом применяется повсеместно. Однако проблема ее использования заключается в том, что не существует

универсальных методов, позволяющих определить уровень стойкости той или иной криптосистемы. Эта проблема наиболее актуальна сегодня, когда трудно понять стоит ли доверять таким алгоритмам, как RSA и El-Gamal.

В данной работе я собираюсь рассмотреть некоторый список возможных атак и сформулировать понятие стойкости по отношению к ним. Это покажет, что некоторые криптографические примитивы не совсем стойки. После этого я попытаюсь описать два подхода, позволяющих строить стойкие системы. Первый основывается на чистой теории сложности. Второй, связан с понятием доказуемая стойкость и более подходящий для практических целей. Также я попытаюсь описать понятие доказуемой стойкости со случайным оракулом, которое является едва ли не центральным в современной теории стойкости.

ОБЗОРНАЯ ЧАСТЬ

1.1.Основные понятия.

Криптогра́фия (от греч. κρυπτός — скрытый и γράφω — пишу) — наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним) и аутентичности (целостности и подлинности авторства, а также невозможности отказа от авторства) информации.

Изначально криптография изучала методы шифрования информации — обратимого преобразования открытого (исходного) текста на основе секретного алгоритма и/или ключа в шифрованный текст (шифротекст). Традиционная криптография образует раздел симметричных криптосистем, в которых шифрование и дешифрование 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.

Криптография не занимается: защитой от обмана, подкупа или шантажа законных абонентов, кражи ключей и других угроз информации, возникающих в защищенных системах передачи данных.

История криптографии насчитывает около 4 тысяч лет. В качестве основного критерия периодизации криптографии возможно использовать технологические характеристики используемых методов шифрования.

Первый период (приблизительно с 3-го тысячелетия до н. э.) характеризуется господством моноалфавитных шифров (основной принцип — замена алфавита исходного текста другим алфавитом через замену букв другими буквами или символами).

Второй период (хронологические рамки — с IX века на Ближнем Востоке (Ал-Кинди) и с XV века в Европе (Леон Баттиста Альберти) — до начала XX века) ознаменовался введением в обиход полиалфавитных шифров.

Третий период (с начала и до середины XX века) характеризуется внедрением электромеханических устройств в работу шифровальщиков. При этом продолжалось использование полиалфавитных шифров.

Четвертый период — с середины до 70-х годов XX века — период перехода к математической криптографии. В работе Шеннона появляются строгие математические определения количества информации, передачи данных, энтропии, функций шифрования. Обязательным этапом создания шифра считается изучение его уязвимости к различным известным атакам — линейному и дифференциальному криптоанализам. Однако, до 1975 года

криптография оставалась «классической», или же, более корректно, криптографией с секретным ключом.

Современный период развития криптографии (с конца 1970-х годов по настоящее время) отличается зарождением и развитием нового направления — криптография с открытым ключом. Её появление знаменуется не только новыми техническими возможностями, но и сравнительно широким распространением криптографии для использования частными лицами (в предыдущие эпохи использование криптографии было исключительной прерогативой государства). Правовое регулирование использования криптографии частными лицами в разных странах сильно различается — от разрешения до полного запрета.

Современная криптография образует отдельное научное направление на стыке математики и информатики — работы в этой области публикуются в научных журналах, организуются регулярные конференции. Практическое применение криптографии стало неотъемлемой частью жизни современного общества — её используют в таких отраслях как электронная коммерция, электронный документооборот (включая цифровые подписи), телекоммуникации и других.

1.2.Симметричная криптография

Симметри́чные криптосисте́мы (также симметричное шифрование, симметричные шифры) — способ шифрования, в котором для (за)шифрования и расшифровывания применяется один и тот же криптографический ключ. До изобретения схемы асимметричного шифрования единственным существовавшим способом являлось симметричное шифрование. Ключ алгоритма должен сохраняться в секрете обеими сторонами. Алгоритм шифрования выбирается сторонами до начала обмена сообщениями.

Симметричные криптографические алгоритмы шифрования (Симметричные криптосистемы) - способ шифрования, в котором для шифрования и расшифрования применяется один и тот же криптографический ключ. Ключ алгоритма должен сохраняться в секрете обеими сторонами.

Алгоритмы шифрования и дешифрования данных широко применяются в компьютерной технике в системах сокрытия конфиденциальной и коммерческой информации от злонамеренного использования сторонними лицами. Главным принципом в них является условие, что передатчик и приемник заранее знают алгоритм шифрования, а также ключ к сообщению, без которых информация представляет собой всего лишь набор символов, не имеющих смысла.

Классическим примером таких алгоритмов являются симметричные криптографические алгоритмы, перечисленные ниже:

1. Простая подстановка
2. Одиночная перестановка по ключу
3. Двойная перестановка

Простая перестановка

Простая перестановка без ключа - один из самых простых методов шифрования. Сообщение записывается в таблицу по столбцам. После того, как открытый текст записан колонками, для образования шифровки он считывается по строкам. Для использования этого шифра отправителю и получателю нужно договориться об общем ключе в виде размера таблицы. Объединение букв в группы не входит в ключ шифра и используется лишь для удобства записи несмыслового текста.

Одиночная перестановка по ключу

Более практический метод шифрования, называемый одиночной перестановкой по ключу очень похож на предыдущий. Он отличается лишь тем, что колонки таблицы переставляются по ключевому слову, фразе или набору чисел длиной в строку таблицы.

Двойная перестановка

Для дополнительной скрытности можно повторно шифровать сообщение, которое уже было зашифровано. Этот способ известен под названием двойная перестановка. Для этого размер второй таблицы подбирают так, чтобы длины ее строк и столбцов были другие, чем в первой таблице. Лучше всего, если они будут взаимно простыми. Кроме того, в первой таблице можно переставлять столбцы, а во второй строки. Наконец, можно заполнять таблицу зигзагом, змейкой, по спирали или каким-то другим способом. Такие способы заполнения таблицы если и не усиливают стойкость шифра, то делают процесс шифрования гораздо более занимательным.

В настоящее время симметричные шифры делятся на:

-блочные шифры. Обрабатывают информацию блоками определённой длины (обычно 64, 128 бит), применяя к блоку ключ в установленном порядке, как правило, несколькими циклами перемешивания и подстановки, называемыми раундами. Результатом повторения раундов является лавинный эффект — нарастающая потеря соответствия битов между блоками открытых и зашифрованных данных;

-поточные шифры, в которых шифрование проводится над каждым битом либо байтом исходного (открытого) текста с использованием гаммирования. Поточный шифр может быть легко создан на основе блочного, запущенного в специальном режиме.

Большинство симметричных шифров используют сложную комбинацию большого количества подстановок и перестановок. Многие такие шифры исполняются в несколько (иногда до 80) проходов, используя на каждом проходе «ключ прохода». Множество «ключей прохода» для всех проходов называется «расписанием ключей» (key schedule). Как правило, оно создается из ключа выполнением над ним неких операций, в том числе перестановок и подстановок.

Типичным способом построения алгоритмов симметричного шифрования является сеть Фейстеля. Алгоритм строит схему шифрования на основе функции F(D, K), где D — порция данных, размером вдвое меньше блока шифрования, а K — «ключ прохода» для данного прохода. От функции не требуется обратимость — обратная ей функция может быть неизвестна. Достоинства сети Фейстеля — почти полное совпадение дешифровки с шифрованием (единственное отличие — обратный порядок «ключей прохода» в расписании), что сильно облегчает аппаратную реализацию.

Существует множество (не менее двух десятков) алгоритмов симметричных шифров, существенными параметрами которых являются:
-стойкость
-длина ключа
-число раундов
-длина обрабатываемого блока
-сложность аппаратной/программной реализации
-сложность преобразования.

1.3.Асимметричная криптография

Криптографическая система с открытым ключом (или Асимметричное шифрование, Асимметричный шифр) — система шифрования и/или электронной цифровой подписи (ЭЦП), при которой открытый ключ передаётся по открытому (то есть незащищённому, доступному для наблюдения) каналу, и используется для проверки ЭЦП и для шифрования сообщения. Для генерации ЭЦП и для расшифрования сообщения используется секретный ключ. Криптографические системы с открытым ключом в настоящее время широко применяются в различных сетевых протоколах, в частности, в протоколах TLS и его предшественнике SSL (лежащих в основе HTTPS), в SSH. Также используется в PGP, S/MIME.

Идея криптографии с открытым ключом очень тесно связана с идеей односторонних функций, то есть таких функций f(x), что по известному x

довольно просто найти значение f(x), тогда как определение x из f(x) сложно в смысле теории.

Но сама односторонняя функция бесполезна в применении: ею можно зашифровать сообщение, но расшифровать нельзя. Поэтому криптография с открытым ключом использует односторонние функции с лазейкой. Лазейка — это некий секрет, который помогает расшифровать. То есть существует такой y, что зная f(x), можно вычислить x. К примеру, если разобрать часы на множество составных частей, то очень сложно собрать вновь работающие часы. Но если есть инструкция по сборке (лазейка), то можно легко решить эту проблему.

Начало асимметричным шифрам было положено в работе «Новые направления в современной криптографии» Уитфилда Диффи и Мартина Хеллмана, опубликованной в 1976 году. Находясь под влиянием работы Ральфа Меркле (Ralph Merkle) о распространении открытого ключа, они предложили метод получения секретных ключей, используя открытый канал. Этот метод экспоненциального обмена ключей, который стал известен как обмен ключами Диффи-Хеллмана, был первым опубликованным практичным методом для установления разделения секретного ключа между заверенными пользователями канала. В 2002 году Хеллман предложил называть данный алгоритм «Диффи — Хеллмана — Меркле», признавая вклад Меркле в изобретение криптографии с открытым ключом. Эта же схема была разработана Малькольмом Вильямсоном в 1970-х, но держалась в секрете до 1997 года. Метод Меркле по распространению открытого ключа был изобретён в 1974 году и опубликован в 1978, его также называют загадкой Меркле.

В 1977 году учёными Рональдом Ривестом (Ronald Linn Rivest), Ади Шамиром (Adi Shamir) и Леонардом Адлеманом (Leonard Adleman) из Массачусетского Технологического Института (MIT) был разработан алгоритм шифрования, основанный на проблеме о разложении на множители. Система была названа по первым буквам их фамилий. Эта же система была изобретена Клиффордом Коксом (Clifford Cocks) в 1973 году, работавшим в центре правительственной связи (GCHQ). Но эта работа хранилась лишь во внутренних документах центра, поэтому о её существовании было не известно до 1977 года. RSA стал первым алгоритмом, пригодным и для шифрования, и для цифровой подписи.

Вообще, в основу известных асимметричных криптосистем кладётся одна из сложных математических проблем, которая позволяет строить односторонние функции и функции-лазейки. Например, криптосистемы

Меркля — Хеллмана и Хора — Ривеста опираются на так называемую задачу об укладке рюкзака.

Основные принципы построения криптосистем

1. Начинаем с трудной задачи P. Она должна решаться сложно в смысле теории: не должно быть алгоритма, с помощью которого можно было бы перебрать все варианты решения задачи P за полиномиальное время относительно размера задачи. Более правильно сказать: не должно быть известного полиномиального алгоритма, решающего данную задачу — так как ни для одной задачи ещё пока не доказано, что для неё подходящего алгоритма нет в принципе.
2. Можно выделить легкую подзадачу P' из P. Она должна решаться за полиномиальное время, лучше, если за линейное.
3. «Перетасовываем и взбалтываем» P', чтобы получить задачу P", совершенно не похожую на первоначальную. Задача P", по крайней мере, должна выглядеть как оригинальная труднорешаемая задача P.
4. P" открывается с описанием, как она может быть использована в роли ключа зашифрования. Как из P" получить P', держится в секрете как секретная лазейка.
5. Криптосистема организована так, что алгоритмы расшифрования для легального пользователя и криптоаналитика существенно различны. В то время как второй решает P" задачу, первый использует секретную лазейку и решает P' задачу.

Применение

Алгоритмы криптосистемы с открытым ключом можно использовать

- Как самостоятельные средства для защиты передаваемой и хранимой информации
- Как средства распределения ключей. Обычно с помощью алгоритмов криптосистем с открытым ключом распределяют ключи, малые по объёму. А саму передачу больших информационных потоков осуществляют с помощью других алгоритмов.
- Как средства аутентификации пользователей.

Преимущества

- Преимущество асимметричных шифров перед симметричными шифрами состоит в отсутствии необходимости предварительной передачи секретного ключа по надёжному каналу.
- В симметричной криптографии ключ держится в секрете для обеих сторон, а в асимметричной криптосистеме только один секретный.

- При симметричном шифровании необходимо обновлять ключ после каждого факта передачи, тогда как в асимметричных криптосистемах пару (E,D) можно не менять значительное время.
- В больших сетях число ключей в асимметричной криптосистеме значительно меньше, чем в симметричной.

Недостатки

- Преимущество алгоритма симметричного шифрования над несимметричным заключается в том, что в первый относительно легко внести изменения.
- Несимметричные алгоритмы используют более длинные ключи, чем симметричные.
- Процесс шифрования- расшифрования с использованием пары ключей проходит на два-три порядка медленнее, чем шифрование-расшифрование того же текста симметричным алгоритмом.
- В чистом виде асимметричные криптосистемы требуют существенно больших вычислительных ресурсов, потому на практике используются в сочетании с другими алгоритмами.

ОСНОВНАЯ ЧАСТЬ

1.Определения стойкости

1.1.Стойкость шифрования

До сих пор мы не исследовали детально вопроса о внедрении криптографических примитивов в реальные протоколы и системы. При проектировании криптографических систем нам нужно иметь абсолютно точное представление о том, какие именно свойства стойкости обеспечивает тот или иной примитив и как защитить разрабатываемые схемы от всех возможных атак.

По существу, криптографические примитивы напоминают собой или функцию RSA, или потенцирование в кольце вычетов. Обе эти функции являются односторонними, причем функция *RSA* — функция с секретом. При использовании примитивов в реальной схеме необходимо быть очень внимательным. Например, функция *RSA* полностью детерминирована, и при ее наивном использовании будут получаться тождественные шифротексты при шифровании тем же ключом одного сообщения дважды. Как мы убедились в главе 1 это плохое свойство. Но есть и более веские причины избегать шифрующих алгоритмов с детерминированным открытым ключом. Чтобы понять, почему нам следует разобраться сначала в том, что означает стойкость криптосистемы. Начать надо с определения

- цели нападающего,
- соответствующих типов атак,
- вычислительной модели.

Мы будем обсуждать эти понятия в контексте криптосистем с открытым ключом, но аналогичные концепции есть и в симметричном шифровании.

1.2.Понятия стойкости

Нам надо разобраться, по существу, с тремя видами стойкости:

- теоретико-информационная или абсолютная стойкость;
- семантическая стойкость;
- полиномиальная стойкость.

Обсудим каждый из них поочередно. Эти понятия существенно более сильные, чем раскрытие секретного ключа или дешифрование криптограммы, о которых мы говорили раньше.

Теоретико-информационная стойкость. Напомним, что схема называется абсолютно стойкой (обладает теоретико-информационной стойкостью), если нападающий с неограниченными вычислительными возможностями не сможет извлечь никакой информации об открытом тексте, используя данный шифротекст. Теорема Шеннона фактически утверждает, что ключ такой системы равен по длине самому сообщению, причем никакой ключ нельзя использовать дважды.

Очевидно, абсолютно стойкую систему невозможно создать в рамках модели шифрования с открытым ключом, поскольку шифрующий (открытый) ключ в такой модели должен использоваться долго, причем его длина мала по сравнению со средним сообщением.

Семантическая стойкость. Семантическая стойкость похожа на теоретико-информационную, но здесь предполагается, что атакующий обладает полиномиальными вычислительными возможностями. Говоря формально, при любом распределении вероятностей на пространстве сообщений, все что нападающий может вычислить за полиномиальное время о данном открытом тексте по данному шифротексту, он может узнать и без шифротекста. Другими словами, обладание шифротекстом никак не помогает извлечь информацию о сообщении.

Приведем упрощенное формальное определение. Предположим, что мы хотим извлечь однобитовую информацию из пространства сообщений, т, е, значение фиксированной функции

$$g : M \longrightarrow \{0,1\}.$$

Допустим, что для любого m из пространства сообщений

$$P(g(m) = 1) = P(g(m) = 0) = \frac{1}{2},$$

где $P(g(m) = a)$ — вероятность события $g(m) = a$. Кроме того, будем считать, что открытые тексты и шифротексты имеют постоянную длину (в частности, количество знаков в шифротексте не даст никакой информация о подлежащем открытом тексте).

Под противником мы будем понимать алгоритм S, который по открытому ключу *Y* и шифротексту C, полученному с помощью секретного ключа *x*, соответствующего *Y*, пытается определить значение функции *g* от открытого текста *m*, ассоциированного с шифротекстом C. Таким образом,

выходные данные алгоритма S состоят из единственного бита, равного значению *g(m).*

Будем считать нападение успешным, если вероятность корректного определения значения *g(m)* больше половины. Ясно, что противник может угадать ответ с вероятностью 1/2 и не глядя на шифротекст. Поэтому успешным атакующим можно назвать лишь тот алгоритм, вероятность правильного ответа у которого увеличивается после изучения шифротекста. В связи с этим определим выигрыш противника по формуле:

$$\mathrm{Adv}_S = \left| P(S(C, Y) = g(d_x(C))) - \frac{1}{2} \right|,$$

где d_x обозначает расшифровывающую функцию с секретным ключом *x,* ассоциированным с открытым ключом У. В этих терминах схема называется семантически стойкой, если

$$\mathrm{Adv}_S \leqslant \frac{1}{p(k)},$$

для всех противников S, всех функций *g* и всех многочленов *p(k)* с достаточно большим параметром стойкости *k*.

Полиномиальная стойкость. Неприятность, связанная с семантической стойкостью, заключается в том, что очень трудно проверить, обладает ли конкретная схема этим свойством. Полиномиальная стойкость, иногда называемая *неразличимостью шифрования,* является несравненно более просто проверяемым свойством. К счастью, мы покажем, что система, обладающая полиномиальной стойкостью, семантически стойка. Следовательно, чтобы проверить криптосистему на семантическую стойкость, достаточно продемонстрировать ее полиномиальную криптостойкость.

Будем говорить, что система обладает свойством неразличимости шифрования, или полиномиальной стойкостью, если никакой атакующий не сможет победить в следующей игре с вероятностью, превышающей 50%. Нападающий *A*, которому дана шифрующая функция f_Y, соответствующая открытому ключу *Y,* совершает два шага:

Поиск. На этой стадии нападающий генерирует два открытых текста M_0 и M_1.

Гипотеза. Нападающему дают шифротекст *Сь,* соответствующий одному из выбранных сообщений M_b, причем скрывают от него, какому именно. Цель нападающего угадать значение b с вероятностью, большей 50%.

Заметим, что отсюда следует вероятностная природа полиномиально стойкой шифрующей функции, поскольку в противном случае атакующий может вычислить

$$C_1 = f_Y(M_1)$$

и сравнить полученное значение с C_b,. Следовательно, выигрыш нападающего равен

$$\mathrm{Adv}_A = \left| P(A(\textit{гипотеза}, C_b, Y, M_0, M_1) = b) - \frac{1}{2} \right|,$$

т. к. гипотеза нападающего состоит в выборе одного из возможных значений *b:* 0 или 1. Схема называется полиномиально стойкой, если

$$\mathrm{Adv}_A \leqslant \frac{1}{p(k)}$$

при любом противнике *A*, произвольном многочлене *p* и достаточно большом *k.*

1.3.Виды атак

Теперь нам следует ввести различные модели атак. Существует три основных модели:

- *пассивная атака,* или атака с выбором открытого текста, *CPA* (от англ. chosen plaintext attack);
- *атака с выбором шифротекста, CCA1* (от англ. chosen cipher-text attack);
- *адаптивная атака с выбором шифротекста, CCA2* (от англ. adaptive chosen ciphertext attack).

Пассивная атака. Пассивная атака — очень слабая форма атаки. Нападающей Еве разрешается просматривать различные шифрованные сообщения. Кроме того, ей открыт доступ к «черному ящику», осуществляющему шифрование, но не расшифровывание. Таким образом, пассивная атака — простейшая модель нападения на криптосистему с открытым ключом, поскольку она предусматривает открытый доступ к функции шифрования.

Атака с выбором шифротекста. Атака с выбором шифротекста *(CCA1),* которую часто называют атакой обеденного перерыва, представляет собой несколько более сильную модель нападения. Здесь Еве предоставлен доступ к расшифровывающему «черному ящику». Она может предлагать ящику расшифровать полиномиальное число шифротекстов по ее выбору во время обеденного перерыва, пока законный пользователь аппарата подкрепляет свои силы. После этого спустя некоторое время, Еве дают тестовый шифротекст и предлагают дешифровать его или восстановить какую-нибудь информацию о подлежащем открытом тексте собственными силами, не прибегая к помощи «черного ящика».

Адаптивная атака с выбором шифротекста. Адаптивная атака с выбором шифротекста *(CCA2)* — наиболее сильная форма атаки. В этом случае Еве можно пользоваться расшифровывающим прибором для расшифровывания

любого шифротекста по ее выбору, кроме тестового. Принято считать, что любая вновь предлагаемая криптосистема открытым ключом должна обладать полиномиальной стойкостью по отношению к адаптивной атаке с выбором шифротекста.

Следовательно, в нашей игре с полиномиальной стойкостью нападающий имеет право обращаться к расшифровывающему прибору на любой стадии игры. При этом ему запрещается лишь подавать на вход прибора тестовый шифротекст C_b, иначе игра станет бессмысленной.

Сформулируем стандартное определение стойкости схемы шифрования с открытым ключом. Аналогичное определение применяется к симметричным системам шифрования.

Определение Алгоритм шифрования с открытым ключом называют *стойким,* если он семантически стоек в отношении адаптивной атаки с выбором шифротекста.

Однако легче проверить, что данная криптосистема удовлетворяет следующему определению.

Определение Алгоритм шифрования с открытым ключом называют *стойким,* если он полиномиально стоек в отношении адаптивной атаки с выбором шифротекста.

На самом деле эти определения связаны между собой. Докажем, например, следующий важный результат.

Теорема 2. При пассивном нападении полиномиально стойкая система обязана быть семантически стойкой.

Доказательство. Применим метод «от противного». Предположим, что существует алгоритм шифрования, не являющийся семантически стойким, т. е. найдется такой алгоритм S, что

$$\mathrm{Adv}_S > \frac{1}{p(k)}$$

для некоторого многочлена p и достаточно большого числа k. Покажем, что этот алгоритм не будет и полиномиально стойким. Для этого мы построим атаку A на полиномиальную стойкость схемы шифрования, использующую алгоритм S, на падающий на семантическую стойкость, как оракул. На стадии *поиска* в атаке A генерируются два сообщения M_0 и M_1, для которых

$$g(M_0) \neq g(M_1),$$

что можно легко сделать, опираясь на упрощенное формальное определение семантической стойкости, поскольку значения функции g от сообщений распределены равномерно.

Противнику *A* выдается шифротекст C_b, соответствующий одному из выбранных сообщений, и предлагается определить b, На этапе *гипотеза* нападающий передает шифротекст C_b оракулу S, который находит наилучшую гипотезу для значения $g(m_b)$. Алгоритм *A* сравнивает эту гипотезу с $g\{M_0\}$ и $g(M_1)$ и определяет значение b.
Ясно, что если S достигает успеха при взломе семантической стойкости схемы, то *A* удается взломать ее полиномиальную стойкость. Поскольку

$$P(A(гипотеза, C_b, Y, M_0, M_1) = b) = P(S(C, Y) = g(d_x(C))),$$

имеет место неравенство

$$\mathrm{Adv}_A = \mathrm{Adv}_S > \frac{1}{p(k)}.$$

Следовательно, полиномиальная стойкость влечет семантическую.

Заметим, что если брать более сложное определение семантической стойкости, то можно показать, что при пассивной атаке понятия семантической и полиномиальной стойкости эквивалентны.

1.4.Другие концепции стойкости

Жесткость. Говорят, что схема шифрования является жесткой, если по данной паре (открытый текст, шифротекст) (m, *C)* невозможно найти корректный шифротекст C` для «связанного» сообщения *M`* не зная *M*. Заметим, что смысл слова «связанный» здесь не совсем ясен и может варьироваться в зависимости от целей. Концепция жесткости приобретает важное значение в свете результата, который мы докажем неформально, основываясь на нашем не вполне строгом определении жесткости.

Лемма. Нежесткая схема шифрования нестойка в отношении адаптивной атаки с выбором шифротекста.
Доказательство. Пусть наша схема не является жесткой. Заменим данный нам шифротекст C_b на связанный с ним C'_b. При этом соответствующие открытые тексты M_b и M'_b тоже должны быть связанными друг с другом. После этого нападающий может потребовать от оракула дешифровать C'_b и раскрыть M' что было разрешено в нашей предыдущей игре. Затем, зная M_b' нападающий может узнать и M_b.

Позже мы увидим, что практически все схемы шифрования с открытым ключом, с которыми мы уже встречались, лишены свойства жесткости.

Известно, однако, что жесткая схема в отношении атаки *CCA2* является и полиномиально стойкой относительно нее, и наоборот.

Текстозависимостъ. Свойством текстозависимости обладают наиболее стойкие схемы. Схема называется *текстозависимой,* если с вычислительной точки зрения практически невозможно построить корректный шифротекст, не имея под рукой соответствующего открытого текста. Следовательно, текстозависимость схемы влечет, что атака *CCA* на нее не может достичь успеха. Поскольку создание шифротекста требует информации об открытом тексте. Вам просто нечего будет подать на вход расшифровывающего оракула. Понятие текстозависимости было определено в контексте модели *случайного оракула.* В этой модели предполагается, что существуют идеализированные хэш-функции, которые являются односторонними и

- вычисляют значения за полиномиальное время;
- наблюдатель не может их отличить от любых других случайных функций.

К модели случайного оракула прибегают в некоторых доказательствах стойкости. Эти доказательства ничего нам не говорят о реальной стойкости схемы, которую мы рассматриваем, но показывают, что любая реальная атака на нее должна использовать фактическое определение встроенной в схему хэш-функции. Обычная практика заключается в том, что мы доказываем стойкость протокола с помощью модели случайного оракула, а затем превращаем его в реальный протокол, заменяя случайного оракула на хэш-функцию типа *SHA-1* или *MD5.*

1.5.Стойкость актуальных алгоритмов шифрования

В этом параграфе мы покажем, что ни *RSA,* ни Эль-Гамаль, алгоритмы шифрования с открытым ключом, не удовлетворяют строгим требованиям семантической стойкости в отношении адаптивной атаки с выбором шифротекста. Это удивительно, поскольку мы знаем, что *RSA* — наиболее используемый и важный алгоритм шифрования. Однако мы еще не показывали, как именно *RSA* применяется в реальных задачах криптографии. Поэтому, сосредоточим внимание на инженерном воплощении алгоритмов шифрования.

Для облегчения изложения будем учитывать эквивалентность понятий семантической стойкости и неразличимости шифрования (полиномиальной стойкости).

1.5.1.RSA

Лемма *RSA* не обладает полиномиальной стойкостью.

Доказательство. Предположим, атакующий знает, что пользователь зашифровал только одно из двух сообщений,

$$M_1 \quad \text{или} \quad M_2.$$

Они могут означать: покупаю или продаю, да или нет, и т. д. Предполагается, что атакующему известен открытый ключ пользователя, а именно N и E. Получив шифротекст C, нападающий стремится определить, с каким из двух сообщений M_1 или M_2 совпадает подлежащий открытый текст m. Для решения этой задачи ему достаточно лишь вычислить

$$C' = M_1^E \pmod N.$$

Тогда

- если $C' = C$, то атакующий получает равенство $m = M_1$
- если $C' \neq C$, то можно заключить, что $m = M_2$.

Очевидно, проблема связана с доступом противника к шифрующей функции, ведь как-никак, это схема с открытым ключом.

Предположим теперь, что процедура расшифровывания не инъективна, т. е. каждый открытый текст может соответствовать большому числу шифротекстов, причем то, какой именно шифротекст получится из данного сообщения, решается шифрующей функцией в процессе работы. Тогда описанная выше атака будет неудачной. Другими словами, в целях обеспечения стойкости природа алгоритма шифрования должна быть вероятностной, а не детерминированной. Позже мы увидим вариант RSA-системы, обладающий таким свойством. Поэтому описанная выше атака к нему не применима. Однако детерминированная функция шифрования не единственная проблема *RSA*.

По существу, *RSA* - нежесткая система ввиду свойства гомоморфности.

Определение (Свойство гомоморфности.) Имея зашифрованные сообщения m_1 и m_2, можно определить шифротекст, соответствующий произведению $m_1 * m_2$, не зная самих сообщений.

Тот факт, что *RSA* обладает свойством гомоморфности, вытекает из уравнения:

$$(m_1 \cdot m_2)^E \pmod N = \left(\left(m_1^E \pmod N\right) \cdot \left(m_2^E \pmod N\right)\right) \pmod N.$$

Опираясь на это свойство, можно показать, что *RSA* не стойка в отношении адаптивной атаки с выбором шифротекста.

Лемма, *RSA* не является CCA2-стойкой.

Доказательство. Предположим, что Ева намерена взломать шифротекст

$$C = m^E \pmod N.$$

Она создает «связанный» шифротекст $C' = 2^E C$ и требует от своего оракула расшифровать C′. При этом получается некоторое сообщение M', после чего достаточно провести следующие вычисления:

$$\frac{M'}{2} = \frac{C'^d}{2} = \frac{(2^E C)^d}{2} = \frac{2^{Ed} C^d}{2} = \frac{2m}{2} = m.$$

1.5.2.Эль-Гамаль

Напомним, что проблема выбора Диффи-Хеллмана (известная как *ПВДХ)* состоит в определении истинности равенства

$$x \cdot y = z \pmod{\#\langle G \rangle}$$

по данным элементам G^x, G^y и G^z из циклической группы (G).

Лемма. Если *ПВДХ* — трудная задача в группе (G), то криптосистема Эль-Гамаль обладает полиномиальной стойкостью относительно пассивного нападения.

Доказательство. Допустим, что у нас есть полиномиальный алгоритм *A*, нарушающий полиномиальную стойкость системы Эль-Гамаль. Опираясь на алгоритм *A*, найдем решение задачи *ПВДХ.* Сейчас, при более ограничительном определением стойкости, а именно, полиномиальной стойкости, мы можем только свести задачу взлома системы к решению (предположительно) более легкой проблемы. Напомним, что алгоритм *A* проходит две стадии:

- Стадия *поиска,* имеющая на входе открытый ключ, выдающая два сообщения и некоторую дополнительную информацию.

- Стадия *гипотезы,* на вход которой подается шифротекст, открытый ключ, два сообщения и некая дополнительная информация. При этом требуется узнать, какое из сообщений соответствует данному шифротексту.

Кроме того, важно помнить, что шифротекст в Эль-Гамаль имеет вид

$$(G^k, m \cdot H^k),$$

где *к* — эфемерный, меняющийся с каждым сообщением, секретный ключ, а *H* открытый ключ.

Наш алгоритм, решающий *ПВДХ* работает следующим образом:

1. Входными данными служат G^x, G^y и G^z.
2. Положим $H = G^x$.
3. $(M_0, M_1, S) = A(поиск, H)$.
4. Определим $C_1 = G^y$.
5. Выберем случайным образом $B \in \{0,1\}$.

6. Положим $C_2 = M_B * G^z$.
7. $B' = A(гипотеза, (C_1, C_2), H, M_0, M_1, S)$,
8. Если $B = B'$ то напечатать ИСТИНА.
9. В противном случае напечатать ЛОЖЬ.

Чтобы показать, что этот алгоритм действительно решает *ПВДХ,* приведем следующие аргументы.

- Если $z = x * y$, то зашифрованные данные, поданные на вход стадии *гипотезы* алгоритма *A,* будут представлять сообщение M_B. Следовательно, если алгоритм *A* может взломать семантическую стойкость системы Эль-Гамаль, то выходные данные B' будут верными и алгоритм напечатает ИСТИНА.
- Предположим теперь, что $z \neq x * y$. В этом случае на вход стадии *гипотеза* почти наверняка попадут неверные данные, т. Е. шифротекст не будет соответствовать ни M_0 ни M_1. Следовательно, то, что получится на выходе этой стадии, т. Е. B' никак не будет зависеть от *B.* Значит, алгоритм после окончания работы напечатает ИСТИНА или ЛОЖЬ с одинаковой вероятностью.

Повторяя алгоритм несколько раз, мы получим вероятностный полиномиальный алгоритм решения *ПВДХ.* Но мы предполагаем, что такого алгоритма не существует. Поэтому и алгоритм *A* – мистификация. Таким образом, предположение о сложности *ПВДХ* влечет невозможность успешных атак на полиномиальную стойкость Эль-Гамаль с выбором открытого текста.

Однако, несмотря на семантическую стойкость в отношении атак с выбором открытого текста, Эль-Гамаль — нежесткая криптосистема.

Лемма. Эль-Гамаль — нежесткая криптосистема.

Доказательство. Предположим, что Ева видит шифротекст

$$(C_1, C_2) = (G^k, m \cdot H^k).$$

Тогда она может создать удовлетворительный шифротекст, соответствующий сообщению *2*m*, не зная ни *m*, ни эфемерного ключа *k*, ни секретного ключа *x*. Искомый шифротекст получается простым умножением второй его половины на 2:

$$(C_1, 2 \cdot C_2) = (G^k, 2 \cdot m \cdot H^k).$$

Опираясь на это отсутствие жесткости, можно показать, как и в случае RSA, что Эль-Гамаль не стоек в отношении адаптивной атаки с выбором шифротекста.

Лемма. Эль-Гамаль не является *CCA2*-стойкой схемой.

Доказательство. Предположим, что Ева намерена взломать шифрованное сообщение

$$C = (C_1, C_2) = (G^k, m \cdot H^k).$$

Тогда она создает связанное сообщение

$$C' = (C_1, 2 \cdot C_2)$$

и требует у своего расшифровывающего оракула дешифровать C' и получить соответствующий открытый текст M'. После этого Ева вычисляет

$$\frac{M'}{2} = \frac{2C_2C_1^{-x}}{2} = \frac{2mH^kG^{-xk}}{2} = \\ = \frac{2mG^{xk}G^{-xk}}{2} = \frac{2m}{2} = m.$$

1.6.Семантически стойкие системы

Мы уже видели, что *RSA* не является семантически стойкой даже в отношении пассивной атаки. Хорошо бы теперь привести пример семантически стойкой системы, основанной на чем-то близком к задаче факторизации целых чисел. Первая такая схема принадлежит Гольдвассеру и Микали, хотя ее и не используют в практических приложениях. Стойкость этой схемы основывается на сложности проблемы квадратичных вычетов. Действительно, очень трудно определить, является ли данное число *a* квадратичным вычетом по модулю составного *N*, если не знать разложения последнего на простые множители.
Напомним, что множество квадратов в группе (Z/NZ) обозначается через

$$Q_N = \left\{x^2 \pmod N \middle| \; x \in (\mathbb{Z}/N\mathbb{Z})^*\right\},$$

а символом J_N обозначают множество элементов, чей символ Якоби равен плюс единице, т. е.

$$J_N = \left\{x \in (\mathbb{Z}/N\mathbb{Z})^* \middle| \; \left(\frac{x}{N}\right) = 1\right\}.$$

Множество псевдо-квадратов совпадает с разностью $J_N \setminus Q_N$. Для RSA-подобных модулей $N = p*q$ число элементов в J_N равно *(p-1)*(q-1)/2,* в то время как множество Q_N насчитывает *(p-1)*(q-1)/4* элементов. Проблема квадратичных вычетов заключается в следующем: пусть дан $X \in J_N$, лежит ли он в Q_N? Ответить на вопрос очень трудно, в то время как убедиться в том, что $X \in J_N$ довольно легко. Изложим систему шифрования Гольдвассера-Микали.

1.Генерирование ключей. В качестве секретного ключа выбираются два простых числа p и q и вычисляется открытый модуль $N = p*q$. Кроме того, открытый ключ содержит целое число $Y \in J_N \setminus Q_N$.

Чтобы найти Y, сначала определяются элементы $y_p \in F_p$ и $y_q \in F_q$ удовлетворяющие соотношени

$$\left(\frac{y_p}{p}\right) = \left(\frac{y_q}{q}\right) = -1,$$

а потом Y вычисляется по y_p и y_q с помощью китайской теоремы об остатках. Очевидно, что полученный Y не будет лежать в Q_N, но он принадлежит множеству J_N, поск

$$\left(\frac{Y}{N}\right) = \left(\frac{Y}{p}\right) \cdot \left(\frac{Y}{q}\right) = \left(\frac{y_p}{p}\right) \cdot \left(\frac{y_q}{q}\right) = (-1) \cdot (-1) = 1.$$

2.Шифрование. В системе Гольдвассера-Микали шифруется один бит информации за один раз. Для шифрования бита b поступают следующим образом:

- берут произвольный элемент $x \in (Z/NZ)$
- и вычисляют $C = Y^b x^2 \pmod N$.

Шифротекстом служит число C. Обратите внимание на то, что алгоритм весьма неэффективен, поскольку отдельный бит открытого текста занимает $\log_2 N$ бит в шифротексте.

3.Расшифровывание. Отметим, что любой шифротекст C принадлежит множеству J_N. Причем если бит b сообщения равен 0, то C будет квадратичным вычетом, в противном случае — квадратичным невычетом. Таким образом, для дешифрования сообщения необходимо уметь решать проблему квадратичных вычетов по модулю N. Законный пользователь, естественно, знает разложение N на простые множители. $\left(\frac{C}{p}\right)$ Поэтому он может вычислить символ Лежандра Если найденный символ равен +1, то С — квадратичный вычет и соответствующий бит открытого сообщения равен О. Если же символ равен — 1, то C — квадратичный невычет и соответствующий бит равен +1.

4.Доказательство стойкости. Здесь мы хотим показать, что система шифрования Гольдвассера-Микали защищена от пассивного нападения, если задача о квадратичных вычетах по модулю N является трудной.

Лемма. Если задача о квадратичных вычетах по модулю N трудная, то криптосистема Гольдвассера-Микали полиномиально стойка в отношении пассивного нападения.

Доказательство. Пусть у нас есть алгоритм A, атакующий нашу криптосистему. Покажем, что с его помощью можно решить задачу о квадратичных вычетах.

Допустим, нам дан элемент $L \in J_N$ и мы хотим определить, является ли он квадратичным вычетом, т. е. справедливо ли включение $L \in Q_N$? Поскольку наша криптосистема шифрует отдельные биты сообщения, на стадии *поиска* атакующий алгоритм A по открытому ключу (У, N) создаст два сообщения

$$M_0 = 0 \quad \text{и} \quad M_1 = 1.$$

Сформируем шифротекст

$$C = L.$$

Заметим, что если L — квадратичный вычет, то шифротекст C будет соответствовать сообщению Mo, в противном случае C корректно отражает сообщение M_1. На стадии *гипотезы* можно потребовать от алгоритма A определить, какому именно сообщению из M_i соответствует шифротекст C, и по ответу на этот вопрос мы легко можем установить, принадлежит элемент L множеству Q_N или нет.

5.Адаптивные противники. Заметим, что приведенные выше рассуждения ничего не говорят о том, является ли шифрующая схема Гольдвассера-Микали стойкой в отношении адаптивных нападений. Сейчас мы покажем, что она не обладает таким свойством.

Лемма. Схема шифрования Гольдвассера-Микали нестойка по отношению к адаптивной атаке с выбором шифротекста.

Доказательство. Пусть C — тестовый шифротекст, и мы хотим определить тот бит b, чьей шифрованной версией он является. Напомним, что

$$C=Y^b x^2 \ (mod\ N).$$

Правила игры запрещают нам обращаться к расшифровывающему оракулу для дешифрования C, но мы можем потребовать от него расшифровать любой другой шифротекст по нашему выбору. Создадим новый шифротекст

$$C' = C \cdot Z^2 \pmod N$$

при некотором произвольном $Z \neq 0$. Легко видеть, что C - шифротекст того же самого бита b, Значит, обращаясь к оракулу на законном основании, можно расшифровать C и получить тот же открытый текст, что и для C.

1.7.Стойкость подписей

Существует много возможных понятий стойкости схем подписи. Как и в случае со стандартной собственноручной подписью, мы заинтересованы в том, чтобы подпись нельзя было подделать. Назовем три основных типа фальсификации подписи.

- *Полное раскрытие.* В этой ситуации злоумышленник может ставить подписи так же, как законный обладатель ключа. Такая ситуация, фактически, возникает при взломе секретного ключа и соответствует аналогичному взлому алгоритма шифрования.
- *Селективная фальсификация.* Здесь нападающий способен подделать подпись на единственном сообщении по его выбору.

Это аналогично ситуации, когда атакующий алгоритм шифрования может дешифровать одно из сообщений, но не взломать секретный ключ.

- *Экзистенциальная фальсификация.* Противник способен подделать подпись на единственном сообщении, которое может быть только случайной строкой битов. Такую ситуацию можно отождествить с семантической стойкостью криптосистемы.

На практике нас обычно волнуют селективные фальсификации. Поэтому мы стремимся к тому, чтобы подпись была стойкой именно в отношении селективной подделки. Однако нам неизвестно, как именно будет использоваться схема подписи. Не исключено, что она будет работать в протоколах запрос-ответ, где случайная строка битов подписывается различными сторонами. Следовательно, разумно настаивать на том, чтобы любая схема подписи была стойкой в отношении экзистенциальной фальсификации.

В соответствии с типом фальсификации возникают типы атак. Наиболее слабая атака производится пассивным противником, который, имея открытый ключ, пытается осуществить селективную или экзистенциальную подделку. Более сильная атака возникает со стороны адаптивного активного противника, имеющего доступ к оракулу, производящему законные подписи для данного открытого ключа. Цель активного противника самостоятельно подписать данное сообщение, не обращаясь для этого конкретного действия к подписывающему оракулу.

Определение. Схему подписи считают стойкой, если адаптивный противник не способен осуществить ее экзистенциальную фальсификацию.

Уже отмечалось, что критическим моментом в схемах подписи является использование хэш-функций. Это можно увидеть вновь, рассматривая грубый RSA-алгоритм подписи, определенный как

$$S = M^d \ (mod\ N).$$

Осуществить здесь экзистенциальную фальсификацию с помощью пассивной атаки — дело тривиальное. Противник выбирает случайное значение S и вычисляет

$$M = S^E \ (mod\ N).$$

Таким образом атакующий получает подпись S на сообщении M.

С другой стороны, активный противник легко сделает селективную фальсификацию в этой схеме. Допустим, он намерен поставить «законную» подпись S на сообщении M. С этой целью противник генерирует случайное число $M_1 \in (Z/NZ)$ и находит $M_2 = \frac{M}{M_1}$.

Затем атакующий требует от оракула подписать сообщения M_1 и M_2. В результате он станет обладателем подписей S_1 и S_2, причем

$$S_i = M_i^d \ (mod\ N).$$

Наконец, вычисляя произведение

$$S = S_1 * S_2 (mod\ N),$$

атакующий обретает долгожданную подпись, поскольку

$$S = S_1 \cdot S_2 \pmod N = M_1^d \cdot M_2^d \pmod N =$$
$$= (M_1 \cdot M_2)^d \pmod N = M^d \pmod N.$$

2. ТЕОРЕТИЧЕСКАЯ СЛОЖНОСТЬ

2.1. Классы полиномиальной сложности

Наиболее распространенной ошибкой при проектировании новых криптографических схем является выбор проблемы, вся трудность которой проявляется лишь в частных случаях, вместо такой задачи, которая была бы сложной в средней ситуации. Чтобы разобраться в этом замечании более подробно, нам необходимо усвоить некоторые из основных идей теории сложности.

Напомним, что *проблемой выбора* или *задачей принятия решений (ДР)* называется задача о выборе положительного или отрицательного ответа на вопрос I (который обычно называют *запросом),* закодированный каким-либо способом (например, как строка битов определенного размера n). Часто имеют в

виду некоторое множество S и задают вопрос вида: верно ли, что $I \in S$? Например:

- I - целое число, а S — множество всех простых чисел. В этой ситуации проблема выбора звучит так: является ли данное число простым или нет?

- I - граф, а S множество всех графов, которые можно раскрасить только k красками. Здесь проблема выбора сводится к выяснению: можно ли данный граф раскрасить по определенным правилам, используя ровно k красок? Напомним, что граф, состоящий из конечного множества вершин V и конечного множества ребер E, считается раскрашенным k красками, если каждая его вершина покрашена одной из k красок так, что любая пара вершин, соединяющаяся ребром, имеет разный цвет.

Коль скоро мы заговорили о проблеме выбора, полезно переформулировать стандартные вычислительные проблемы в этих терминах. В качестве примера рассмотрим важную с точки зрения криптографии задачу о рюкзаке.

Определение. (Проблема выбора в задаче о рюкзаке) Пусть дан набор из N предметов, каждый из которых имеет определенный вес W_i. Можно ли сложить некоторые из предметов в рюкзак так, чтобы общий вес рюкзака составил S? Иными словами, *можно* ли найти последовательность $\{b_i\}$, состоящую из нулей и единиц, для которой

$$S = b_1W_1 + b_2W_2 + \cdots + b_NW_N?$$

Заметим, что время, требуемое на решение задачи о рюкзаке, в наихудшем случае растет как экспонента от числа предметов N.
Как было отмечено, задача о рюкзаке является проблемой выбора. А что если нам потребуется алгоритм вычисления элементов последовательности $\{b_i\}$?

Определение. (Задача о рюкзаке.) Пусть дан набор из N предметов, каждый из которых имеет определенный вес W_i. Требуется сложить некоторые из предметов в рюкзак так, чтобы общий вес рюкзака составил S. Иными словами, *нужно* найти последовательность $\{b_i\}$, состоящую из нулей и единиц, для которой

$$S = b_1W_1 + b_2W_2 + \cdots + b_NW_N?$$

Предполагается, что если такая последовательность и существует, то только одна.

Имея оракула, решающего проблему выбора в задаче о рюкзаке, можно построить алгоритм решения задачи в классической постановке.

Соответствующий алгоритм приведен ниже, где оператор **0(W[1] ,.,.,W[n],S)** обозначает оракула, решающего проблему выбора.

```
if  (0(W[1],..., W[n],S)== False)
{ Output False;  }
T=S;
b[1]  = b[2]  =...  = b[n]  = 0.
for  (i=1;  i<=n;  i++)
{ if  (T==0)
{ Output (b[1],  . . . .  b[n]).  }
if 0(W[i+l],…,W[n],T-W[i])==True)
{ T=T-W[i]; b[i]=l;  }
}
```

Говорят, что проблема выбора $\mathcal{DP}$ принадлежит классу задач сложности $\mathcal{P}$, если существует алгоритм, отвечающий «да» на любой запрос *I* (на который должен последовать положительный ответ) за полиномиальное время. Мы измеряем время в терминах битовых операций, и нолиномиальность означает, что число битовых операций ограничено некоторой функцией, полиномиально зависящей от битового размера запроса *I.* Если ответ на запрос должен быть отрицательным, то от алгоритма вообще не требуется ответа, но если он все же последует, то должен быть верным.

Заменяя в предыдущем определении положительный ответ на отрицательный, мы получим класс задач со - $\mathcal{P}$. К нему относятся те проблемы выбора, для которых существует алгоритм, идентифицирующий запросы с подразумеваемым отрицательным ответом за полиномиальное время.

Лемма.

$$\mathcal{P} = \text{co} - \mathcal{P}$$

Доказательство. Пусть *A* — алгоритм, правильно отвечающий на запрос *I*, ответ на который должен быть положительным, за время n^c для некоторой константы с. Трансформируем его в полиномиальный алгоритм, дающий верный ответ на запрос с подразумеваемым отрицательным ответом за время $n^c + 1$. Для этого мы запустим алгоритм *A* и если он не даст ответа на запрос в течении n^c шагов, то остановим его и дадим ответ «нет».

К проблемам класса сложности $\mathcal{P}$ относятся те, для решения которых существует эффективный алгоритм. Другими словами, это задачи, в которых легко проводить вычисления. Например,

- проверка равенства $z = x \bullet y$ для данных целых чисел x. y и z принадлежит классу $\mathcal{P}$, если умножение — легкая операция;
- проверка соответствия данного шифротекста C ключу k и сообщению m в любимой Вами криптосистеме.

В последнем примере, естественно, предполагается, что алгоритмы шифрования и расшифровывания в криптосистеме осуществляются за полиномиальное время.

Проблему выбора относят к классу $\mathcal{NP}$ недетерминированного полиномиального времени, если для любого запроса с подразумеваемым положительным ответом существует свидетельство соответствующего ответа, которое может быть проверено за полиномиальное время. Если же ответ на запрос отрицателен, то алгоритм вообще может не закончиться, но если он все же остановится, то обязан дать ответ «нет». Свидетельство стоит представлять себе как доказательство принадлежности запроса I множеству S.

Примеры задач класса $\mathcal{NP}$.

- Является ли данное число N составным? Здесь в качестве свидетеля выступает любой нетривиальный делитель числа N. Проверка осуществляется за полиномиальное время, поскольку операция деления имеет полиномиальную сложность.
- Можно ли раскрасить данный граф k красками? Найти свидетеля здесь — это предъявить правильную раскраску.
- Проблема рюкзака при данном наборе весов. Свидетель правильная последовательность $\{b_i\}$.

Обратите внимание на то, что ни в одном из этих примеров не предполагается, что свидетеля можно найти за полиномиальное время. Единственное требование проверка должна проводиться за полиномиальное число операций. Очевидно, имеет место включение

$$\mathcal{P} \subset \mathcal{NP}$$

Основная проблема в области теоретической информатики это совпадают ли множества $\mathcal{P}$ и $\mathcal{NP}$? Большинство считает, что справедлива

Гипотеза.

$$\mathcal{P} \neq \mathcal{NP}.$$

Множество со - $\mathcal{NP}$ определяется аналогично классу со - $\mathcal{P}$ как совокупность задач выбора, имеющих свидетельство отрицательного ответа, которое может быть проверено за полиномиальное время. В отличие от положения дел с парой со - $\mathcal{P}$ и $\mathcal{P}$ имеет место утверждение:

Если $\mathcal{P} \neq \mathcal{NP}$, то $\mathcal{NP} \neq$ co - $\mathcal{NP}$.

Поэтому стоит считать, что $\mathcal{NP} \neq$co - $\mathcal{NP}$.

Выясним на примере, насколько малым может быть свидетель для задачи из класса $\mathcal{NP}$. Рассмотрим задачу о множителях, т.е. вопрос о наличии нетривиальных делителей у данного числа $n \in Z$. Как уже было отмечено, эта задача принадлежит классу $\mathcal{NP}$. Доказать разложимость числа можно следующими способами.

- Предъявить нетривиальный множитель. Здесь размер свидетеля равен $O(\ln n)$.
- Предъявить свидетеля Миллера-Рабина a. Согласно обобщенной гипотезе Римана справедлива оценка

$a \leq O((\ln n)^2)$.

Поэтому размер свидетеля оценивается как $O(\ln \ln n)$.

Говорят, что данная проблема выбора $\mathcal{DP}$ является $\mathcal{NP}$ -полной, если любая задача класса $\mathcal{NP}$ может быть сведена к ней за полиномиальное время. Другими словами, это условие означает, что

$$\mathcal{DP} \in \mathcal{P} \quad \Longrightarrow \quad \mathcal{P} = \mathcal{NP}.$$

В некотором смысле, $\mathcal{NP}$ -полные задачи являются самыми трудными из имеющих решение. Существует огромное число $\mathcal{NP}$ -полных задач, из которых нас интересуют всего две:

- задача о раскраске графа
- задача о рюкзаке.

Нам известно, что задача о делимости натурального числа принадлежит классу $\mathcal{NP}$, но не известно является ли она $\mathcal{NP}$ -полной. Принято считать, что все трудные задачи, на которых основываются криптографические примитивы, например, факторизация, проблема дискретного логарифмирования, и т. д., не являются $\mathcal{NP}$ -полными, хотя и лежат в классе $\mathcal{NP}$.

Отсюда можно заключить, что задача факторизации не является очень сложной проблемой по сравнению с задачами о раскраске графа или укладке рюкзака. Почему же мы не берем $\mathcal{NP}$ -полные проблемы для конструирования криптографических систем? Ведь они представляют собой класс хорошо изученных трудных задач, для которых существование эффективного решения крайне сомнительно.

Как мы увидим позже, после знакомства с системой Меркля и Хеллмана, основанной на задаче о рюкзаке, идея использования $\mathcal{NP}$-полных задач для криптографических примитивов плохо реализуется. До этого момента мы лишь упоминали, что теория сложности и, в частности, $\mathcal{NP}$-полнота имеет дело с

наисложнейшим случаем. Для криптографических же целей нам удобны задачи, которые при соответствующем выборе параметров являются трудными в среднем. Как мы позже убедимся, задача об укладке рюкзака, которую ранее предлагали использовать в криптографии, всегда имеет «среднюю» сложность, и при любом выборе параметров можно найти эффективный алгоритм, который ее решает.

2.2.Битовая стойкость

Ранее мы ввели понятие проблемы выбора как задачи с односложным ответом: «да» или «нет». Мы показали, что некоторые другие задачи, например, задача о рюкзаке, сводится к такой проблеме. Аналогичная ситуация возникает в криптографии, где хотелось бы знать, является ли задача вычисления одного бита сообщения столь же сложной, как и задача вычисления всего сообщения. Предположим, что используется RSA-функция

$$x \mapsto Y = x^E \pmod{N}.$$

Может возникнуть ситуация, когда противника волнует лишь вычисление $B = x \pmod 2$, а не полное восстановление x. Нам хотелось бы, чтобы решение этого уравнения было бы сравнимо по сложности с полным восстановлением сообщения. Другими словами, нам нужно исследовать так называемую битовую стойкость функции *RSA.*

Легко убедиться, что битовая стойкость тесно связана с семантической: если нападающий сможет определить четность открытого текста, имея только соответствующий шифротекст, то семантическая стойкость алгоритма шифрования будет взломана.

Нам понадобится следующее понятие.

Определение. Пусть $f: S \longrightarrow T$ — одностороннее отображение конечных множеств S и T, а $B : S \to \{0,1\}$ — булева функция (называемая *предикатом).* Предикат *B(x)* называется *сильным* для *f*, если по данному $x \in S$ значение *B(x)* вычисляется легко, а по *f(x)* — очень трудно.

Один из способов проверки этого свойства предиката *B* в отношении односторонней функции *f* заключается в том, чтобы предположить существование оракула, вычисляющего *B(x)* по данному элементу *f(x),* и показать, что соответствующий оракул фактически обращает функцию *f.*

Понятие *k*-битового предиката и сильного *k*-предиката определяется аналогично, только значения он принимает в строках битов длины *k*, а не 1, как это было в предыдущем определении. Покажем, что различные предикаты, которые можно определить для полезной с точки зрения криптографии односторонней функции *f*, фактически являются сильными,

2.3.Сильные предикаты для дискретных логарифмов

Пусть A - конечная абелева группа простого порядка Q, порожденная элементом G, Рассмотрим предикат

$$B_2 : x \longrightarrow x \pmod 2$$

и докажем следующую теорему.

Теорема. Предикат B_2 является сильным для функции $x \longrightarrow G^x$.

Доказательство. Символом $O(H,G)$ обозначим оракула, определяющего наименьший значащий бит дискретного логарифма от H по основанию G, т.е. вычисляющего $B_2(x)$, где $x = log_G H$. Покажем, как использовать O для решения проблемы дискретного логарифмирования.
Пусть дано значение $H = G^x$. Вычислим $T = 1/2 \pmod Q$ и положим $Y = 0$, $Z = 1$. Теперь, пока H не станет равным 1, будем осуществлять следующие шаги:

- $B = O(H,G)$;

- если B = 1, то положим $Y = Y + Z$ и $H = H/G$;

- переопределим $H = H^T$ и $Z = 2 \bullet Z$.

Как только H станет равным 1, мы получим, что $Y = \log_G H$.

Проследим за работой алгоритма на примере , взяв элемент $G = 64 \in F_{607}$ порядка $Q = 111$. Предположим, что нам нужно вычислить $\log_{64} 56$, Руководствуясь алгоритмом из доказательства теоремы, заполняем табл. 1:

H	$O(H,G)$	Z	Y
56	0	1	0
451	1	2	2
201	1	4	6
288	0	8	6
100	1	16	22
454	0	32	22
64	1	64	86

Можно убедиться в верности найденного решения, проверив равенство:

$$64^{86} = 56 (\mathrm{mod} 607).$$

2.4.Сильные предикаты для задачи RSA

Задача *RSA,* а именно, уравнение $C = m^E \pmod N$, обладает следующими сильными предикатами:

- $B_1(m) = m \pmod 2$;
- $B_h(m) = 0$, если $m < N/2$, $B_h(m) = 1$ в противном случае;
- $Bk(m) = m \pmod{2^k}$, где $k = O(\ln \ln N)$.

Обозначим соответствующие оракулы через $O_1(C,N)$, $O_h(C,N)$ и $O_k(C,N)$. Мы не будем исследовать последний из них, но отметим, что первые два взаимосвязаны:

$$O_h(C,N) = O_1(C \cdot 2^E \pmod N, N),$$
$$O_1(C,N) = O_h(C \cdot 2^{-E} \pmod N, N).$$

Покажем, как используя оракулы O_h и O_1 можно обратить функцию *RSA* при помощи алгоритма, основанного на стандартном двоичном поиске. Положим Y = C, $L = 0$ и $H = N$. Затем, до тех пор, пока $H — L \geq 1$, делаем следующее:

- $B = O_h(Y, N)$;
- $Y = Y*2^E \pmod N$;
- $M = (H + L)/2$;
- если $B = 1$, то положим $L = M$; в противном случае положим $H=M$.

При выходе из цикла значение [H] должно совпадать с прообразом C относительно функции *RSA*.

Разберем пример, в котором N =10403 и E = 7 — открытая информация, а мы хотим найти прообраз элемента C = 3 относительно функции *RSA,* опираясь на оракул $O_h(Y,N)$. Шаги алгоритма занесены в табл.2:

Y	$O(Y,N)$	L	H
3	0	0	10403
$3*2^7$	1	0	5201,5
$3*4^7$	1	2600,7	5201,5
$3*8^7$	1	3901,1	5201,5
$3*16^7$	0	4551,3	5201,5
$3*32^7$	0	4551,3	4876,4
$3*64^7$	1	4551,3	4713,8
$3*128^7$	0	4632,5	4713,8
$3*256^7$	1	4632,5	4673,2
$3*512^7$	1	4652,9	4673,2
$3*1024^7$	1	4663,0	4673,2
3 •*2048*	1	4668,1	4673,2
$3*4096^7$	1	4670,7	4673,2
$3*8192^7$	0	4671,9	4673,2

-	-	4671,9	4672,5

Таблица 2. Таблица промежуточных значении алгоритма обращения функции *RSA*

Из последней строки таблицы следует, что прообразом элемента 3 относительно функции

$$x \longrightarrow x^7 \pmod{10\,403}$$

служит число $m = 4672$.

2.5.Случайная саморедукция

Мы уже отмечали, что недостатки криптографической схемы Меркля - Хеллмана и других криптосистем, основанных на теории сложности, возникают ввиду того, что эти схемы ассоциированы с трудными в общем случае, но легкими при средних значениях параметров проблемами. Возникает естественный вопрос: откуда мы знаем, что задача *RSA* или проблема выбора Диффи-Хеллмана избавлены от этого недостатка? Не может ли так оказаться, что при известных модуле N и шифрующей экспоненте E уравнение

$$C = m^E \ (mod\ N)$$

трудноразрешимо при некоторых C, но легко решается для средних значений? Оказывается, можно доказывать, что задача *RSA* при фиксированном модуле N и проблема выбора Диффи -Хеллмана с данной группой A являются трудными в среднем случае.

Техника доказательства основывается на случайной саморедукции, сводящей конкретные начальные условия проблемы к случайному набору исходных данных. Это означает, что возможность решения задачи в среднем случае обеспечивает нам успех и в самом наихудшем. Значит, сложность решения задачи реально не зависит от того, наихудший или средний случай Вы рассматриваете.

Лемма. Проблема *RSA* позволяет применить к ней случайную саморедукцию.

Доказательство. Пусть нам нужно решить уравнение

$$C = m^E \ (\mathrm{mod}\ N)$$

относительно m, причем считается, что C здесь относится к «наиболее сложным» исходным данным. Сведем уравнение к «среднему» случаю, выбрав произвольным образом $S \in (Z/NZ)$ и положив

$$C' = S^E C.$$

Теперь мы пытаемся решить уравнение

$$C' = m'^E \pmod{N}$$

относительно неизвестной m'. Если нам не повезло, и мы опять пришли к трудному случаю, будем выбирать другие значения S до тех пор, пока не получим «среднего» условия. Предполагая простоту среднего случая, мы можем решить уравнение $C' = m' \pmod{N}$ относительно m' и вычислить

$$m = \frac{m'}{S},$$

что даст нам решение поставленной задачи.

3. ДОКАЗУЕМАЯ СТОЙКОСТЬ СО СЛУЧАЙНЫМ ОРАКУЛОМ

3.1.Введение

Современные способы проверки стойкости протоколов основываются на доказуемой стойкости. Это несколько дезориентирующее название соответствующей техники, поскольку она реально не доказывает криптостойкости в теоретико-информационном смысле, о котором мы говорили ранее. Сторонники доказуемой стойкости стремятся показать, что алгоритм успешной атаки на криптосистему можно использовать для создания другого алгоритма, существование которого считается невозможным.

Специалисты, например, пытаются показать, что атаку с выбором шифротекста, направленную на семантическую стойкость *RSA,* можно использовать для разложения на множители любого натурального числа. Иными словами, такое доказательство — относительный результат, при котором «доказательство» стойкости привязано к трудноразрешимости задачи факторизации.

Главный вклад доказуемой стойкости в криптографию состоит в точном определении понятий стойкого шифрования и стойкой схемы подписи. Многие из концепций, используемых в этой работе, экзистенциальная фальсификация, семантическая стойкость, неразличимость шифрования, адаптивная атака с выбором шифротекста и т. д., — появились благодаря изучению доказуемой стойкости.

Объясним технику доказуемой стойкости на конкретных примерах. Предположим, что нам дан атакующий алгоритм, взламывающий некоторую стойкость алгоритма *RSA* (например, семантическую) с определенной достаточно большой вероятностью. Однако сначала необходимо определить, что здесь означает понятие «достаточно большая вероятность».

Предположим, что у схемы есть параметр стойкости k, измеряющий, например, количество битов в ключе. В частности, параметр стойкости алгоритма *RSA* мог бы равняться числу двоичных знаков модуля N. Скажем, что противник достигает успеха с *достаточно большой* вероятностью, если вероятность достижения им своей цели больше, чем

$$\frac{1}{p(k)},$$

где $p(к)$ — некоторый многочлен, зависящий от k.

Допустим на минуту, что наш противник A пассивен, т.е. не обращается за помощью к расшифровывающему «черному ящику». Нам хотелось бы создать новый алгоритм B_A, имеющий на входе натуральное число N и вызывающий алгоритм A некоторое число раз, полиномиально зависящее от k. Цель нового алгоритма - найти простые множители числа N с достаточно большой вероятностью. Существование алгоритма B_A показало бы, что наличие пассивного алгоритма A, взламывающего *RSA,* влечет возможность разложения N на множители за полиномиальное время с достаточно большой вероятностью. Поскольку мы не верим в изобретение полиномиального алгоритма факторизации целых чисел на данном уровне развития науки, можно сделать вывод, что соответствующий алгоритм A также неосуществим на сегодняшний день.

Мы уже использовали этот прием, когда показывали, что успешная пассивная атака на криптосистему Эль-Гамаль дает эффективный алгоритм решения проблемы выбора Диффи-Хеллмана, или при демонстрации того, что успех в пассивной атаке на схему Гольдвассера-Микали дает эффективный алгоритм решения задачи о квадратичных вычетах.

Итак, по данному алгоритму A мы строим новый алгоритм B_A использующий A как стандартную процедуру. На вход алгоритму B_A подается трудная математическая задача, которую мы намерены решить, в то время как входными данными алгоритма A служит некоторая криптографическая задача. Трудности возникают, когда A моделирует активного противника, имеющего доступ к дешифрующему или подписывающему оракулу, соответствующему данному открытому ключу. Алгоритм B_A, использующий A как подпрограмму, должен обеспечить A ответами на вопросы, которые тот задает оракулу. В этой ситуации перед алгоритмом B_A возникает несколько проблем:

- Ответы, поставляемые алгоритму A, должны быть корректными, т.е. шифротексты необходимо расшифровывать, а подписи необходимо проверять. В противном случае алгоритм A может заметить, что оракул дает лживые ответы. Следовательно, теперь у алгоритма B_A нет никаких гарантий в том, что A достигает успеха с достаточной вероятностью. Ответы псевдо-оракула должны быть распределены с той плотностью вероятностей, которую A ожидает от истинного оракула, поскольку в противном случае A заметит подлог.
- Отвечать необходимо на любой вопрос, который алгоритм A задает оракулу.
- Алгоритм B_A должен генерировать ответы, не прибегая к секретному ключу. Например, в случае *RSA,* если B_A нацелен на поиск множителей числа N, то он едва ли сможет воспользоваться этими множителями для ответов алгоритму A прежде, чем определит их.

Последняя из перечисленных проблем наиболее существенна. Фактически, мы требуем от алгоритма B_A расшифровать или подписать сообщение, не опираясь на

знание секретного ключа, но именно это и считается невозможным ввиду криптостойкости схемы.

Для преодоления возникающих трудностей принято использовать так называемую «модель случайного оракула». Случайный оракул — это идеализированная хэш-функция, которая на каждый новый запрос выдает случайный ответ, равномерно распределенный по области значений, с условием; если один и тот же запрос поступит дважды, то ответ должен быть одинаковым.

В модели случайного оракула предполагается, что противник *A* при атаке на схему не пользуется явной хэш-функцией, определенной в этой схеме. Другими словами, противник *A* достигнет успеха даже в том случае, если мы заменим настоящую хэш-функцию схемы случайным оракулом. Алгоритм B_A отвечает на запросы противника *A*, обманывая последнего «сфабрикованными» случайным оракулом ответами, чтобы удовлетворить свои собственные потребности. Чтобы увидеть, как это осуществляется на практике, стоит перейти к следующему разделу о доказательстве стойкости алгоритмов подписи.

Доказательство с моделью случайного оракула носит еще более относительный характер, чем предыдущие. Такие доказательства говорят, что в предположении о трудноразрешимости определенной задачи, например, факторизации, успешного нападения, не использующего подлежащей хэш-функции, быть не может. При этом, естественно, не утверждается, что нет успешных атак, которые имеют доступ к фактической хэш-функции криптосистемы.

3.2.Стойкость алгоритмов подписи

Сначала мы рассмотрим стойкость алгоритмов цифровой подписи, поскольку доказательства здесь более просты, чем в алгоритмах шифрования. Первый основной инструмент, о котором мы будем говорить, — это лемма, принадлежащая Стерну и Пойнтчевалю. Она применяется к определенному типу схем подписи, которые используют хэш-функции следующим образом: чтобы подписать сообщение

- производят (возможно пустое) обязательство Σ_1;
- вычисляют хэш-значение $H = \mathrm{H}(\Sigma_1 \| \mathrm{M})$;
- находят Σ_2 — «подпись» на Σ_1 и H.

Чтобы запомнить запрос к хэш-функции, запишем выходные данные схемы в виде $(\Sigma_1, \mathrm{H}(\Sigma_1 \| \mathrm{M}), \Sigma_2)$. Например,

$$DSA\colon \Sigma_1 = \varnothing,\ H = H(M),$$
$$\Sigma_2 = \left(R, \frac{H + xR}{k} \pmod{Q}\right),$$

где $R = \left(Q^k \pmod P\right) \pmod Q$.

Схема подписи Шнорра: $\Sigma_1 = G^k$, $H = H(\Sigma_1 || M)$

$$\Sigma_2 = xH + k \pmod Q.$$

Во всех перечисленных схемах предполагают, что хэш-функция принимает значения в F_Q.

Напомним, что в модели случайного оракула хэш-функция позволяет алгоритму B_A генерировать произвольные ответы на запросы противника A. Предположим, что в модели случайного оракула нападающий алгоритм A может генерировать экзистенциальную фальсификацию для сообщения M с достаточно большой вероятностью. Тогда выходные данные алгоритма A имеют вид:

$$(M, \Sigma_1, H, \Sigma_2).$$

Можно считать, что противник создает критический хэш-запрос

$$H = H(\Sigma_1 || M),$$

поскольку в противном случае мы можем самостоятельно генерировать запрос нападающего.

Алгоритм B_A теперь дважды вызывает подпрограмму противника A с неизменными случайными входными данными, но слегка измененным случайным оракулом. Противник A выполняет свою работу за полиномиальное время и при этом выдает полиномиальное количество хэш-эапросов. Если на все хэш-запросы противник получает тот же самый ответ, то A должен будет выдать неизменную подпись. Однако алгоритм отвечает на все случайные запросы нападающего, как и раньше, кроме одного, выбранного случайным образом, и на него отвечает по-другому. С достаточно большой вероятностью этот «исключительный» запрос окажется критическим хэш-запросом и поэтому (с достаточно большой вероятностью) противник B_A получит две подписи на одном и том же сообщении, которые имеют разные ответы на хэш-запросы. Иными словами, мы получим

$$(M, \Sigma_1, H, \Sigma_2) \quad \text{и} \quad (M, \Sigma_1, H', \Sigma_2').$$

Теперь, опираясь на эти два набора выходных данных алгоритма A, можно попытаться решить трудную задачу, составляющую цель алгоритма B_A.

Подробности и полный разбор этого приема можно найти в работе Стерна и Пойнтчеваля.

3.2.1.Примеры пассивного противника

Схема подписи Шнорра. Лемма Стерна и Пойнтчеваля позволяет доказать следующую теорему.

Теорема. Из предположения о трудноразрешимости задачи дискретного логарифмирования в данной группе вытекает (в модели случайного оракула), что успешная пассивная атака, направленная на схему подписи Шнорра с данной группой, невозможна.

Доказательство. Пусть входными данными в алгоритм B_A служит задача дискретного логарифмирования $Y = G^x$, которую мы намерены решить. Предположим, что мы вызываем процедуру пассивного противника A, обладающего открытым ключом Y в качестве входных данных, и попытаемся использовать аргументы леммы Стерна и Пойнтчеваля, С достаточно большой вероятностью мы получим две подписи

$$\left(M, \Sigma_1 = G^k, H, \Sigma_2 = xH + k \pmod{Q}\right)$$

и

$$\left(M, \Sigma_1' = G^{k'}, H', \Sigma_2' = xH' + k' \pmod{Q}\right),$$

где $H = \mathrm{H}(\Sigma_1 || M)$ — ответ случайного оракула при первом проходе алгоритма A, а $H' = \mathrm{H}(\Sigma_2' || M)$ — при втором.

Цель алгоритма B_A — найти неизвестное число x. Он делает вывод: $k = k'$, поскольку $\Sigma_1 = \Sigma_1'$,. Следовательно,

$$Cx = D \pmod{Q},$$

где

$$C = H - H' \pmod{Q}, \qquad D = \Sigma_2 - \Sigma_2' \pmod{Q}.$$

Кроме того, известно, что $A \neq 0$, поскольку в противном случае два хэш-значения были бы равны между собой. Следовательно, алгоритм B_A может решить исходное логарифмическое уравнение, вычисляя

$$x = C^{-1}D \pmod{Q}.$$

Позже мы покажем, что схема подписи Шнорра в модели случайного оракула защищена и от активного нападения.

DSA-подписи. Как мы сейчас покажем, приведенные в предыдущем разделе рассуждения не применимы для схемы подписи *DSA*.

Пусть входными данными B_A является уравнение $Y = G^x$ относительно неизвестной x, которое мы намерены решить. Опять будем вызывать процедуру противника A, на вход которого подается открытый ключ Y, и будем применять лемму Стерна и Пойнтчеваля. С достаточно большой вероятностью мы получим подписи

$$(M, \Sigma_1 = \varnothing, H, \Sigma_2 = (R, S)) \quad \text{и} \quad (M, \Sigma_1' = \varnothing, H', \Sigma_2' = (R', S'))$$

где H = *H(M)* — ответ случайного оракула из первого вызова *A*, а H' = *H(M)* — из второго. Кроме того,

$$R = G^k \pmod P \pmod Q, \qquad R' = G^{k'} \pmod P \pmod Q,$$
$$S = \frac{H + xR}{k} \pmod Q, \qquad S' = \frac{H' + xR'}{k'} \pmod Q.$$

Цель алгоритма B_A — найти *x*. Однако здесь мы не можем считать, что $k = k'$, поскольку мы не знаем, верно ли равенство: R = R'. Значит, прием, сработавший при проверке стойкости схемы подписи Шнорра, в этой ситуации не применим. Фактически, доказательство стойкости схемы подписи *DSA* пока отсутствует.

Можно попытаться подправить доказательство, несколько видоизменив схему *DSA,* применяя хэш-функцию одновременно к *M* и *R*, а не только к *M,* Такое изменение схемы в терминах леммы Стерна и Пойнтчеваля влечет, что $\Sigma_1 = R$, $H = H(M||R)$ и

$\Sigma_2 = (H+xR)/k \pmod Q$.

Даже при такой модификации, приближающей схему *DSA* к схеме Шнорра, мы не сможем доказать ее стойкости. По лемме мы получим две подписи, в которых

$$R = G^k \pmod P \pmod Q, \qquad R' = G^{k'} \pmod P \pmod Q,$$
$$S = \frac{H + xR}{k} \pmod Q, \qquad S' = \frac{H' + xR'}{k'} \pmod Q,$$

причем $R = R'$. Но мы все еще не сможем сделать вывод о равенстве $k = k'$ поскольку оно не вытекает из уравнения

$$G^k = G^{k'} \pmod P \pmod Q.$$

Препятствие — в приведении по модулю *Q*. Удалив последнюю операцию, мы получим возможность доказать стойкость подписи. Однако при этом теряется привлекательное свойство малого размера подписи *DSA*.

3.2.2.Активный противник

Для доказательства стойкости схем в отношении активного противника нам необходимо показать, как алгоритм B_A будет отвечать на требования подписи со стороны алгоритма *A*. С этой целью мы опять подключаем модель случайного оракула, в которой будем использовать способность алгоритма B_A выбирать значения хэш-функции. Заметим, что аргумент хэш-функции может быть неизвестен вплоть до получения подписи на сообщении. Если хэш-функция при этом применяется только к сообщению *M* и не зависит от других величин (как Σ_1

в предыдущих рассуждениях), то алгоритм A может послать запрос хэш-оракулу, входными данными которого служит сообщение M, перед тем, как подпись будет создана. В этой ситуации алгоритм B_A не способен изменить ответ по сравнению с предыдущим.

Процесс ответов алгоритма B_A на запросы подписи нападающего A без его участия и без информации о секретном ключе называется моделированием запросов подписи. Такое моделирование по существу означает, что активный противник обладает не большими возможностями, чем пассивное нападение в модели случайного оракула, поскольку любой активный противник может быть превращен в пассивного простым моделированием запросов подписи.

Схема подписи Шнорра. Моделирование запросов подписи в доказательстве стойкости схемы Шнорра осуществляется довольно легко, Оно тесно связано с протоколами доказательств с нулевым разглашением. Мы предполагаем, что симулятор хранит список L всех предыдущих запросов случайного оракула. По введенному в него сообщению M симулятор делает следующее:

2. Вычисляет случайные числа S и H, удовлетворяющие условию: $1 < S, H < Q$.
3. Полагает $R = Q^S Y^H$.
4. Если $(R||M,H') \in L$ при $H' \neq H$, то симулятор возвращается к шагу 1.

4. Присоединяет к множеству L элемент $(R||M, H)$, т.е. на запрос $(R||M)$ хэш-оракул теперь будет всегда выдавать ответ H.

5. Выдает «подпись» *(H,S)* .

Итак, мы получили теорему.

Теорема. Если задача дискретного логарифмирования в группе A трудноразрешима, то (в модели случайного оракула) не существует успешной активной атаки на схему подписи Шнорра с группой A.

В отношении *DSA* аналогичных результатов пока не получено. Доказательство стойкости известно для схемы *EC-DSA,* где вместо моделирования хэш-функции и сведения стойкости к проблеме дискретного логарифмирования в качестве главного объекта моделируется групповая операция, а стойкость сводится к проблеме повторяющихся значений реально используемой хэш-функции.

3.2.3.RSA-FDH

Следует отметить, что в нашем обсуждении схем Шнорра, *DSA* и *EC-DSA* предполагалось, что значения хэш-функции H принадлежат полю F_Q. Такие хэш-функции трудно построить на практике, хотя предыдущие рассуждения использовали именно это свойство. Аналогичная ситуация возникает в одном из

вариантов схемы подписи RSA, называемом *RSA-FDH* (от англ, *full domain hash).* Хэш-функция в ней — отображение

$$H : \{0,1\} \longrightarrow (\mathbb{Z}/N\mathbb{Z})^*,$$

где N — RSA-модуль. Такую хэш-функцию тоже трудно реализовать, но предположив, что она существует, и включив ее в модель случайного оракула, мы сможем доказать стойкость следующего алгоритма подписи *RSA.*

Пусть N — модуль алгоритма *RSA,* E — шифрующая, а d— расшифровывающая экспоненты. Обозначим через f функцию, действующую по правилу:

$$f : \begin{cases} (\mathbb{Z}/N\mathbb{Z})^* \longrightarrow (\mathbb{Z}/N\mathbb{Z})^*, \\ x \mapsto x^E. \end{cases}$$

В этих терминах задача *RSA* заключается в определении x по данному $Y = f(x).$ В схеме *RSA-FDH* подпись на сообщении имеет вид

$$S = H(M)^d = f^{-1}(H(M)).$$

Можно доказать следующую теорему.

Теорема. Если в модели случайного оракула существует активный противник A, способный добиться экзистенциальной фальсификации в схеме *RSA-FDH,* используя q_H хэш-запросов и q_S обращений к подписывающему оракулу, то найдется алгоритм, который по данному Y может восстановить соответствующий прообраз функции *RSA* с вероятностью $1/q_H$.

Доказательство. Опишем алгоритм $B_A,$ который по данному $Y \in$ (Z/NZ) вычисляет $x = f^1(Y)$. Алгоритм B_A сначала выбирает значение T $\in [1, \ldots, q_H]$ и заводит нумерованный список всех сделанных хэш-запросов. Затем он вызывает подпрограмму A и отвечает на сделанный противником хэш-запрос для введенного сообщения M_i следующим образом:

- если A делает хэш-запрос, совпадающий с уже сделанным ранее запросом под номером T, то B_A дает Y в качестве ответа и модифицирует список запросов так, чтобы выполнялось равенство $Y = H(M_T)$

- если хэш-эапрос противника A выглядит как M_I с $I \neq T$ то B_A вычисляет случайный элемент $S_i \in$ (Z/NZ) и добавляет к списку хэш-запросов $H(M_I) = S_I^E \pmod N = H_I,$ храня запись значения $S_I,$ а в качестве ответа выдает H_I.

Если противник A требует подписать сообщение M_I перед хэш-запросом с этим сообщением, то перед ответом B_A моделирует соответствующий хэш-запрос вместо алгоритма A. После этого ответ на запрос о подписи выглядит так:

- если сообщение M_I совпадает с M_T то алгоритм останавливается и сообщает о сбое;
- если $M_I \neq M_T$, то в качестве ответа на запрос подписи B_A выдает S_I.

Допустим, что противник закончит работу и выдаст в качестве результата подписанное сообщение *(S, M)*. Без ограничения общности можно предполагать, что при этом алгоритм A делал хэш-запрос для M. Если $M \neq M_T$, то алгоритм B_A останавливается с сообщением о неудачной попытке обратить функцию. Если же $M = M_T$, то имеет место соотношение

$$f(S) = H(M_T) = Y.$$

Следовательно, мы вычислили требуемый прообраз функции f.

Анализируя алгоритм B_A, заметим, что при корректной работе противник A создал экзистенциальную подделку (M, S), и поэтому он не мог в процессе работы требовать от оракула подписать сообщение M. Выбранное в начале работы значение T не зависит от вида алгоритма A. Поэтому алгоритм A не может постоянно требовать поставить подпись на сообщение M_T. Значит, грубо говоря, вероятность успешного завершения алгоритма B_A приблизительно равна $1/q_H$ т.е. вероятности того, что экзистенциальная подделка была сделана именно на сообщении M_T, а не на каком-то другом.

3.2.4.RSA-PSS

Другой способ применения *RSA* в качестве подписывающего алгоритма основан на использовании так называемой *RSA-PSS* или вероятностной схемы подписи *RSA* (от англ. *probabilistic signature scheme)*. Ее стойкость тоже доказывается с помощью модели случайного оракула. Мы не станем давать здесь подробное доказательство, ограничившись простым описанием схемы, поскольку ее значение постоянно возрастает. Преимущество этой схемы перед *RSA-FDH* состоит в том, что в ней используется традиционная хэш-функция, принимающая значения в битовых строках длины L, а не в кольце вычетов по модулю составного числа.

Как обычно, фиксируем модуль N алгоритма *RSA,* открытую экспоненту E и секретную экспоненту d. Обозначим параметр безопасности через k, т.е. модуль N состоит из k двоичных знаков. Определим два целых числа k_0 и k_1 так, чтобы

$$k_0 + k_1 \leqslant k - 1.$$

Например, можно взять *к.* = 128 или 160.

Кроме того, зададим две хэш-функции, одна из которых разворачивает данные, а другая их сжимает:

$$G : \{0,1\}^{k_1} \longrightarrow \{0,1\}^{k-k_1-1},$$
$$H : \{0,1\}^{*} \longrightarrow \{0,1\}^{k_1}.$$

Пусть

$$G_1 : \{0,1\}^{k_1} \longrightarrow \{0,1\}^{k_0}$$

обозначает функцию, сопоставляющую строке $w \in \{0,1\}^{k1}$ первые k_0 знаков числа $G(w)$, а функция

$$G_2 : \{0,1\}^{k_1} \longrightarrow \{0,1\}^{k-k_0-k_1-1}$$

сопоставляет строке $w \in \{0,1\}^{k1}$ оставшиеся $k - k_0 - k_1 - 1$ знаков числа $G(w)$.

Чтобы поставить подпись на сообщении M, делают следующее:

- генерируют случайную строку $R \in \{0,1\}^{k0}$,
- вычисляют $W = H(M||R)$,
- определяют $Y = 0||W||(G_1(W) \oplus R)||G_2(W)$,
- выдают подпись в виде $S = Y^d \pmod N$.

Для проверки подписи (S, M)

- вычисляют $Y = S^E \pmod N$;
- разбивают Y на компоненты:

$B||W||\alpha||\gamma$.

где B — однобитовое слово, W — k_1-битовое, α — k_0-битовое, а γ состоит из $k - k_0 - k_1 - 1$ знаков;

- вычисляют $R = \alpha \oplus G_1(W)$;
- подпись имеет силу, если

$B = 0$, $G_2(W) = \gamma$ и $H(M||R) = W$.

Если доверить моделирование хэш-функций G и H случайному оракулу, то можно показать, что изложенная схема подписи стойка в том смысле, что существование успешного алгоритма, осуществляющего экзистенциальную фальсификацию, влечет обращение функции *RSA*.

3.3.Стойкость шифрующих алгоритмов

Мы видели, что в предположениях проблемы выбора Диффи - Хеллмана довольно легко создать семантически стойкую схему шифрования с открытым ключом, имея в виду лишь пассивного противника. Например, криптосистема Эль-Гамаль обладает этим свойством. Кроме того, мы убедились, что можно легко построить семантически стойкие шифры, основанные на проблеме квадратичных вычетов, опять-таки принимая во внимание только пассивных противников. К сожалению, система Гольдвассера и Микали, о которой идет речь, обладает весьма неприятным свойством, сильно увеличивающим объем сообщений. Оказывается, создание криптосистемы, основанной на проблеме дискретного логарифмирования, защищенной от активного противника или семантически стойкой в предположениях *RSA,* сталкивается со значительными трудностями. В этом параграфе мы сначала остановимся на ранних попытках конструирования криптосистемы, основанной на примитиве Эль-Гамаль, которая была бы защищена от активного противника. На их примере удобно продемонстрировать основные принципы проектирования. Затем мы перейдем к описанию главной системы, построенной на RSA-примитиве, которая в модели случайного оракула защищена от активного противника, а именно, системы *RSA-OAEP.*

3.3.1.Иммунизация криптосистем, основанных на Эль-Гамаль

Напомним, что шифрование Эль-Гамаль имеет вид

$$(G^k, mY^k),$$

где $Y = G^x$ — открытый ключ. Используя довольно элементарную технику, мы уже показали в главе 2, что в предположениях проблемы выбора Диффи-Хеллмана шифрование Эль-Гамаль обладает семантической стойкостью. Однако там же было продемонстрировано, что такая система не стойка в отношении активного противника, поскольку соответствующий шифротекст не обладает жесткостью.

Было понято, что проблема, связанная с активным противником, состоит в том, что нападающий может слишком легко создать корректный шифротекст. Дело в том, что если бы противнику было трудно моделировать шифротекст, поддающийся расшифрованию, то атака с выбором шифротекста не дала бы ему никаких особых преимуществ. Действительно, у него нет причин требовать расшифрования шифротекста, который он может получить только зашифровав самостоятельно какой-то открытый текст. Это означает, что необходима такая расшифровывающая процедура, которая по введенному в нее шифротексту выдает или подлежащий открытый текст, или сообщение о несоответствии этого

шифротекста какому-либо открытому сообщению. Для этой цели нужно добавлять к шифротексту некоторую дополнительную информацию, по которой расшифровывающая процедура могла бы определять, получен ли этот шифротекст с помощью законной процедуры шифрования из какого-то связного сообщения.

Женг и Себерри первыми применили эту философию для создания практической криптосистемы, которая определяет современный подход к разработке шифрующих функций с открытым ключом. Их работа имеет большое значение. Поэтому мы расскажем об этих ранних попытках разработать стойкие криптосистемы с открытым ключом в качестве иллюстрации такого подхода.

Первое, на что стоит обратить внимание: шифрование с открытым ключом обычно используется для того, чтобы скрыть ключ, с помощью которого будет шифроваться большое сообщение. Следовательно, нет необходимости в шифровании сообщения, которое принадлежит группе *A* (как в Эль-Гамаль). Однако можно воспользоваться идеей Эль-Гамаль для передачи ключа, с помощью которого будет выработан сеансовый ключ, шифрующий фактическое сообщение. Введем некоторые обозначения.

A — нескрываемая группа простого порядка Q, порожденная элементом G.

V(H) — функция, которая по элементу *H* группы генерирует случайную строку битов. Ее часто называют функцией, производящей ключи.

H — хэш-функция со значением в /-битовых строках.

$Y = G^x$ — открытый ключ, соответствующий секретному ключу *X*.

Схема Женга-Ссберри 1. Чтобы зашифровать сообщение *m*, вычисляют

1. $k \in \{1,...,Q-1\}$.
2. $z = V(Y^k)$.
3. $t = H(m)$.
4. $C_1 = G^k$.
5. $C_2 = z \oplus (m||t)$.
6. Шифротекст — (C_1,C_2).

При расшифровании предпринимают такие шаги.

1. $z' = V(C_1^x)$.
2. $w = z' \oplus C_2$.
3. t' — последние l битов строки w.

4. m' — первые $\#w - l$ знаков строки w.
4. Если $H(m') = t'$ то m' — расшифрованное сообщение.

6. В противном случае алгоритм выдает сообщение о некорректности шифротекста.

Особенность криптосистемы состоит в добавлении к шифротексту Эль-Гамаль специальной информации, а именно, зашифрованного хэш-значения от открытого текста. Поскольку (предположительно) хэш-функцию обратить трудно, практически невозможно написать корректный шифротекст не зная соответствующий открытый. Дописывание дополнительного хэш-значения вносит требуемую шифротексту избыточность, которая тестируется расшифровывающей функцией.

Схема Женга-Себерри 2. Вторая система использует универсальную одностороннюю хэш-функцию, которая является, по существу, параметрическим набором хэш-функций H_i с $i < l$. Ее можно представлять себе как функцию, снабженную переключателями, или как *MAC*.

Говоря более формально, универсальная односторонняя хэш-функция это функция H_k, снабженная переключателями, обладающая следующим свойством: если противнику дан X и выбран наугад секретный ключ k, то противнику должно быть крайне сложно подобрать такой элемент y, что

$$H_к(y) = H_к(X).$$

Чтобы зашифровать сообщение m во второй схеме Женга-Себерри, мы вычисляем

1. $k \in \{1,\ldots,Q-1\}$.
2. z — $\#m$ левых знаков числа $V(Y^k)$.
3. s — t правых знаков числа $V(Y^k)$.
4. $C_1 = G^k$
5. $C_2 = H_s(m)$.
6. $C_3 = z \oplus m$.
7. Шифротекст-(C_1,C_2,C_3).

Эта система аналогична первой, но здесь хэш-значение сообщения не шифруется, а пересылается в открытом виде. Эта система сложнее предыдущей, т. к. мы не знаем, какой конкретно хэш-функцией или ключом получено соответствующее хэш-значение. Вторая схема Женга-Себерри очень близка системе *DHIES,* которая на данный момент считается наилучшей практической реализацией функции шифрования Эль-Гамаль.

Схема Женга-Себерри 3. В третьей и последней схеме Женга и Себерри эксплуатируется DSA-подобная подпись, или «символ» шифруемого сообщения. Схема работает, комбинируя шифрование Эль-Гамаль с DSA-подписью, однако открытый ключ схемы подписи *DSA* носит эфемерный характер и составляет часть шифротекста.

1. $k,t \in \{1,..,Q-1\}$.
2. $r = Y^{k+t}$.
3. $z = G^r$.
4. $C_1 = G^k$.
5. $C_2 = G^t$
5. $C_3 = (H(m) + xr)/k$ (modQ),
6. $C_4 = z + m$.
7. Шифротекст (C_1, C_2, C_3, C_4).

Женг и Себерри доказали стойкость своих схем при очень сильных предположениях, а именно, они считали, что пространство шифротекстов взаимно однозначно соответствует пространству сообщений, что является близким предположению о текстозависимости алгоритма шифрования. Однако можно показать, что первая из схем не стойка. Предположим, что на стадии *поиска* противник выдает два сообщения M_1 и M_2. Затем выбирают скрываемый бит b и предлагают противнику сообщение M_b в зашифрованном виде, т.е. шифротекст

$$C = (C_1, C_2) = \left(G^k, z \oplus (M_b || H(M_b))\right)$$

Противник на стадии *гипотезы* может предпринять следующие операции. Сначала он генерирует новое сообщение M_3 отличное от M_1 и M_2, но совпадающее с ними по длине. Затем нападающий требует от оракула расшифровать шифротекст

$$\left(C_1, C_2 \oplus (M_1 || H(M_1)) \oplus (M_3 || H(M_3))\right)$$

Если $b = 1$, то этот шифротекст — корректная шифрованная версия сообщения M_3. Поэтому оракул, расшифровав его, должен получить M_3. А если $b = 0$, то шифротекст с очень малой долей вероятности будет шифрованной версией хоть какого-то сообщения, не говоря уже о M_3. Таким образом мы получаем алгоритм, который за полиномиальное время определяет значение скрываемого бита b.

3.3.2.RSA-OAEP

Напомним, что необработанная RSA-функция не дает семантически стойкой шифрующей системы даже в отношении пассивного противника. Для создания стойкой системы нам нужно или добавлять избыточную информацию к открытому тексту перед шифрованием, или вставлять лишние данные в сам шифротекст. Кроме того, в целях получения недетерминированного процесса, добавляемые куски должны выбираться случайным образом, В *RSA* это достигается использованием специальной пополняющей схемы, и за последние годы было предложено много таких схем. Однако некоторые из самых ранних предложений на сегодняшний день считаются слишком слабыми.

Одна из наиболее удачных на данный момент ПОПОЛНЯЮЩИХ схем была изобретена Белларе и Рогавей. Она называется *оптимизированным асимметричным пополнением шифрования* или *OAEP* (от англ. Optimized Asymmetric Encryption Padding), *OAEP*— пополняющая схема, которую можно использовать с любой односторонней функцией с секретом, в частности, с RSA-функцией. Когда ее используют совместно с *RSA,* то обозначают *RSA-OAEP.*

В первое время после создания *RSA-OAEP* считалось, что эта система является текстозависимой, но вскоре стало ясно, что это не так. Однако с помощью модели случайного оракула можно показать, что *RSA-OAEP* семантически стойка в отношении адаптивной атаки с выбором шифротекста.

Расскажем сначала в общем виде о процедуре шифрования *OAEP.*Пусть f - односторонняя функция с секретом, которая отображает k-битовые строки в k-битовые строки. Если $к$ = 1024, то такой функцией можно считать RSA-функцию $C = m^E$. Пусть k_0 и k_1 такие числа, что 2^{k0} и 2^{k1} нереализуемо (например, $k_0,k_1 >$ 128). Положим n =$k — k_0 — k_1$ и обозначим через

$$G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{n+k_1}, \quad H : \{0,1\}^{n+k_1} \longrightarrow \{0,1\}^{k_0}$$

хэш-функции. Пусть m — сообщение, состоящее из n битов. Зашифруем его, используя функцию

$$\varepsilon(m) = f\Big(\{m||0^{k_1} \oplus G(r)\}||\{r \oplus H(m0^{k_1} \oplus G(r))\}\Big)$$

где

- $m||0^{k1}$ означает m, за которым стоит k_1 нулей,
- r — случайная строка битов длины k_0,
- знак || означает конкатенацию строк.

Можно представлять себе *OAEP* как двухэтапную сеть Фейстеля (см. рис,1).

Чтобы расшифровать сообщение ε(m), мы вычисляем

$$A = \{T||\{r \oplus H(T)\}\} = \Big\{\{m||0^{k_1} \oplus G(r)\}||\{r \oplus H(m||0^{k_1} \oplus G(r))\}\Big\}$$

Таким образом, мы знаем

$$T = m||0^{k_1} \oplus G(r).$$

Следовательно, можно вычислить *H(T)* и найти *r*, зная $r \oplus H(T)$. По *r* получаем *G(r)* и восстанавливаем сообщение *m*. Заметим, что нам необходимо проверить, действительно ли строка $T \oplus G(r)$ оканчивается k_1 нулями. Если это не так, то мы должны сделать вывод, что соответствующий шифротекст некорректен.

Сформулируем основной результат, посвященный системе *RSA-OAEP*.

Если предположения *RSA* справедливы, то в модели случайного оракула система *RSA-OAEP* семантически стойка в отношении адаптивной атаки с выбором шифротекста при условии моделирования хэш-функций *G* и *H* случайным оракулом.

3.3.3.Преобразование схем *CCA* в схемы *CCA2*

Предположим, что у нас есть схема шифрования с открытым ключом, семантически стойкая в отношении атаки с выбором открытого текста, например, Эль-Гамаль. По определению, такая схема должна быть недетерминированной, поэтому мы записываем шифрующую функцию как

$$E(m, r)$$

где *m* предназначенное для шифрования сообщение, а *r* - случайная дополнительная информация. Расшифровывающую функцию будем, как обычно, обозначать через *D(C)*. Следовательно, в шифровании Эль-Гамаль имеем

$$E(m, r) = (G^r, m \cdot H^r).$$

Фуджисаки и Окамото показали, как превратить такую схему в криптосистему, семантически стойкую в отношении адаптивного нападения в модели случайного оракула. Фактически, они доказывают текстозависимость трансформированной схемы. Мы опустим доказательство вообще, но приведем конструкцию трансформации, которая настолько, же проста, насколько элегантна.

Меняем шифрующую функцию следующим образом:

$$E'(m, r) = E(m||r, F(m||r)),$$

где *F* — какая-то хэш-функция. Расшифровывающий алгоритм тоже несколько видоизменяется. Сначала вычисляют

$$m' = D(C),$$

а потом проверяют равенство

$$C = E(m', F(m')).$$

Если равенство истинно, то m восстанавливается из $m' = m||r$. В противном случае делается вывод о некорректности шифротекста.

В частности, для Эль-Гамаль получается шифрующий алгоритм

$$(G^{F(m||r)}, (m||r) \cdot H^{F(m||r)}),$$

который лишь немного менее эффективен, чем исходная криптосистема.

2.4.Атаки на асимметричные системы

2.4.1.Атака Винера на RSA

Мы уже отмечали, что в алгоритме *RSA* для ускорения операций с открытым ключом используют малые шифрующие экспоненты. В некоторых же приложениях этой криптосистемы существеннее ускорить процессы расшифровывания. Поэтому имеет смысл выбирать небольшую расшифровывающую экспоненту *d.* Ясно, что при этом получается большое значение открытой экспоненты *E,* Слишком маленькое число в качестве секретной экспоненты *d* мы брать не можем, поскольку атакующий определит ее простым перебором. Более того, учитывая изощренную атаку Винера, опирающуюся на непрерывные дроби, необходимо выбирать *d* среди чисел, размер которых не меньше, чем \N*.

По вещественному числу $\alpha \in$ R определим последовательности:

$$\alpha_0 = \alpha, \quad p_0 = q_0 = 1, \quad p_1 = a_0 a_1 + 1, \quad q_1 = a_1,$$

$$a_i = \lfloor \alpha_i \rfloor, \quad \alpha_{i+1} = \frac{1}{\alpha_i - a_i},$$

$$p_i = a_i p_{i-1} + p_{i-2} \text{ при } i \geqslant 2,$$

$$q_i = a_i q_{i-1} + q_{i-2} \text{ при } i \geqslant 2.$$

Целые числа α_0, α_1, α_2, ... называются непрерывной дробью, представляющей α, а рациональные числа p_i/q_i — подходящими дробями. Каждая из подходящих дробей несократима, а скорость роста их знаменателей сравнима с показательной.

Одним из важных результатов теории непрерывных дробей является то, что если несократимая дробь p/q удовлетворяет неравенству:

$$\left|\alpha - \frac{p}{q}\right| \leqslant \frac{1}{2q^2},$$

то *p/q* — одна из подходящих дробей в разложении α в непрерывную дробь.
Винер предлагает использовать непрерывные дроби при атаке на *RSA* следующим образом. Пусть у нас есть модуль $N = pq$, причем $q < p < 2q$. Допустим, что наша расшифровывающая экспонента удовлетворяет неравенству: $d < \frac{1}{3}N^{\frac{1}{4}}$, и нападающему ЭТО известно. Кроме того, ему дана шифрующая экспонента *E*, обладающая свойством

$$Ed = 1 \pmod{\varphi},$$

где $\varphi = \varphi(N) = (p-1)(q-1)$. Будем также считать, что $E < \varphi$ поскольку это выполнено в большинстве приложений. Заметим, из предположений следует существование такого целого *к,* что

$$Ed - k\varphi = 1.$$

Следовательно,

$$\left|\frac{E}{\varphi} - \frac{k}{d}\right| = \frac{1}{d\varphi}.$$

Поскольку $\varphi \approx N$, получаем, что

$$|N - \varphi| = |p + q - 1| < 3\sqrt{N}.$$

Отсюда можно сделать вывод, что *E/N* довольно хорошее приближение в *k/d*. Действительно,

$$\left|\frac{E}{N} - \frac{k}{d}\right| = \left|\frac{Ed - Nk}{dN}\right| = \left|\frac{Ed - k\varphi - Nk + k\varphi}{dN}\right| =$$
$$= \left|\frac{1 - k(N - \varphi)}{dN}\right| \leqslant \left|\frac{3k\sqrt{N}}{dN}\right| = \frac{3k}{d\sqrt{N}}.$$

Поскольку $E < \varphi$, очевидно, $к < d$. Кроме того, по предположению $d < \frac{1}{4}N^{\frac{1}{4}}$. Значит,

$$\left|\frac{E}{N} - \frac{k}{d}\right| < \frac{1}{2d^2}.$$

Поскольку *НОД(k,d)* = 1, мы видим, что *k/d* — подходящая дробь в разложении дроби *E/N* в непрерывную. Таким образом, раскладывая число *E/N* в непрерывную дробь, можно узнать расшифровывающую экспоненту, поочередно подставляя знаменатели подходящих дробей в выражение:

$$\left(M^E\right)^d = M \pmod{N}$$

для некоторого случайного числа *M.* Получив равенство, найдем *d.* Общее число подходящих дробей, которое нам придется при этом проверить, оценивается как O(ln*N*). Таким образом, изложенный метод дает линейный по сложности алгоритм определения секретного ключа в системе *RSA,* если последний не превосходит $\frac{1}{3}N^{\frac{1}{4}}$. В качестве примера рассмотрим модуль *RSA,* равный
N = 9449868410449.
Пусть открытый ключ криптосистемы задан как
E = 6792605526025,

а секретный ключ удовлетворяет неравенству $d < \frac{1}{3}N^{\frac{1}{4}} \approx 584$. Разложим число α = *E/N* в непрерывную дробь и проверим знаменатель каждой подходящей дроби: не является ли он секретным ключом. Подходящие дроби разложения α имеют вид:

$$1, \frac{2}{3}, \frac{3}{4}, \frac{5}{7}, \frac{18}{25}, \frac{23}{32}, \frac{409}{569}, \frac{1659}{2308}, \cdots$$

Поочередно проверяя знаменатель убедимся что *d=569.*

2.4.2.Криптосистемы, основанные на задаче о рюкзаке

В основу одной из наиболее ранних криптосистем с открытым ключом была заложена задача о рюкзаке, которая считалась трудной при общем выборе начальных данных. Фактически, она принадлежит классу $\mathcal{NP}$ -полных задач, однако, можно показать, что криптосистема, основанная на ней, не является стойкой.

Идея состоит в выборе двух наборов параметров задачи о рюкзаке: открытый набор делает задачу трудноразрешимой, а секретный легкой. Кроме того, должна существовать некоторая дополнительная функция-ловушка, с помощью которой трудная задача трансформируется в легкую.

Такой подход сильно напоминает предположения в криптосистеме *RSA:* очень трудно извлечь корень степени E из заданного числа по составному модулю, но задача становится почти тривиальной, если модуль — простое число. Обладание дополнительной информацией, а именно, разложением модуля криптосистемы на простые делители, позволяет свести сложную задачу к легкой. Однако существенное различие между задачей *RSA* и проблемой рюкзака состоит в том, что при подходящем выборе разложение модуля на множители — задача трудная в среднем, а найти параметры для трудной в среднем проблемы рюкзака очень сложно, несмотря на то, что общая задача о рюкзаке труднее общей задачи факторизации.

В то время, как общая задача о рюкзаке очень сложна, существует большой класс начальных данных, при которых задача об укладке решается быстро. Эти параметры относятся к так называемым *прогрессивно возрастающим* последовательностям. Параметры для простых задач выбираются так, чтобы каждый из *следующих* весов был больше суммы всех предыдущих. Например, последовательность

$$w_i \geqslant \sum_{j=1}^{i-1} w_j.$$

{2, 3, 6, 13, 27, 52}

обладает этим свойством. Можно взять и такие веса;

$$\{1,2,4,8,16,32,64,\ldots\}.$$

Если веса в задаче о рюкзаке обладают этим свойством, то выбор очередного веса, который следует положить в рюкзак - линейная операция. Продемонстрируем это на алгоритме:

```
For(i=n; i>=1; i--)
{
If(S>=w[i])
{
b[i]=1;
S=S-w[i];
}
Else
{
b[i]=0;
}
}
If(S==0)
{напечатать (b[1],b[2],…,b[n])}
Else
Напечатать «нет решения».
```

Обычно последовательность весов содержит 250 значений. Параметры m и n выбираются из 400-битовых чисел. Но даже увеличив параметры, мы не получим стойкой криптосистемы, поскольку она может быть взломана с помощью приведенных базисов решоток.

Определим плотность множества весов $\{w_1,\ldots,w_n\}$ в задаче о рюкзаке формулой:

$$d = \frac{n}{\max\{\log_2 w_i \mid 1 \leqslant i \leqslant n\}}.$$

Покажем, что задача о рюкзаке с небольшой плотностью легко решается с помощью приведения базисов. Так как конструкция Меркля - Хеллмана всегда дает последовательность весов с низкой плотностью, то соответствующую криптосистему несложно взломать.

Итак, пусть $\{W_1, \ldots, W_n\}$ — веса предметов и S — окончательный вес рюкзака, который нужно получить. Рассмотрим N +1-мерную решетку L, порожденную столбцами матрицы

$$A = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 1/2 \\ 0 & 1 & 0 & \dots & 0 & 1/2 \\ 0 & 0 & 1 & \dots & 0 & 1/2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 1/2 \\ W_1 & W_2 & W_3 & \dots & W_N & S \end{pmatrix}$$

Предположим, что строка бит ($b_1, \dots, b_n$) — решение нашей задачи.
Вектор

$$y = A \cdot x$$

при $x = (b_1, \dots, b_n, -1)$ принадлежит решетке. Его координаты имеют вид

$$y_i = \begin{cases} b_i - \frac{1}{2}, & 1 \leqslant i \leqslant N, \\ 0, & i = N + 1. \end{cases}$$

Значит, вектор y имеет очень маленькую длину, поскольку

$$||y|| = \sqrt{y_1^2 + \dots + y_{N+1}^2} = \frac{\sqrt{N}}{2}.$$

Решетка, построенная по задаче о рюкзаке с низкой плотностью, как правило, обладает относительно большим детерминантом. Поэтому y — хороший кандидат на наименьший вектор в решетке. Применив LLL-алгоритм к матрице A, получим матрицу A' приведенного базиса. Ее первый столбец A_1 тоже обычно имеет наименьшую длину. Следовательно, вероятность равенства $A_1 = y$ весьма высока. Получив y, мы сможем найти x и решить исходную задачу о рюкзаке.

ПРАКТИЧЕСКАЯ ЧАСТЬ

В практической части реализована программа, демонстрирующая работу криптосистем с открытым ключом: RSA и систему Меркля-Хеллмана. Также в программе реализованные атаки на эти системы и показаны способы защиты от них.

1.Используемые инструменты

Программа написана на языке программирования C# в среде разработки Microsoft Visual Studio 2008.

При создании программы использованы следующие библиотеки:

using System;
using System.ComponentModel;
using System.Text;
using System.Windows.Forms;
using Oyster.Math;

В пространство **System** вложен целый ряд других пространств имен.

using System.ComponentModel предназначено для контроля работы компонентов.

using System.Text содержит в себе методы для работы с текстовыми данными, например для перевода из в различные кодировки.

using System.Windows.Forms этот класс создает форму, т.е. непосредственно, отображаемый интерфейс программы.

using Oyster.Math класс предназначенный для работы с большими числами.

2.Описание программы

Программа состоит из двух частей.

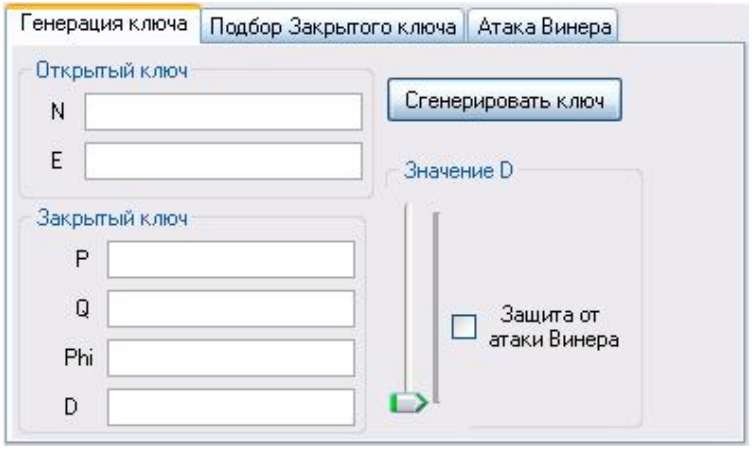
-Исследование RSA

-Исследование Меркля-Хеллмана.

Первая часть включает в себя следующие разделы:

1.Генерация ключа.

Данный раздел предназначен для создания пары открытый/закрытый ключ системы RSA.

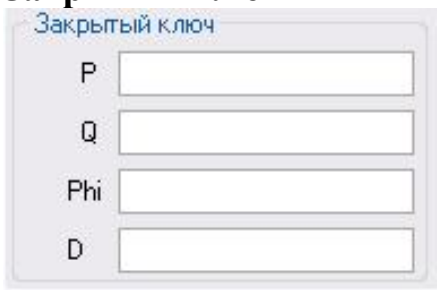


Форма этого раздела имеет три блока:

Открытый ключ

В котором отображаются значение открытых переменных ключа. N и E.

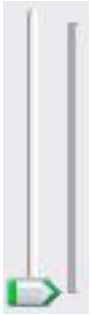
Закрытый ключ



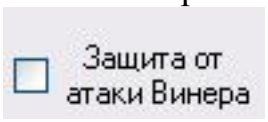
Здесь будут записываться все закрытые данные о ключе. Такие как простые множители числа N, функция Эйлера от N. И секретная экспонента D.

Значение D

Блок включает в себя ползунок, регулирующий размер значения секретной экспоненты D

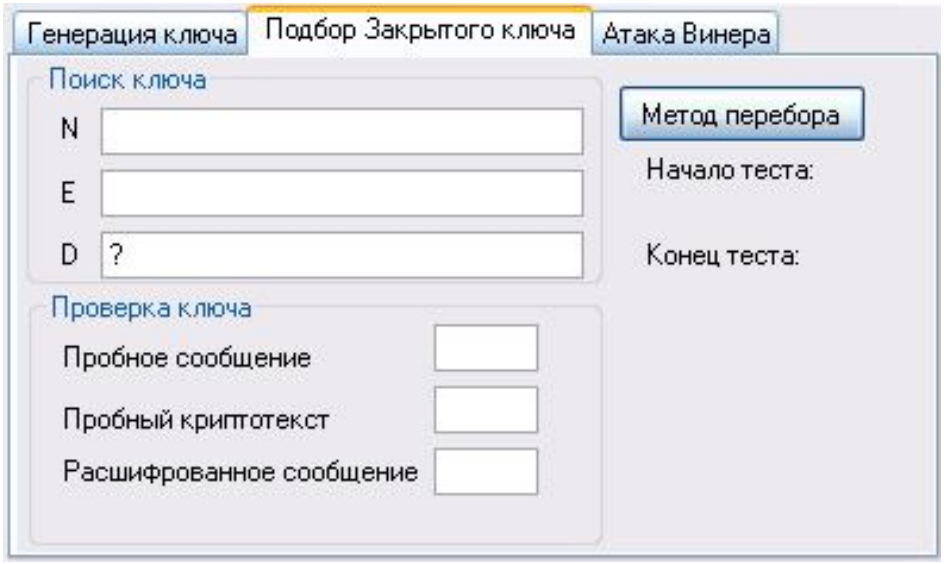


А также переключатель, позволяющий генерировать ключи, защищенные от атаки Винера.

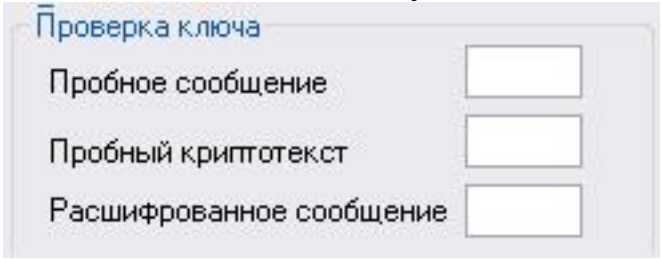


2.Подбор закрытого ключа

В этом разделе реализована атака грубой силой на алгоритм. По значению модуля шифрования N и открытой экспоненты E программа методом подбора находит секретную экспоненту D.



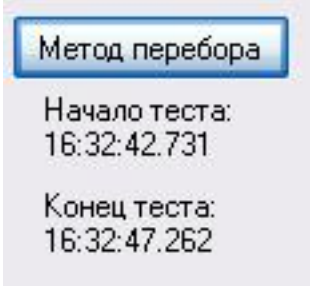
В блок поиск ключа, атакующий вводит N и E.



В блок проверка ключа вместо пробного сообщения вводится произвольное число K.

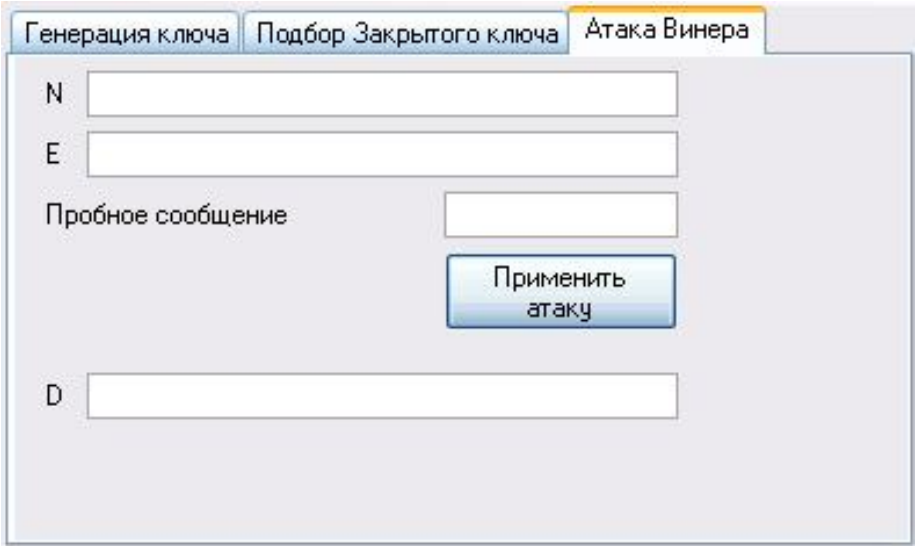
Это число будет шифроваться известным нам открытым ключом и расшифровываться подбираемым произвольным числом. Как только расшифрованное число будет равняться K, атакующий получит значение секретной экспоненты D.

Время работы алгоритма отображается справа на форме



3.Атака Винера

Третий раздел, реализующий атаку, основанную на непрерывных дробях

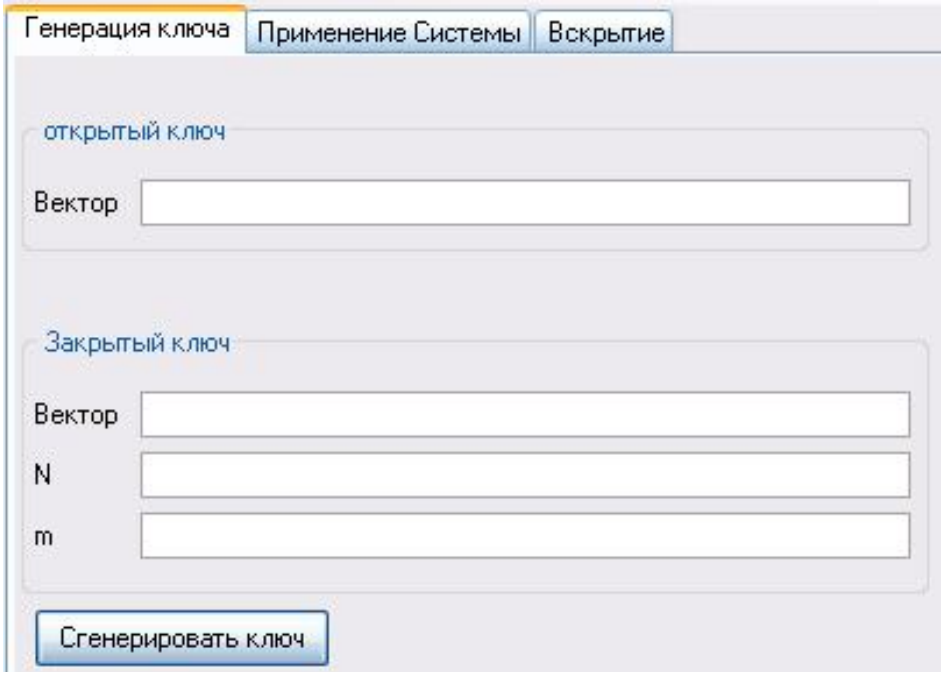


Атакующий опять вводит Открытый ключ и пробное сообщение. Алгоритм работает до тех пор пока расшифрованные данные не будут равны пробному сообщению. Алгоритм работает за полиномиальное время.

Вторая часть включает в себя:

1.Генерация ключа.

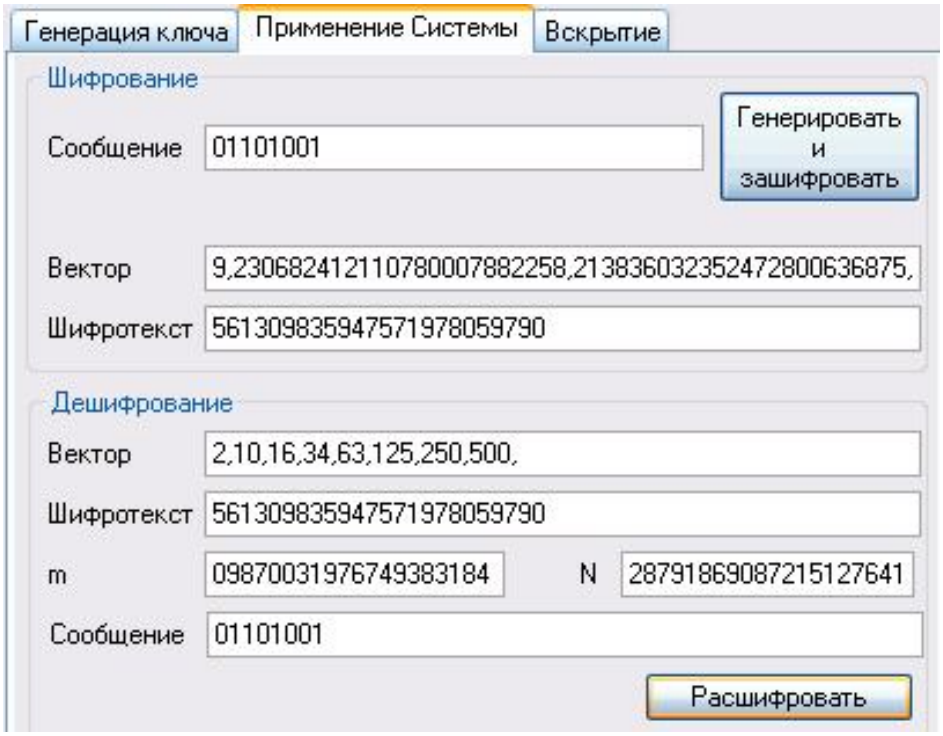
Здесь происходит формирование пары открытый/закрытый ключ.



Открытым ключом является сверхрастущий вектор A.

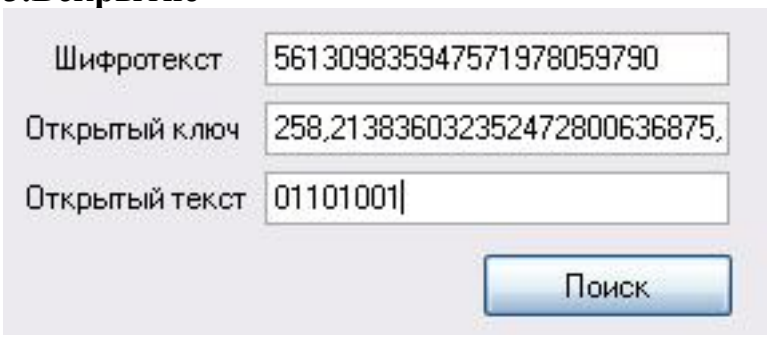
Закрытый ключ состоит из двух взаимнопростых чисел m и N, а также вектора $B=A*m(\mathrm{mod}\ N)$.

2.Применение системы



В этом разделе показывается пример работы системы Меркля-Хеллмана.
При нажатии на кнопку «Генерировать и зашифровать» происходит случайная генерация одного байта данных и их шифрование.
Дешифрование восстанавливает из криптотекста исходные данные по секретному ключу.

3.Вскрытие



Нахождение исходного сообщения только по известному открытому ключу.

БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

1.Шум как вредный производственный фактор

ПРОИЗВОДСТВЕННЫЙ ШУМ Интенсивное шумовое воздействие на организм человека неблагоприятно влияет на протекание нервных процессов, способствует развитию утомления, изменениям в сердечно-сосудистой системе и появлению шумовой патологии, среди многообразных проявлений которой ведущим клиническим признаком является медленно прогрессирующее снижение слуха по типу кохлеарного неврита. В производственных условиях источниками шума являются работающие станки и механизмы, ручные механизированные инструменты, электрические машины, компрессоры, кузнечно-прессовое, подъемно-транспортное, вспомогательное оборудование (вентиляционные установки, кондиционеры) и т.д. Допустимые шумовые характеристики рабочих мест регламентируются. По характеру спектра шумы подразделяются на широкополосные и тональные. По временным характеристикам шумы подразделяются на постоянные и непостоянные. В свою очередь непостоянные шумы подразделяются на колеблющиеся во времени, прерывистые и импульсные. В качестве характеристик постоянного шума на рабочих местах, а также для определения эффективности мероприятий по ограничению его неблагоприятного влияния, принимаются уровни звукового давления в децибелах (дБ) в октавных полосах со среднегеометрическими частотами 31,5; 63; 125; 250; 1000; 2000; 4000; 8000 Гц. В качестве общей характеристики шума на рабочих местах применяется оценка уровня звука в дБ(А), представляющая собой среднюю величину частотных характеристик звукового давления. Характеристикой непостоянного шума на рабочих местах является интегральный параметр - эквивалентный уровень звука в дБ(А). Основные мероприятия по борьбе с шумом - это технические мероприятия, которые проводятся по трем главным направлениям: - устранение причин возникновения шума или снижение его в источнике; - ослабление шума на путях передачи; - непосредственная защита работающих. Наиболее эффективным средством снижения шума является замена шумных технологических операций на малошумные или полностью бесшумные, однако этот путь борьбы не всегда возможен, поэтому большое значение имеет снижение его в источнике. Снижение шума в источнике достигается путем совершенствования конструкции или схемы той части оборудования, которая производит шум, использования в конструкции материалов с пониженными акустическими свойствами, оборудования на источнике шума дополнительного звукоизолирующего устройства или ограждения, расположенного по возможности ближе к источнику. Одним из наиболее простых технических средств борьбы с шумом на путях передачи является звукоизолирующий кожух, который может закрывать отдельный шумный узел машины. Значительный эффект снижения шума от оборудования дает применение акустических экранов, отгораживающих шумный механизм от рабочего места или зоны обслуживания машины. Применение звукопоглощающих облицовок для отделки потолка и стен шумных помещений приводит к изменению спектра шума в сторону более низких частот, что даже при относительно небольшом снижении уровня существенно

улучшает условия труда. Учитывая, что с помощью технических средств в настоящее время не всегда удается решить проблему снижения уровня шума большое внимание должно уделяться применению средств индивидуальной защиты (антифоны, заглушки и др.). Эффективность средств индивидуальной защиты может быть обеспечена их правильным подбором в зависимости от уровней и спектра шума, а также контролем за условиями их эксплуатации.

Шум — один из наиболее распространенных неблагоприятных физических факторов окружающей среды, приобретающих важное социально-гигиеническое значение, в связи с урбанизацией, а также механизацией и автоматизацией технологических процессов, дальнейшим развитием авиации, транспорта.

Например, при запуске реактивных двигателей самолетов уровень шума колеблется от 120 до 140 дБ, при клепке и рубке листовой стали— от 118 до 130 дБ, работе деревообрабатывающих станков — от 100 до 120 дБ, ткацких станков — до 105 дБ; бытовой шум, связанный с жизнедеятельностью людей, составляет 45—60 дБ.

Длительное воздействие шума на организм человека приводит к развитию утомления, нередко переходящего в переутомление, к снижению производительности и качества труда. Особенно неблагоприятно шум действует на орган слуха, вызывая поражение слухового нерва с постепенным развитием тугоухости. Как правило, оба уха страдают в одинаковой степени. Начальные проявления профессиональной тугоухости чаще всего встречаются у лиц со стажем работы в условиях шума около 5 лет. Риск потери слуха у работающих при 10-летней продолжительности воздействия шума составляет 10% при уровне 90 дБ (шкала А), 29% — при 100 дБ (шкала А) и 55% — при 110 дБ (шкала А).

Неспецифическое воздействие шума обычно проявляется раньше, чем изменения в органе слуха, и выражается в нарушениях нервно-психической сферы в форме невротического и астенического синдрома в сочетании с вегетативной дисфункцией, сопровождающихся раздражительностью, общей слабостью, головной болью, головокружением, повышенной утомляемостью, расстройством сна, ослаблением памяти и др. Не исключена возможность развития нейроциркуляторного синдрома, преимущественно по гипертоническому типу. У лиц, подвергающихся воздействию шума, также могут наблюдаться изменения секреторной и моторной функций желудочно-кишечного тракта, сдвиги в обменных процессах — нарушение основного, витаминного, углеводного, белкового, жирового и солевого обменов, нарушения функционального состояния сердечно-сосудистой системы в виде брадикардии, повышения тонуса периферических сосудов и др.

В производственных условиях воздействие шума на работающих обычно сочетается с рядом других неблагоприятных факторов — вибрацией, определенной степенью напряженности и тяжести труда, неудовлетворительными микроклиматическими условиями, воздействием химических веществ, инфразвука и ультразвука, электромагнитного поля и др. Шум может усугублять неблагоприятное воздействие сопутствующих факторов физической и

химической природы, оказывая, прежде всего отрицательное влияние на состояние здоровья и работоспособность профессиональных групп, труд которых сопровождается нервным напряжением.

Эффективная защита работающих от неблагоприятного влияния шума требует осуществления комплекса организационных, технических и медицинских мер на этапах проектирования, строительства и эксплуатации производственных предприятий, машин и оборудования. В целях повышения эффективности борьбы с шумом введены обязательный гигиенический контроль объектов, генерирующих шум, регистрация физических факторов, оказывающих вредное воздействие на окружающую среду и отрицательно влияющих на здоровье людей.

Эффективным путем решения проблем борьбы с шумом является снижение его уровня в самом источнике за счет изменения технологии и конструкции машин. Большое значение в борьбе с шумом имеют архитектурно-планировочные и строительные мероприятия. В тех случаях, когда технические способы не обеспечивают достижения требований действующих нормативов, необходимо ограничение длительности воздействия шума и применение противошумов — специальных приспособлений для индивидуальной защиты от шума. Эти приспособления снижают уровень громкости шума, но не мешают восприятию необходимых команд и сигналов. Противошумы по назначению и конструктивному исполнению подразделяют на три типа: вкладыши, наушники и шлемы. Выбор индивидуальных средств защиты органа слуха зависит от мощности шумов, спектрального их состава, времени действия за рабочую смену. Противошумы следует применять с первого дня пребывания в шумной обстановке, что способствует предотвращению нарушений слуха и возникновению других неблагоприятных эффектов, связанных с воздействием шума.

Работающие в условиях интенсивного шума подлежат предварительным и периодическим медосмотрам с целью выявления противопоказаний для работы, связанной с шумом, и ранних форм профзаболевания.

2.Пожарная профилактика

Ответственные органы и их обязанности. Пожарная профилактика традиционно ограничивалась обучением технике безопасности и мерами по предупреждению пожаров и всегда входила в обязанности муниципальных управлений пожарной охраны. Сегодня круг мероприятий по пожарной профилактике расширен, и в него вошли проверка и утверждение проектов строительства, контроль за выполнением норм по пожарной безопасности, борьба с поджогами (в т.ч. с пожароопасными играми подростков), сбор данных, а также инструктаж и обучение широкой общественности и специальных контингентов.

Задачи пожарной профилактики можно разделить на три широких, но тесно связанных комплекса мероприятий:

1) обучение, в т.ч. распространение знаний о пожаробезопасном поведении (о необходимости установки домашних индикаторов задымленности и хранения зажигалок и спичек в местах, недоступных детям);

2) пожарный надзор, предусматривающий разработку государственных норм пожарной безопасности и строительных норм, а также проверку их выполнения;
3) обеспечение оборудованием и технические разработки (установка переносных огнетушителей и изготовление зажигалок безопасного пользования).

Из трех перечисленных комплексов мероприятий сложнее всего, по-видимому, пожарный надзор. В сферу надзора включены нормы пожарной профилактики, строительные пожарные нормы и правила, стандарты изготовления и установки противопожарного оборудования и стандарты пожарной безопасности на товары широкого потребления.

Противопожарная защита

Мероприятия по противопожарной защите включают:
1) контроль материалов, продуктов и оборудования;
2) активное ограничение распространения огня с использованием средств пожарной сигнализации, систем автоматического пожаротушения и переносных огнетушителей;
3) устройство пассивных систем, ограничивающих распространение огня, дыма, жара и газов за счет секционирования помещений;
4) эвакуацию людей из горящего здания в безопасное место.

Системы пожарной сигнализации. В случае возгорания должна сразу же сработать система пожарной сигнализации, за которой следует регламентированная система мероприятий.

Автоматические системы пожаротушения. Применяются жидкостные, углекислотные, порошковые и пенные автоматические системы пожаротушения.

Наиболее распространенная водяная система – это просто система водопроводных труб, оканчивающихся спринклерными головками с термочувствительными клапанами. Под действием тепла клапан спринклерной головки открывается, и из нее бьет струя воды, широко разбрызгиваемая механическими отражателями. Каждая головка срабатывает индивидуально в соответствии с температурой в месте ее расположения. (Иначе работают заливающие системы, о которых будет сказано ниже.) Чтобы система работала нормально, спринклерные головки не должны быть залиты краской, на них не должны висеть посторонние предметы и пространство вокруг них не должно быть загромождено.

В «мокрых» водяных системах пожаротушения трубопроводы всегда наполнены водой под давлением. В «сухих» системах трубопроводы заполнены сжатым воздухом или азотом, пока не откроется спринклерная головка, после чего давление в трубе падает и вода начинает поступать с напорной стороны. В системах предваряющего действия сигнализатор пожара открывает клапан и наполняет трубы водой, прежде чем откроется спринклерная головка. Иногда принципы сухой системы и предваряющего действия сочетаются в одной системе. В заливающих системах спринклерные головки всегда открыты, а сигнализатор пожара управляет общим водяным клапаном, так что при возгорании вода поступает сразу во все спринклерные головки. Предусматриваются также

специальные водяные системы для защиты наружных стен здания и для других особых задач.

Водяные автоматические системы пожаротушения бытового назначения выпуска конца века требуют столь малых количеств воды, что одной спринклерной головки достаточно для площади почти 40 м2. Распределение разбрызгиваемой воды таково, что при работе спринклерной головки обеспечивается защита мебели в углах комнат и даже потолка.

Противопожарные преграды. Устройство противопожарных преград, предупреждающих распространение огня из одной части здания в смежные помещения, представляет собой крайне важную меру обеспечения пожарной безопасности. Широкое распространение огня влечет за собой увеличение человеческих потерь и материального ущерба, а также резко затрудняет борьбу с пожаром. Хотя огонь лишь в 20% домашних пожаров выходит за пределы того этажа, где произошло возгорание, на эти 20% приходится почти 70% погибших, более 30% пострадавших и около 70% имущественных потерь. Дым и газы могут распространяться гораздо дальше, чем пламя, создавая опасность для людей, находящихся далеко от зоны огня. Дым и газообразные продукты горения даже от небольшого огня могут вызывать повреждение или перебои в работе компьютеров и другого чувствительного электронного оборудования.

Противопожарные преграды – это специальные противопожарные стены (брандмауэры) и несгораемые перекрытия, спроектированные так, чтобы они препятствовали распространению пламени, дыма и жара по горизонтали (из помещения в помещение на том же этаже) или по вертикали (с этажа на этаж) внутри здания, а также от одного здания к другому. Предусматриваются также противопожарные зоны – участки здания, выполненные из огнестойких материалов и разделяющие его на изолированные секции. Такие зоны без огня и дыма могут служить убежищем для людей, застигнутых в здании пожаром.

Эвакуация людей. При пожарах в зданиях в первую очередь должна решаться задача защиты людей путем их эвакуации в безопасную зону. В некоторых случаях вместо немедленной эвакуации применяется метод «защиты на месте» – люди временно укрываются во внутренней противопожарной зоне. Такие зоны (холлы перед лифтами, расширенные лестничные клетки), защищенные автоматическими системами пожаротушения, отделяются от смежных зон свободными промежутками или дымонепроницаемыми и огнестойкими ограждающими конструкциями.

Система эвакуации должна давать людям возможность выхода в безопасную зону во время пожара. Она должна обеспечивать непрерывный ничем не перекрываемый путь выхода из любого места здания на улицу и предусматривать специальные легко открывающиеся дверные запоры, горизонтальные выходы, междуэтажные лестницы, дымонепроницаемые шахты, пожарные лестницы, эскалаторы, горизонтальные пассажирские транспортеры, лифты, окна, эвакуационное освещение и выходные знаки.

Владельцы зданий и жильцы нередко загромождают коридоры, двери и лестничные клетки различными хранимыми вещами, что при пожаре может

привести к трагическим последствиям. Поэтому в интересах общественной безопасности нужно всегда сообщать о блокированных или закрытых выходах администрации здания или пожарному управлению.

Большое внимание системам эвакуации уделяется в нормативах по технике безопасности и строительных нормах и правилах. В частности, регламентируются минимальная ширина дверей и коридоров, максимальная длина тупиковых коридоров, ширина и наклон лестничных маршей. Для облегчения и ускорения эвакуации необходимо, чтобы двери открывались в сторону выхода. Специальные дверные запоры должны открываться при легком нажатии.

Эвакуационное освещение должно способствовать ускорению эвакуации. Нормативами регламентируются освещенность у дверей и расположение светильников. В некоторых случаях должно предусматриваться аварийное освещение с питанием от автономного электрогенератора или аккумуляторной батареи. Кроме того, в некоторых случаях обязательна установка светящихся надписей «ВЫХОД».

Нормативы эксплуатации зданий требуют, чтобы управляющие высотными общественными, коммерческими и промышленными зданиями проводили периодические учебные пожарные тревоги, в ходе которых персонал зданий и пожарная охрана лучше знакомились бы со своими задачами и обязанностями в случае пожара. Кроме того, такие учения дают возможность остальным работникам, посетителям и жильцам узнать, кто входит в пожарную охрану их здания, усвоить правила техники безопасности, запомнить схему эвакуации, расположение кнопок сигнализации, переносных огнетушителей и пожарных лестниц. Многие нормативы рекомендуют проживающим в отдельных домах тоже периодически проводить учебные пожарные тревоги, чтобы все члены семьи знали пути выхода и места встречи, а также была проверена работоспособность домашних индикаторов задымленности.

ЗАКЛЮЧЕНИЕ

В данной работе я показала, что «Стойкость криптосистемы» сложное понятие, подходить к исследованию которого следует с применение совокупности различных методов.

Так на примере системы Меркля-Хеллмана в этой работе доказано, что даже самая стойкая с теоретико-информационной точки зрения криптосистема может быть легко взломана на практике.

С другой стороны применение таких изученных и надежных алгоритмов как RSA и Эль-Гамаль не дает совершенно никакой уверенности в том, что зашифрованные данные надежно укрыты. И поэтому умение использовать криптографические методы, на практике, также важно, как и теоретические методы оценки стойкости системы.

Вообще говоря о стойкости ассиметричных криптосистем, следует понимать, что все доказательства их стойкости сводятся к трудности решения какой-либо математической задачи.

Однако доказательства, того, что используемые в асимметричной криптографии задачи, такие как факторизация числа, дискретный логарифм, не разрешимы, не существует. Поэтому современная криптография с открытым ключом практически не имеет инструментария для полноценного доказательства стойкости. Наиболее совершенным инструментом доказательства сегодня считается «модель случайного оракула», описанная в этой работе. С использованием «случайного оракула» сегодня доказана стойкость таких распространенных схем как DSA и RSA-OAEP.

В этой работе реализовано несколько атак на распространенные криптосистемы с открытым ключом и показано что наивное использование шифрующих функций не самая удачная идея.

В работе показано, что для того, чтобы повысить уровень стойкости современных криптоалгоритмов некоторое время применялись элементы теории сложности, однако, приведенные в работе уязвимости дискредитируют такой подход. И при разработке криптосистем, в настоящее время, лучше пользоваться элементами доказуемой стойкости.

ЛИТЕРАТУРА

1. «Криптография» Н. Смарт
2. «Криптография с открытым ключом» А. Саломааа
3. «Handbook of applied cryptography» A. Menezes
4. «Курс теории чисел и криптографии» Н. Коблиц
5. «Полемика по поводу понятий криптографической безопасности» Д. Стинсон
6. «Modern cryptography, probabilistic proof and pseudo-randomness» O. Goldreich
7. «Прикладная криптография» Б. Шнайер
8. «Практическая криптография» Б. Шнайер, Н. Фергюсон
9. «Информационная безопасность» В. Шаньгин
10. «Современная криптография: теория и практика» Венбо Мао