МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

01.03.02 ПРИКЛАДНАЯ МАТЕМАТИКА И ИНФОРМАТИКА

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

БАКАЛАВРСКАЯ РАБОТА

на тему Сравнительный анализ реализаций протокола обмена ключами Диффи - Хеллмана

Студент А.М. Войтович _____

Руководитель Г.А. Тырыгина _____

Допустить к защите
Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____

«_____»_____20_____г.

Тольятти 2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная
математика и информатика»
_____А.В.Очеповский

«____»_____2016 г.

ЗАДАНИЕ
на выполнение бакалаврской работы

Студент Войтович Алексей Михайлович
1. Тема
Сравнительный анализ реализаций протокола обмена ключами Диффи - Хеллмана
2. Срок сдачи студентом законченной бакалаврской работы
24.06.2016
3. Исходные данные к выпускной квалификационной работе
Протокол обмена ключами Диффи - Хеллмана
4. Содержание бакалаврской работы (перечень подлежащих разработке вопросов, разделов)
Введение
Глава 1 Протокол обмена ключами Диффи-Хеллмана
1.1 Общие сведения о протоколе
1.2 Математическое обоснование протокола обмена ключами Диффи-Хеллмана
1.3 Применение протокола Диффи-Хеллмана в SSL
1.4 Применение протокола Диффи-Хеллмана в SSH
1.5 Применение протокола Диффи-Хеллмана в IPSec
1.6 Применение протокола Диффи-Хеллмана в PKI
1.7 Использование дополнительных средств для увеличения надежности протокола Диффи-Хеллмана
1.8 Уязвимость протокола Диффи-Хеллмана
1.8.1 Атака посредника
1.8.2 Атаки, основанные на теории чисел.
1.8.3 Проверки подлинности
1.8.4 Атака на параметры проверки подлинности

1.9 Протокол Диффи-Хеллмана с тремя и более участниками
1.10 Задача Диффи–Хеллмана и задача дискретного логарифмирования
Глава 2 Реализация и тестирование протокола
2.1 Разработка реализации протокола обмена ключами Диффи-Хеллмана на языке программирования C++
2.2 Разработка реализации протокола обмена ключами Диффи-Хеллмана на языке программирования Java
2.3 Тестирование протокола
Глава 3 Сравнительный анализ реализаций
3.1 Подсчет времени вычисления ключей для протокола Диффи-Хеллмана
3.2 Практическое сравнение реализаций
Заключение
Список используемой литературы
5. Ориентировочный перечень графического и иллюстративного материала Презентация, графики, рисунки
6. Дата выдачи задания «11» января 2016 г.

Руководитель бакалаврской работы _____ Г.А. Тырыгина

Задание принял к исполнению _____ А.М. Войтович

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**

Кафедра **«Прикладная математика и информатика»**

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная
математика и информатика»
_____А.В.Очеповский

«____»_____2016 г.

КАЛЕНДАРНЫЙ ПЛАН
выполнения бакалаврской работы

Студента Войтович Алексея Михайловича
по теме Сравнительный анализ реализаций протокола обмена ключами Диффи – Хеллмана

Наименование раздела работы	Плановый срок выполнения раздела	Фактический срок выполнения раздела	Отметка о выполнении	Подпись руководителя
Изучение теоретических основ	20.01.2016	20.01.2016	Выполнено	
Написание введения бакалаврской работы	01.02.2016	01.02.2016	Выполнено	
Изучение математических основ	25.02.2016	25.02.2016	Выполнено	
Написание 1 главы бакалаврской работы	23.03.2016	23.03.2016	Выполнено	
Написание кода программ	6.04.2016	6.04.2016	Выполнено	
Тестирование реализаций	8.04.2016	8.04.2016	Выполнено	
Написание 2 главы бакалаврской работы	14.04.2016	14.04.2016	Выполнено	

Проведение сравнительного анализа реализаций	23.04.2016	23.04.2016	Выполнено	
Написание 3 главы бакалаврской работы	30.04.2016	30.04.2016	Выполнено	
Написание заключения бакалаврской работы	8.05.2016	8.05.2016	Выполнено	
Подведение итогов, редактирование бакалаврской работы	9.05.2016	9.05.2016	Выполнено	
Представление бакалаврской работы	12.05.2016	12.05.2016	Выполнено	
Создание презентационного материала	15.05.2016	15.05.2016	Выполнено	
Предварительная защита	31.05.2016 - 14.06.2016	31.05.2016	Выполнено	
Проверка на наличие заимствований (плагиата) в системе antiplogiat.ru	17.06.2016	17.06.2016	Выполнено	
Сдача на кафедру отзыва научного руководителя и ознакомление с ним	20.06.2016	20.06.2016	Выполнено	
Сдача на кафедру комплекта документов для защиты	24.06.2016	24.06.2016	Выполнено	
Защита бакалаврской работы	27.06.2016 - 30.06.2016	29.06.2016	Выполнено	

Руководитель бакалаврской работы _____ Г.А. Тырыгина

Задание принял к исполнению _____ А.М. Войтович

Аннотация

Темой данной бакалаврской работы является «Сравнительный анализ реализаций протокола обмена ключами Диффи-Хеллмана».

Работа выполнена студентом Тольяттинского государственного университета, института математики, физики и информационных технологий, группы ПМИб-1201, Войтович Алексея Михайловича.

Объект работы: протокол обмена ключами Диффи-Хеллмана.

Предмет исследования: сравнительный анализ двух реализаций протокола обмена ключами Диффи-Хеллмана языках программирования C++ и Java.

Цель работы: написание реализаций, на основе протокола обмена ключами Диффи-Хеллмана. Для ее решения были поставлены следующие задачи:

1. Изучить криптографический алгоритм.
2. Написать реализации на основе протокола распределения ключей Диффи-Хеллмана используя разные языки программирования.
3. Произвести проверку работоспособности и корректности работы программы.
4. Произвести сравнительный анализ обеих реализаций протокола и выявить возможные преимущества.

Отчет состоит из введения, трех глав и заключения.

В первой главе представлены теоретические аспекты протокола обмена ключами Диффи-Хеллмана.

Во второй главе производится реализация и тестирование протокола.

В третьей главе производится сравнительный анализ реализаций.

Бакалаврская работа представлена на 40 страницах, включает 11 иллюстраций, 1 приложение, список используемой литературы содержит 20 источников.

Оглавление

Введение

Сегодня мир превратился в глобальный рынок, где электронная коммерция и ERP снова вынудили корпоративный мир задуматься. Преимущества новых технологий вынуждают предприятия изучать и адаптироваться к ним. Но с развитием технологии возникает необходимость в принятии дополнительных мер безопасности для защиты конфиденциальных данных от посторонних лиц, как перехватчика. Тем не менее, главным средством передачи важной информации является интернет, где также присутствуют злоумышленники. Следовательно, очень важно выяснить лазейки в безопасности и разработать контрмеры, которые могут быть приняты заранее для предотвращения потери данных. Существует множество мер противодействия, которые используются в наше время, например: криптография, стеганография, межсетевые экраны, списки доступа, прокси-шлюзы приложений, протоколы безопасности, такие как SSL, TLS, и т.д.

Криптография является лучшей стратегией безопасности, которая развивалась на протяжении десятилетий, особенно после внедрения и распространения компьютеров. Криптография широко известна как наука о кодировании данных для предотвращения несанкционированного доступа через незащищенные каналы связи. Шифрование подразделяется на три категории: симметричное шифрование, асимметричное шифрование и хэш-функции. Секретный ключ и симметричное шифрование развивались первыми. Симметричное шифрование использует один секретный или личный ключ для шифрования и дешифрования данных. Процесс расшифровки прямо противоположен процессу шифрования. Тем не менее, одной из основных проблем является передача ключа для расшифровки через интернет. Это основное препятствие, потому что если ключ будет перехвачен злоумышленниками, то зашифрованные данные могут быть легко расшифрованы.

Эта проблема передачи ключа для симметричного шифрования была

затронута в научно-исследовательском проекте, проведенном Уитфилдом Диффи и Мартином Хеллманом из Массачусетского технологического института в 1976 году. Алгоритм, разработанный ими, широко известен как алгоритм Диффи-Хеллмана, который используется для безопасного обмена секретным ключом между двумя сторонами в режиме реального времени в незащищенной сети. Общий секретный ключ очень важен для двух сторон, которые, возможно, не общались раньше, поэтому они могут зашифровать свои сообщения. По сути это протокол согласования ключа, который поддерживает секретность между двумя сторонами для обмена ключами. Согласование ключей представляет собой метод, в котором устройства передачи данных в сети устанавливают общий секретный ключ между ними, без обмена какими-либо секретными данными. В этом методе устройствам необходимо обменяться своими открытыми ключами. Оба устройства после приема открытых ключей выполняют операцию генерации ключа с использованием своего секретного ключа, чтобы получить общий секретный ключ. Благодаря своим превосходным атрибутам безопасности, в наше время алгоритм широко используется и имеет множество модификаций. Некоторые исследователи изменили алгоритм Диффи-Хеллмана и использовали его в таких протоколах безопасности, как «уровень защищённых сокетов» (SSL), «протокол безопасности межсетевого взаимодействия» (IPSec) и т.д.

Тем не менее, некоторые недостатки алгоритма Диффи-Хеллмана все еще существуют. Одним из недостатков алгоритма Диффи-Хеллмана является объем вычислений, в связи с этим увеличивается временная сложность при генерации открытых ключей.

Актуальность:

В наше время существует проблема обеспечения защиты конфиденциальности данных при общении через незащищенную сеть интернет. Одним из аспектов этой проблемы является невозможность безопасной передачи секретного ключа. В решение данного аспекта помогает протокол обмена ключами Диффи-Хеллмана. Этот протокол задействован во многих системах

защиты информации, предоставляющих возможность безопасно передавать данные через незащищенные каналы передачи информации, таких как интернет.

Объект работы: протокол обмена ключами Диффи-Хеллмана.

Предмет исследования: сравнительный анализ двух реализаций протокола обмена ключами Диффи-Хеллмана языках программирования C++ и Java.

Цель работы: написание реализаций, на основе протокола обмена ключами Диффи-Хеллмана. Для ее решения были поставлены следующие задачи:

1. Изучить криптографический алгоритм.
2. Написать реализации на основе протокола распределения ключей Диффи-Хеллмана используя разные языки программирования.
3. Произвести проверку работоспособности и корректности работы программы.
4. Произвести сравнительный анализ обеих реализаций протокола и выявить возможные преимущества.

Отчет состоит из введения, трех глав и заключения.

В первой главе представлены теоретические аспекты протокола обмена ключами Диффи-Хеллмана.

Во второй главе производится реализация протокола и тестирование.

В третьей главе производится сравнительный анализ реализаций.

Глава 1 Протокол обмена ключами Диффи-Хеллмана

1.1 Общие сведения о протоколе

Шифровальный секретный ключ необходим для передачи данных в ассиметричном шифровании. Решающую роль в этом типе шифрования играет обмен шифровальным ключом без возможности перехвата злоумышленником. Протокол обмена ключами Диффи-Хеллмана позволяет безопасно передавать или обмениваться шифровальным ключом для обеих сторон. Протокол обмена ключами Диффи-Хеллмана был первым протоколом с открытым ключом, опубликованным в 1976 году. Создание первого практического метода, позволяющего передать через незащищенную сеть секретные данные, стало результатом совместных усилий Уитфилда Диффи и Мартина Хеллмана.

Тем не менее, считается, что этот метод был впервые изобретен Малькомом Уильямсом из Великобритании, но данные так и не были опубликованы. Хотя алгоритм Диффи-Хеллмана трудоемкий, это же является сильной стороной алгоритма, что делает его использование в генерации шифровальных ключей популярным [1].

Основная цель алгоритма состоит в том, чтобы дать возможность пользователям безопасно обмениваться ключами, которые могут быть использованы для последующего шифрования. Это гарантирует, что никто кроме пользователя A и B не сможет узнать какую-либо информацию о согласованном значении, а также гарантирует, что соответствующий партнер на самом деле вычислит это значение. Процесс алгоритма Диффи-Хеллмана описывается следующим образом:

Для начала пользователи, Алиса и Боб, договариваются использовать конечное поле F_qи элемент $g \in F_q$, порождающий группу, имеющую большой порядок. Для простоты будем рассматривать поле F_p, в котором число p является большим простым числом. Затем Алиса и Боб могут найти элемент g, порождающий группу F_p^*. Каждое число из интервала $[1, p)$ можно представить в виде

$$g^x(mod\ p), \tag{1.1}$$

где x - некоторое число. Теперь числа p и q можно использовать в качестве общих исходных данных в основном варианте протокола обмена ключами Диффи-Хеллмана [6].

Протокол обмена ключами Диффи–Хеллмана:

Общие исходные данные:

(p, g) : p - большое простое число, g- порождающий элемент группы F_p^*.

Результат:

1. Элемент группы F_p^*, разделенный между Алисой и Бобом.

2. Алиса генерирует элемент $a \in U[1, p-1)$, вычисляет число $g_a \leftarrow g^a(mod\ p)$и отправляет его Бобу.

3. Боб генерирует элемент $b \in U[1, p-1)$, вычисляет число $g_b \leftarrow g^b(mod\ p)$и отправляет его Алисе.

4. Алиса вычисляет значение $k \leftarrow g_b^a(modp)$.

5. Боб вычисляет число $k \leftarrow g_a^b(modp)$.

Из протокола видно, что значение, вычисляемое Алисой, равно

$$k = g^{ba}(mod\ p), \tag{1.2}$$

а значение, вычисляемое Бобом, -

$$k = g^{ab}(mod\ p). \tag{1.3}$$

Заметим, что, поскольку

$$ab \equiv ba(mod\ p), \tag{1.4}$$

обе стороны вычисляют одно и то же значение. Это обеспечивает разделение ключа между двумя сторонами. Таким образом, числа p и gможно разослать всем участникам системы [2].

Описание протокола Диффи–Хеллмана следует дополнить следующими замечаниями:

- простое число p следует выбирать так, чтобы число $p-1$ имело достаточно большой простой множитель p'. Слова “достаточно большой” означают, что

$$p' > 2^{160}; \quad (1.5)$$

• число g не обязано быть порождающим элементом группы F_p^*. Необходимо лишь, чтобы оно было порождающим элементом ее подгруппы, имеющей достаточно большой порядок, например, подгруппы порядка p'. В этом случае Алиса и Боб должны проверить условия

$$g \neq 1 \text{ и } g^{p'} \equiv 1(modp). \quad (1.6)$$

По этой причине число p'должно быть частью общих исходных данных;

• Алиса (соответственно Боб) должна проверить условие

$$g_b \neq 1 \text{ (соответственно } g_a \neq 1 \text{).} \quad (1.7)$$

Это условие гарантирует, что для выбранной ею степени, принадлежащей интервалу (1, p'.), разделенный ключ g^{ab}будет элементом подгруппы порядка p'группы F_p, т.е. элементом достаточно большой подгруппы [5];

• по окончании протокола Алиса (соответственно Боб) должна стереть свою степень a (соответственно степень b). Этим они обеспечивают заблаговременную секретность ключа g^{ab}.

Протокол Диффи-Хеллмана отлично противостоит пассивному нападению, но в случае реализации атаки «человек посередине» он не устоит.

В самом деле, в протоколе ни Алиса, ни Боб не могут достоверно определить, кем является их собеседник, поэтому вполне возможно представить следующую ситуацию, при которой Боб и Алиса установили связь с Злоумышленником, который Алисе выдает себя за Боба, а Бобу представляется Алисой.

1.2 Математическое обоснование протокола обмена ключами Диффи-Хеллмана

При расчетах, необходимых для протокола обмена ключами Диффи-Хеллмана используется понятие группы.

Группа

Группа $(G,*)$ состоит из набора элементов G и бинарной операции $*$, которая берет элементы G как входные параметры. Операция $*$ обладает следующими свойствами:

1. Ассоциативность: $a*(b*c)=(a*b)*c$, для всех $a,b,c\in G$.
2. Нейтральный элемент: Существует элемент $1\in G$, называющийся нейтральным, который обладает свойством $1*a=a*1=a$, для всех $a\in G$.
3. Обратный элемент: Для каждого элемента $a\in G$ существует такой элемент, обозначаемый a^{-1}, что $a*a^{-1}=a^{-1}*a=1$.

Абелева группа представляет собой группу, имеющую дополнительное свойство:

4. Коммутативность:$a*b=b*a$ для всех $a,b\in G$.

Для конечных групп (G конечна) порядок группы определяется как мощность (размер) G. Порядок элемента a конечной группы G определяется как наименьшее значение t такое, что $a^t:=\underbrace{a*a*...*a}_{t}=1$.

Циклическая группа

Циклическая группа представляет собой группу, обладающую свойством, что существует элементg таким образом, что все элементы в G могут быть выражены, как g^i (для различных i). Если g вычисляет все элементы группы $(G,*)$, g является образующей и мы говорим что он образует группу $(G,*)$. Обратите внимание, что порядок образующей g равен порядку образованной им группы [4].

Подгруппа

Мы говорим, что G' является подгруппой G, если $(G',*)$ образует группу и $(G'\subseteq G)$. Если G является конечной группой, то порядок подгруппы G' всегда будет делиться на порядок G.

Примеры групп

Группы, используемые для протокола обмена ключами Диффи-Хеллмана, представляют собой набор элементов Z_p^* с умножением по модулю p, где p представляет собой простое число, мультипликативную группу поля F_{2^m} и

аддитивную группу образованную совокупностью точек определенных эллиптической кривой в конечном поле.

Все эти группы имеют свойство, что операция возведения в степень вычислительно проста и вычисление дискретного логарифма является крайне трудоемким процессом [16].

1.3 Применение протокола обмена ключами Диффи-Хеллмана в SSL

Уровень защищенных сокетов (SSL) представляет собой криптографический протокол, разработанный Netscape в 1995 году. Версия 3.0 протокола SSL стала основным методом и де-факто стандартом для обеспечения безопасности потока информации между пользователями и веб-сервером.

Чаще всего этот метод используется в финансовой или бизнес сфере. «Обеспечение безопасности» в данном случае обозначает конфиденциальность, подлинность и целостность данных. Пользователи могут и не знать, что они пользуются протоколом SSL, но они вероятно заметили значок замка и «HTTPS:» в URL, которые указывают на работу протокола. Обычно протокол SSL использует TCP-порт 443.

Инженерный совет интернета (IETF) принял этот протокол в 1999 году. Протокол был переименован в протокол безопасности транспортного уровня (TSL).

Протоколы SSL и TSL состоят из двух уровней: нижний уровень, называемый протокол записи, зависящий от протокола TCP и управляющий симметричным шифрованием, поэтому связь является конфиденциальной и надежной. Верхний уровень называется протокол подтверждения подключения и отвечает за аутентификация сторон и согласование методов и ключей шифрования. Именно на этом уровне может быть использован протокол обмена ключами Диффи-Хеллмана. Протокол обмена ключами Диффи-Хеллмана считается самым надежным вариантом из доступных для обмена ключами.

В начале процессам обмена информацией, клиент и сервер обмениваются незашифрованными приветствующими сообщениями. Они обмениваются

приветствиями, затем они обмениваются информацией о том какой метод шифрования, обмена ключами и параметры сжатия каждый из них принимает и предпочитает. При выборе метода обмена ключами одним из вариантов является протокол обмена ключами Диффи-Хеллмана. Процесс обмена ключами использует ассиметричное шифрование для гарантии подлинности обоих сторон. Обычно это делается с помощью свежо вычисленной пары эфемерных ключей для каждой сессии. После этого обмена, вычисляются секреты и ключи, и стороны начинают шифровать трафик между ними, используя вычисленные ключи и согласованные методы [8].

При практическом использование, сервер проходит проверку подлинности пользователя, но не наоборот.

1.4 Применение протокола обмена ключами Диффи-Хеллмана в SSH

Безопасная оболочка (SSH) представляет собой как протокол, так и программу, использующихся для шифрования трафика между двумя компьютерами. Чаще всего используется для замены инструментов, таких как протоколы Telnet и Ftp, а также удаленные команды Беркли. Эти инструменты не шифруют свой трафик, в том числе процесс аутентификации, поэтому имена и пароли учетной записи передаются в незашифрованном виде.

Автором протокола SSH является Тату Илонен. Разработанный в 1995 году, протокол SSH был доступен для UNIX и других платформ. Существуют коммерческие и бесплатные версии реализаций. Рабочая группа SSH была спонсирована IETF, что позволило формализовать и стандартизировать протокол. Помимо обеспечения безопасной интерактивной оболочки, SSH также включает в себя другие функциональные возможности.

Протокол транспортного уровня SSH определён в документах IETF проекта «протокол транспортного уровня SSH». Этот уровень протокола использует TCP-порт 22, а также поддерживает высокоуровневые протоколы, такие как протокол подключения SSH, который обеспечивает удаленный интерактивный сеанс входа, удаленное выполнение команд и другие [13].

Две стороны в связи (например, клиент и сервер) начинают разговор с согласования параметров (например, предпочитаемые алгоритмы шифрования и сжатия). Затем вычисляется общий секрет при помощи протокола обмена ключами Диффи-Хеллмана, аналогично тому как было описано выше в протоколах SSL и TSL. Функция хеширования используется на общий секрет для получения ключа шифрования с целью согласования алгоритма симметричного шифрования. С этого момента обе стороны шифруют весь трафик между собой с помощью алгоритма симметричного шифрования.

1.5 Применение протокола обмена ключами Диффи-Хеллмана в IPSec

Протокол безопасности Интернета (IPSec) является протоколом, разработанным IETF для обеспечения защищенной связи на сетевом уровне IP. Ранее описанные протоколы SSL и SSH применяются в практических целях и обеспечивают только защиту передаваемого трафика, в то время как IPSec предназначен для защиты всего трафика передаваемого с помощью протокола IP, независимо от приложения.

Как и предыдущие протоколы, IPSec использует протокол обмена ключами Диффи-Хеллмана и ассиметричное шифрование для подтверждения личности, предпочитаемого алгоритма шифрования и общего секрета. Затем алгоритм использует симметричное шифрование для зашифровки больший объёмов данных, и они передаются через сеть.

Перед тем как протокол IPSec начинает шифровать поток данных, происходит предварительный обмен необходимой информацией. В этом помогает протокол обмена ключами в сети интернет (IKE). Протокол IKE работает в два этапа. Первый этап позволяет обеим сторонам динамически согласовывать параметры безопасности. Они используют протокол Диффи-Хеллмана, чтобы сгенерировать общий секрет и подтвердить подлинность друг друга. Во время второго этапа, общий секрет используется для шифрования передаваемой информации [12].

1.6 Применение протокола обмена ключами Диффи-Хеллмана в PKI

Инфраструктура открытых ключей (PKI) представляет собой систему протоколов и сервисов, поддерживающих шифрование с открытым ключом, которое является главным термином для ассиметричного шифрования. В шифрование с открытым ключом, математически согласованная пара ключей заменяют собой один ключ, который используется в симметричном шифрование. Первый из ключей используется для зашифровки, а второй ключ задействован в расшифровке зашифрованных данных. Один из ключей публикуется в открытом доступе, в другой храниться в строжайшей секретности.

Два дополняющие друг друга ключа используются в шифрование с открытым ключом. Если кто-то зашифрует сообщение открытым ключом другого человека, то только он может расшифровать это сообщение, потому что он знает свой секретный ключ. Это позволяет поддерживать конфиденциальность и широко применяется в шифрование с открытым ключом. С другой стороны, если кто-то зашифрует сообщение при помощи секретного ключа, то любой сможет расшифровать его с помощью открытого ключа и будет понятно, кто написал данное сообщение. Это называется цифровой подписью, которая гарантирует подлинность, неопровержимость и целостность передаваемых данных. Используя эти аспекты, шифрование с открытым ключом является очень надежной.

Значительной проблемой шифрования с открытым ключом является подтверждение правильности открытого ключа другого человека. То есть, до шифрования, существовала возможность того, что передаваемая информация могла быть перехвачена или изменена; с шифрованием, появилась вероятность перехвата и изменения открытого ключа.

Эта проблема решается с помощью иерархической системы Центра сертификации (CA). CA предоставляет цифровой сертификат, содержащий подтверждение подлинности человека и открытого ключа. Для гарантии подлинности открытого ключа, сертификат подписан секретным ключом CA. Для гарантии подлинности сертификата, он подписан высокоуровневой

цифровой подписью CA. Таким образом, обе стороны могут обмениваться зашифрованной информацией, получая открытый ключ и сертификат от PKI. Ключевым значением в этом процессе является то, что открытый ключ не изменяется при передаче, так что для получения ключа может быть использован протокол обмена ключами Диффи-Хеллмана.

В то время как авторы имеют разные определения PKI, это по существу является системой поддержки открытых ключей, используемых для шифрования с открытым ключом [20].

1.7 Использование дополнительных средств для увеличения надежности протокола обмена ключами Диффи-Хеллмана

Протокол обмена ключами Диффи-Хеллмана является одним из лучших протоколов для безопасного обмена ключами. Он имеет множество преимуществ, что позволяет ощущать чувство защищенности, используя его при шифровании. Однако протокол обмена ключами Диффи-Хеллмана уязвим для атак посредника. Из-за атак такого типа, исследователи пытались найти наилучшую защиту от них, в результате было найдено несколько вариантов решения данной проблемы. Самые известные решения данной проблемы являются: цифровая подпись (ЦП) и имитовставка (MAC). Несмотря на их удобства, они всё ещё имеют некоторые недостатки [18].

Цифровая подпись (ЦП): в основном, ЦП использует секретный ключ для подписи сообщения и открытый ключ для подтверждения сообщения на другой стороне. ЦП сталкивается с трудностями в обеспечении защиты секретных ключей, их верификации и конфиденциальности. С точки зрения защиты секретных ключей, если они подвергаются воздействию злоумышленника, то защита данных не имеет смысла.

Таким образом, большинство разработчиков и дизайнеров криптосистем считают, что пользователи должны хранить секретные ключи безопасными способами (например, никогда не отправлять или не хранить секретные ключи в виде обычного текста). Если мы потеряем их, то можем понести большие

убытки. Впоследствии, любой, кто получит секретный ключ, сможет подписывать и отправлять сообщения другой стороне (лицу, имеющему открытый ключ - Бобу), он распознает его как подлинное сообщение (владелец секретного ключа, подписавший сообщение - Алиса), следовательно, Боб будет думать, что сообщение было отправлено Алисой [7].

Каждый человек, который владеет секретным ключом, может подписать документ. Другими словами, ЦП не гарантирует автоматического подтверждения, что Алиса подписала документ. Это подтверждает то, что кто-то владеющий доступом к закрытому ключу подписал документ Алисы. Поэтому, как и печать, секретный ключ может быть украден и использоваться другими людьми. Что касается проверки достоверности и секретности, когда ЦП не прошла проверку подлинности у Боба, система помечает сообщение как недействительное, поскольку не может быть определено, было ли сообщение испорчено Евой или Алиса использовала неправильный секретный ключ. Это значит, что Алиса должна нести ответственность за безопасность своих секретных ключей. ЦП может обеспечить подлинность, но не может обеспечить безопасность секретного ключа.

Поэтому для шифрования и дешифрование требуется обеспечить дополнительную защиту. Без этого, ЦП не может защитить сообщения от перехвата и изменения атакой посредника.

Имитовставка (MAC): при использованиеMAC с обеих сторон (Алиса и Боб) один и тот же ключ для формирования ключе образную метку MAC и подтверждения подлинности.

Как правило, Алиса формирует метку MAC, используя алгоритм MAC и отправляет метку Бобу вместе с сообщением. Боб использует алгоритм MAC с тем-же самым ключом, чтобы получить в результате метку MAC. Затем Боб сравнивает получившуюся метку MAC с той, которую прислала Алиса в своем сообщение. Если они идентичны, то сообщение будет помечено как действительное, иначе оно будет помечено как недопустимое [14].

Таким образом, отправитель и получатель должны согласовать общий ключ и алгоритм MAC, прежде чем установить связь, так же, как и в симметричных криптосистемах. Из-за этого Боб не имеет доказательства, что сообщение пришло от Алисы (в отличии от цифровой подписи), которые означают, что MAC не обеспечивает неапеллируемость. Любой, кто получает сообщение и может подтвердить MAC, например, Боб, может формировать MAC для различных сообщений. Таким образом, MAC является симметричной схемой, которая не обеспечивает неапеллируемость.

В результате, Ева может атаковать каналы связи и перехватывать сообщения, которые будут отправлены Алисой Бобу, а потом Ева отправит их копию Бобу. Это убедит Боба что сообщение получено от Алисы. Ева так же может отвечать и Алисе, которая будет считать, что общается с Бобом.

Эти методы позволяют протоколу обмена ключами Диффи-Хеллмана повысить сопротивляемость к атакам посредника.

1.8 Уязвимость протокола обмена ключами Диффи-Хеллмана

Атаки на протокол Диффи-Хеллмана, подразделяются на несколько типов:

1) DoS-атака: При такой атаке, злоумышленник пытается нарушить успешную возможность проведения протокола между Алисой и Бобом. Злоумышленник может достигнуть своей цели несколькими способами, например, путем удаления сообщений которые отправляют друг другу Алиса и Боб, или перегрузка сторон ненужными вычислениями или связями [17].

2) Атака снаружи: Злоумышленник пытается нарушить протокол (например, путем добавления, удаления, повторения сообщений), так что он получает важную информацию (которую он не мог бы получить, зная только общедоступные данные).

3) Атака изнутри: Вполне возможно, что один из участников протокола обмена ключами Диффи-Хеллмана намеренно создает ненадежную сессию, для того чтобы получить данные о секретном ключе собеседника. Эта атака очень эффективна если собеседник использует статический секретный ключ, который

задействован во многих сеансах протокола обмена ключами. Вредоносное программное обеспечение может быть очень эффективно при использование этой атаки [9].

Достоверность этих атак зависит от того, какие предположения мы делаем о противнике. Например, если противник может удалить и заменить любое сообщение из канала связи общего пользования, DoS-атаку невозможно предотвратить. К счастью, атаки снаружи (в которых атакующий получает общий секретный ключ) и атаки изнутри могут быть предотвращены во многих ситуациях.

1.8.1 Атака посредника

Активный противник, внедренный в канал связи между Алисой и Бобом, может манипулировать сообщениями протокола и начать атаку посредника**.**

Это атака, при которой злоумышленник, хакер или взломщик (назовем его секретным передатчиком Ева) имеет возможность изменить проводимый обмен данными между отправителем Алисой и получателем Бобом так, чтобы они считали, что общаются непосредственно друг с другом. Атака посредника является типом кибератак, при которой злоумышленник входит в контакт с отправителем и получателем, подражает обеим сторонам и получает доступ к информации, которую обе стороны пытались послать друг другу. Протоколы SSL (secure sockets layer – уровень защищенных сокетов) и TSL (transport layer security – безопасность транспортного уровня) устраняют эту проблему, но последние исследования показали, что атаки возможны даже при использовании этих протоколов безопасности. Таким образом, существуют строгие условия разработки механизма обмена ключами, который имеет высокую сопротивляемость атакам посредника благодаря встроенным математическим функциям.

Атака посредника на протокол обмена ключами Диффи-Хеллмана:

Общие исходные данные:

(p, g) : p - большое простое число, g - порождающий элемент группы F_p^*.

Результат:

1. Алиса генерирует элемент $a \in U[1, p-1)$, вычисляет значение $g_a \leftarrow g^a(mod\ p)$ и отправляет его Злоумышленнику (“Бобу”).
2. Злоумышленник (“Боб”) вычисляет значение $g_m \leftarrow g^m(mod\ p)$, где $m \in [1, p\text{-}1)$ и отправляет его Бобу.
3. Боб генерирует элемент $b \in U\ [1, p-1)$, вычисляет значение $g_b \leftarrow g^b(mod\ p)$.
4. Отправляет значение g_b Злоумышленнику (“Алисе”).
5. Злоумышленник (“Алиса”) отправляет Алисе число g_m.
6. Алиса вычисляет значение $k_1 \leftarrow g_m^a(mod\ p)$. (Этот ключ распределен между Алисой и Злоумышленником, поскольку Злоумышленник может вычислить значение $k_1 \leftarrow g_a^m(mod\ p)$.).
7. Боб вычисляет значение $k_2 \leftarrow g_m^b(mod\ p)$. (Этот ключ распределен между Бобом и Злоумышленником, поскольку Злоумышленник может вычислить значение $k_2 \leftarrow g_b^m(mod\ p)$.).

В ходе этой атаки Злоумышленник перехватывает и блокирует первое сообщение Алисы Бобу, т.е. число g_a, маскируется под Алису и посылает Бобу следующее сообщение.

Злоумышленник (“Алиса”) - Бобу: $g_m(\stackrel{\text{def}}{=} g^m)(mod\ p)$.

Боб, следуя инструкциям протокола, возвращает Злоумышленнику (“Алисе”) число g_b. Это число снова перехватывается и блокируется Злоумышленником. Теперь Злоумышленник и Боб согласовали между собой ключ $g^{bm}(mod\ p)$, хотя Боб считает, что он разделил этот ключ с Алисой.

Аналогично Злоумышленник, имитируя Боба, может согласовать другой ключ с Алисой (т.е. число $g^{am}(mod\ p)$). Впоследствии Злоумышленник может использовать оба ключа для чтения и подмены “секретных” сообщений, которыми обмениваются Алиса и Боб, или поочередно имитировать этих пользователей [15].

Атака на протокол обмена ключами Диффи-Хеллмана вполне реальна, поскольку этот протокол не предусматривает проверки аутентичности источника

сообщений. Для того чтобы ключи были согласованы только между Алисой и Бобом, обе стороны должны быть уверены друг в друге.

1.8.2 Атаки, основанные на теории чисел.

Вышеописанная атака посредника, хоть она и рушит защиту протокола обмена ключами Диффи-Хеллмана, но требует задействования Евы(злоумышленника) для работы в полную силу. Например, если секретные ключи используются в сочетание с MAC, Ева должна перехватить и изменить все подтверждающие подлинность сообщения, чтобы предотвратить обнаружение фальсификации ключей Алисой и Бобом. В некоторых следующих подразделах, Алиса и Боб имеют одинаковые секретные ключи (которые знает Ева). Таким образом, Еве нужно быть задействованной только во время использования протокола обмена ключами Диффи-Хеллмана, после этого она может разрушить защиту протокола, используя общий ключ, в любой момент [3].

Атака вырожденными сообщениями

Существуют вырожденные случаи, в которых протокол не работает. Например, когда g^x и g^y равны единице, тогда и секретный ключ становиться единицей. Так как канал связи является общедоступным, любой может обнаружить эту аномалию. К счастью, эта ситуация невозможна в правильной работе протокола, поскольку оба элемента x и y выбираются из множества $\{1, \dots, p-2\}$.

Тем не менее, возможно проведение атаки изнутри и участники протокола обмена ключами Диффи-Хеллмана должны убедиться, что общий секретный ключ не должен выдавать значение $g^z = 1$.

Простые показатели

Если один из элементов x и y, могут быть легко вычислены, то работа протокола может быть сломана. Например, если x равен единице, то $g^x = g$, значение которого может узнать любой наблюдающий злоумышленник. Очень трудно определить, где провести грань, позволяющую определять такие значения g^i, чтобы значение i было трудно вычислить, так как это полностью зависит от стратегии злоумышленника. Любые наборы значений i могут быть

уязвим, в зависимости от предварительно вычисленного значения g^i. В любом случае, добиться того, чтобы элементы x и y не были равны единице, кажется весьма обоснованным действием.

Атака простого замещения

Злоумышленник может изменить значение секретного ключа на «невозможное». Если протокол Диффи-Хеллмана будет выполняться людьми, то эта аномалия может быть легко выявлена. Однако на практике протокол Диффи-Хеллмана выполняется с помощью компьютеров и недобросовестно сделанные реализации могут не выявить следующую атаку.

1. Злоумышленник перехватывает значения g^x и g^y и заменяет их единицей.
2. Алиса и Боб вычисляют один и тот же общий секретный ключ, который равняется единице.

Если компьютерная программа не понимает, что значения g^x, g^y и g^{xy} не должны равняться единице, то протокол уязвим. Поэтому, всегда стоит проверять, что значения g^x и g^y являются положительными целыми числами меньше, чем $p-1$ и больше единицы.

Атака на основе составного порядка подгрупп

Злоумышленник может воспользоваться подгруппами, которые не имеют больших простых порядков. Это лучше всего показать на примере. Предположим, что Алиса и Боб выбрали простое число $p = 2q + 1$, где q – простое число и сгенерированный элемент g порядка $p - 1 = 2q$. Ева может перехватить сообщения с элементами g^x и g^y и возвести их в степень q. Она заменит элементы g^x и g^yна g^{xq} и g^{yq} соответственно. В результате вычислений секретный ключ станет g^{xyq}, что позволит Еве найти это значение перебором. Это происходит благодаря тому, что порядок элемента $g^q = g^{\frac{p-1}{2}}$ равен 2, куда вытекает, что секретный ключ может принимать только одно из двух значений. Следовательно, Ева может использовать перебор для того, чтобы определить

значение секретного ключа; например, когда Алиса и Боб используют его для симметричного шифрования.

В более общем плане, эта атака может быть легко использована на простых числах вида $p = Rq + 1$ (R маленькое число), единственное различие состоит в том, что существует возможность найти значение R при помощи перебора.

Урок, который можно извлечь из этой атаки является то, что мы должны выбирать элемент g, который будет порождать подгруппу большого простого порядка, или по крайней мере убедиться, что составной порядок подгруппы не является уязвимым [12].

Обратим внимание на то, что атака изнутри может быть использована с помощью этого приема. Алиса просто выбирает значение x равное q. В данном случае, даже механизм подтверждения подлинности не защитит Боба.

1.8.3 Проверка подлинности

В данном пункте мы рассмотрим вопросы, связанные с проверки подлинности. Поскольку протокол обмена ключами Диффи-Хеллмана уязвим для атак, то не имеет смысла говорить о безопасности протокола обмена ключами Диффи-Хеллмана без обсуждения проверки подлинности.

Проверка подлинности заключается в установление достоверности, которая может быть описана фактической точностью и надежностью. Не существует формального определения, потому что различные параметры и требования меняются для каждого применения.

Например, проверка подлинности цифровой подписи или метки MAC является простой, так как требуется применить функцию верификации. Тогда как доказать подлинность сообщения является более сложной задачей. Например, если Алиса посылает сообщение Бобу, то он хотел бы установить, что:

1. Сообщение не было изменено.
2. Отправителем сообщения является Алиса.
3. Сообщение было предназначено ему.
4. Сообщение не было отправлено повторно.

5. Сообщение было отправлено без большой задержки по времени.

Хотя и существуют методы, которые могут помочь в создание механизма подтверждения подлинности (цифровые подписи, метки MAC), правильно их использование достаточно трудно.

1.8.4 Атака на параметры проверки подлинности

Как правило, все параметры, используемые в криптографическом протоколе должны пройти проверку подлинности. Например, предположим, что протокол обмена ключами Диффи-Хеллмана использует различные системные параметры (например, g и p). Если участники протокола не подтвердили подлинность своих выбранных параметров, то злоумышленник может обмануть их используя ложные параметры.

Эта атака является тяжело обнаружимой и может быть не замечена даже лучшими криптографами и экспертами по вопросам безопасности. Достаточно просто взглянуть на атаки, направленные на протокол SSL, чтобы убедиться в этом.

1.9 Протокол обмена ключами Диффи-Хеллмана с тремя и более участниками

Использование протокола обмена ключами Диффи-Хеллмана не ограничивается двумя участниками. Он может быть применен на неограниченное количество пользователей.

Рассмотрим ситуацию, когда Алиса, Боб и Кэрол вместе генерируют исходный ключ. В данном случае последовательность действий, будет следующая:

1. Стороны договариваются о параметрах алгоритма p и q.
2. Стороны генерируют свои ключи - a, b, c.
3. Алиса вычисляет $A = g^a$ и посылает его Бобу.
4. Боб вычисляет $A = (g^a)^b = g^{ab}$ и посылает его Кэрол.
5. Боб вычисляет $A = g^b$ и посылает его Кэрол.

6. Кэрол вычисляет $A = (g^b)^c = g^{bc}$ и посылает его Алисе.

7. Алиса вычисляет $A = (g^{bc})^a = g^{bca} = g^{abc}$ и использует его как свою тайну.

8. Кэрол вычисляет $A = g^c$ и посылает его Алисе.

9. Алиса вычисляет $A = (g^c)^a = g^{ca}$ и посылает его Бобу.

10. Боб вычисляет $A = (g^{ca})^b = g^{cab} = g^{abc}$ и использует его как свою тайну.

В данной ситуации любой участник может видеть $A = g^a$, $A = g^b$, $A = g^c$, $A = g^{ab}$, $A = g^{ac}$, $A = g^{bc}$, но при этом не может видеть любую комбинацию $A = g^{abc}$.

Для того чтобы данный алгоритм был эффективно применен для большой группы людей, необходимо соблюдение двух основных принципов:

- передача ключа должна начинаться с «пустого» ключа g. Весь секрет состоит в повышении текущего значения показателя каждого участника один раз;
- любое промежуточное значение может быть раскрыто публично, но окончательное значение представляет из себя секретный ключ, который никогда не должен быть публично раскрыт. Таким образом, каждый пользователь получает свою копию тайного ключа и передает его последующему [10].

1.10 Задача Диффи-Хеллмана и задача дискретного логарифмирования

Для протокола обмена ключами Диффи-Хеллмана обоснованным действием в криптографическом контексте является невозможность злоумышленника перехватить секретную информацию из данных, проходящих через открытый канал связи. При использовании классического протокола обмена ключами Диффи-Хеллмана, возникает проблема с вычислением значения K из уравнения

$$g^k \equiv c \; mod \; p. \tag{1.8}$$

Это приводит к задаче дискретного логарифмирования [11]. Существует бесконечное множество решений, но, как правило, мы выбираем наименьшее положительное значение.

Определение (Задача дискретного логарифмирования): пусть p – простое число и g сгенерировано в Z_p^*. Принимая во внимание элемент

$$c \in Z_p^*, \tag{1.9}$$

найдем такое простое число k, чтобы

$$0 \leq k < p - 1 \text{ и } g^k \equiv c \; mod \; p. \tag{1.10}$$

Решение задачи дискретного логарифмирования дает нам дискретный логарифм.

Определение: пусть p – простое число и g сгенерировано в Z_p^*. Дискретный логарифм от

$$c \in Z_p^*, \tag{1.11}$$

по основанию g, обозначаемый $\log_g c$, является единственным целым числом k таким, что

$$0 \leq k < p - 1 \text{ и } g^k \equiv c \; mod \; p. \tag{1.12}$$

Как

$$g^k \equiv \; g^{(k+l*ord(g))} \tag{1.13}$$

для любого $k \in Z$, дискретный логарифм не является уникальным, если мы не установим требуем ,чтобы

$$0 \leq k < p - 1. \tag{1.14}$$

Аналогичные свойства обычных логарифмов справедливы и для дискретных логарифмов по модулю $ord(g) = p - 1$, то есть

$$\log_g xy = \; \log_g x + \log_g y \; mod \; ord(g) \tag{1.15}$$

и

$$\log_g x^y = \; y \log_g x \; mod \; ord(g). \tag{1.16}$$

Обратите внимание, что существуют так же множество сопутствующих проблем. Например, вычислительная задача Диффи-Хеллмана, которая требует вычислить значение

$$g^{ab} mod\ p \tag{1.17}$$

при известных $g^{a} mod\ p$ и $g^{b} mod\ p$. Очевидно, что решение задачи дискретного логарифмирования позволило бы решить задачу Диффи-Хеллмана, но неизвестно, справедливо ли обратное утверждение [19].

Глава 2 Реализация и тестирование протокола

2.1 Разработка реализации протокола обмена ключами Диффи-Хеллмана на языке программирования C++

Исходя из требований к программному продукту, алгоритм подсчета чисел, являющимися ключами, в виде функции. На входе функция получает три числа, а на выходе отдает одно число – ключ. На рисунке 2.1 представлен код функции, которая вычисляет ключ:

```
long int power(long long int a, long long int b, long long int mod)
{
    long long int t;
    if (b == 1)
        return a;
    t = power(a, b / 2, mod);
    if (b % 2 == 0)
        return (t*t) % mod;
    else
        return (((t*t) % mod)*a) % mod;
}
```

Рисунок 2.1 — Код функции вычисления ключа на языке программирования C++

На рисунке 2.2 показана функция, отвечающая за выполнение протокола обмена ключами Диффи-Хеллмана. Здесь мы вычисляем секретные ключи для обоих сторон на основе сгенерированных элементов x, y, p и g.

```
void prot()
{
    a = power(g, x, p);
    b = power(g, y, p);
    cout << "key for the first person is : " << power(b, x, p) << endl;
    cout << "key for the second person is :" << power(a, y, p) << endl;
}
```

Рисунок 2.2 — Код функции отвечающей за выполнение протокола обмена ключами Диффи-Хеллмана на языке программирования C++

На рисунке 2.3 представлена функция подсчета времени. В этой функции происходит расчет времени до начала выполнения протокола обмена ключами Диффи-Хеллмана и после выполнения. Так же происходит вызов функции

отвечающей за выполнение протокола обмена ключами Диффи-Хеллмана, которая показана на рисунке 2.2.

```
double time()
{
     start = GetTickCount();
     prot();
     finish = GetTickCount();
     return(finish - start);
}
```

Рисунок 2.3 — Код функции подсчета времени работы протокола на языке программирования C++

Блок-схема функции, вычисляющей ключи для протокола обмена ключами Диффи-Хеллмана, изображена на рисунке 2.4. На данной блок схеме подробно и пошагово рассмотрен алгоритм функции представленный на рисунке 2.1. Нам на вход податься три переменные a, b и mod. Эти переменные используются для вычисления секретного ключа протоколом Диффи-Хеллмана по следующей формуле:

$$g^z \ mod \ p, \tag{2.1}$$

где значения g, z и p соответственно равны элементам a, b и mod.

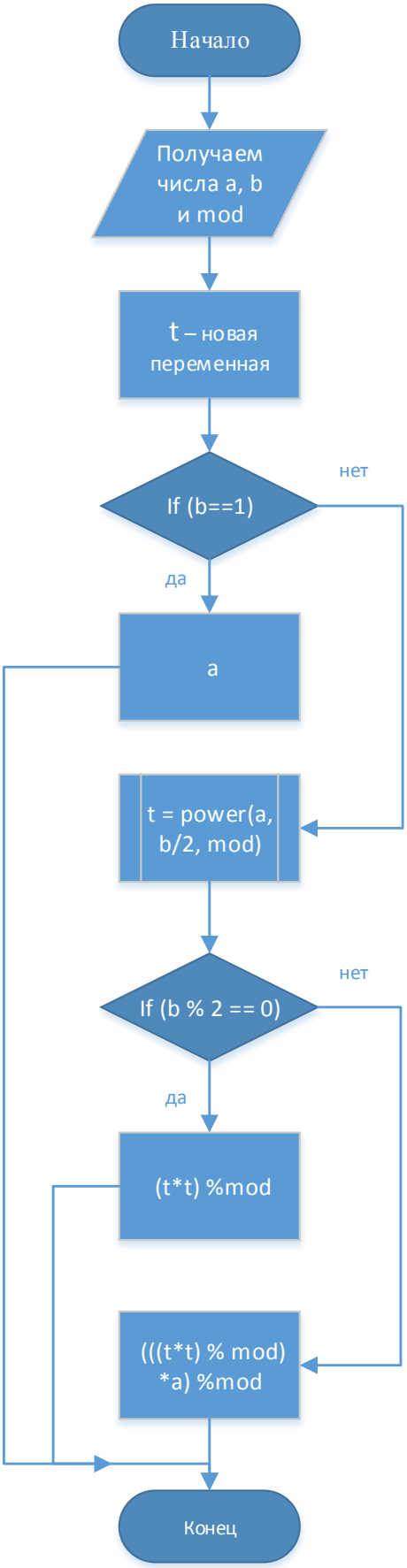

Рисунок 2.4 — Блок-схема функции вычисления секретных ключей

2.2 Разработка реализации протокола обмена ключами Диффи-Хеллмана на языке программирования Java

Реализация на языке Java значительно отличается от реализации на языке C++. В данной реализации используется встроенная функция modPow, что позволяет вычислять ключи гораздо быстрее, чем в реализации на языке C++. Использование данной функции, так же уменьшает объём кода, что тоже благоприятно влияет на быстродействие программы.

Для наименьшего различия в результатах тестирования, было решено соблюдать наибольшую идентичность структуры программ.

```
public static void prot()
    {
        R1=g.modPow(x,p);
        R2=g.modPow(y,p);
        K1=R2.modPow(x,p);
        K2=R1.modPow(y,p);
        System.out.println("First person key is:"+K1);
        System.out.println("Second person key is:"+K2);
    }
```

Рисунок 2.5 — Код функции отвечающей за выполнение протокола обмена ключами Диффи-Хеллмана на языке программирования Java

На рисунке 2.5 представлена функция, отвечающая за вычисления секретных ключей для обоих сторон, участвующих в протоколе обмена ключами Диффи-Хеллмана. Данная функция написана максимально похоже на реализацию функции на языке программирования C++.

Далее на рисунке 2.6 изображена функция подсчета времени.

```
public static long mytim()
    {
        before = System.currentTimeMillis();
        prot();
        after = System.currentTimeMillis();
        diff = after - before;
        return diff;
    }
```

Рисунок 2.6 — Функция подсчета времени на языке программирования Java

На этом рисунке, по аналогии с функцией на языке C++, происходит расчет времени до выполнения протокола обмена ключами Диффи-Хеллмана, вызов протокола на исполнение и расчет времени после завершения исполнения протокола.

2.3 Тестирование протокола

В данных реализациях будут использоваться большое число p, сгенерированное число g, порождающее группу F_p^*, причем

$$2 \leq g \leq p - 2. \quad (2.2)$$

Далее каждая из двух сторон выбирает свой секретный ключ x и y соответственно при

$$1 \leq x \leq p - 2 \text{ и } 1 \leq y \leq p - 2. \quad (2.3)$$

Для тестирования работы протокола обмена ключами будут использоваться небольшие и заранее сгенерированные числа. В ходе выполнения, программа должна высчитать общий секретный ключ.

Первым протоколом, участвующем в тестирование будет протокол, реализованный на языке программирования C++.

```
Enter the value of p:
23
Enter the value of g:
5
Enter the value of x for the first person :
6
Enter the value of y for the second person :
15
key for the first person is : 2
key for the second person is :2
```

Рисунок 2.7 — Результат работы программы на языке программирования C++

Результатом тестирования протокола, изображенного на рисунке 2.7, стали два одинаковых секретных ключа, полученных у каждой стороны.

Следующим протоколом, проходящим тестирование, является протокол, реализованный на языке программирования Java. Результат тестирования изображен на рисунке 2.8.

```
Enter prime number:
23
Enter primitive root of 23:
5
Enter value for x less than 23:
6
R1=8
Enter value for y less than 23:
15
R2=19
Key calculated at Alice's side:2
Key calculated at Bob's side:2
deffie hellman secret key Encryption has Taken
```

Рисунок 2.8 — Результат работы программы на языке программирования Java

После результатов тестирования, мы с уверенностью может сказать, что обе реализации протокола обмена ключами работают правильно.

Глава 3 Сравнительный анализ реализаций

3.1 Подсчет времени вычисления ключей для протокола обмена ключами Диффи-Хеллмана

За подсчет и вывод времени выполнения вычислений в реализации протокола обмена ключами Диффи-Хеллмана на языке программирования C++, отвечает следующая часть кода программы:

```
double start;
double finish;
start = GetTickCount();
a = power(g, x, p);
b = power(g, y, p);
cout << "key for the first person is : " << power(b, x, p) << endl;
cout << "key for the second person is :" << power(a, y, p) << endl;
finish = GetTickCount();
out << finish - start << "\n";
```

Рисунок 3.1 — Код подсчета времени выполнения вычислений на C++

В данном участке кода, изображенном на рисунке 3.1, мы объявляем переменные start и finish типа double, и присваиваем им значения полученные функцией GetTickCount() которые показывают число миллисекунд, которые истекли с тех пор как система была запущена. Переменная start будет вызываться до начала вычислений, а переменная finish по окончании. На выходе выводиться время вычислений в миллисекундах.

На языке Java так же происходит подсчет времени выполнения вычислений. Далее представлен код программы отвечающий за них:

```
long before;
long after;
long diff;
before = System.currentTimeMillis();
R1=g.modPow(x,p);
R2=g.modPow(y,p);
K1=R2.modPow(x,p);
K2=R1.modPow(y,p);
System.out.println("First person key is "+K1);
System.out.println("First person key is "+K2);
after = System.currentTimeMillis();
diff = after - before;
out.println(diff);
```

Рисунок 3.2 — Код подсчета времени выполнения вычислений на Java

В данном участке кода, изображенном на рисунке 3.2, мы объявляем переменные before, after и diff типа long. Переменной before присваивается время перед началом выполнения вычислений при помощи функции System.currentTimeMillis(). Переменной after присваивается время после вычислений. На выходе подсчитывается разница во времени между переменными after и before и записывается в переменную diff. Затем выводиться время вычислений в миллисекундах.

3.2 Практическое сравнение реализаций

При разработке реализаций протокола обмена ключами Диффи-Хеллмана были предусмотрены функции, отвечающие за подсчет времени работы протокола в миллисекундах.

Время замерялось до начала и после проведения вычисления секретного ключа. Среднее время будем рассчитывать за 100 проходов. Это позволит максимально снизить уровень влияния аппаратуры, на время сортировки.

Рисунок 3.3 — Время вычисления ключей в обоих реализациях

На рисунке 3.3 показан график сравнения времени вычисления секретных ключей при помощи протокола обмена ключами Диффи-Хеллмана. На оси абсцисс у нас расположено количество итераций, а на оси ординат время (в миллисекундах) вычисления секретного ключа.

Как мы видим, время вычисления ключа на языке Java значительно быстрее, чем на C++. В нашем случае время вычисления секретного ключа

зависит от языка программирования. Поскольку на языке программирования Java использовались встроенные функции, недоступные на языке программирования C++, что и оказало большое влияние на скорость работы программ. Погрешность в графиках обуславливается аппаратными характеристиками компьютера, на котором проходило тестирование.

Заключение

В ходе выполнения бакалаврской работы был рассмотрен протокол обмена ключами Диффи-Хеллмана, его теоретические основы, атаки, проводимые на протокол и методы для противостояния им, достоинства и недостатки протокола, а также были написаны две реализации на языках программирования Java и C++.

Протокол обмена ключами Диффи-Хеллмана является достаточно надежным методом обмена ключами через незащищенные каналы передачи информации. Он используется в системах, обеспечивающих безопасность передаваемых данных и протоколах, использующихся сегодня. За долгие годы использования протокола обмена ключами Диффи-Хеллмана было разработано множество атак, но и сам протокол не стоял на месте и множество исследователей изменяли и модифицировали его. Это доказывает, что безопасный обмен секретными ключами является важным аспектом защиты информации. Что делает протокол обмена ключами Диффи-Хеллмана востребованным и в наше время.

В работе были рассмотрены теоретические основы протокола обмена ключами Диффи-Хеллмана. Был пошагово описан ход выполнения протокола.

Также были рассмотрены системы обеспечения безопасности данных, в которых протокол обмена ключами Диффи-Хеллмана широко используется для безопасного обмена секретными ключами.

Были рассмотрены дополнительные средства, увеличивающие надежность протокола обмена ключами Диффи-Хеллмана, помогающие в решение проблемы – атаки посредника. Одним из рассмотренных средств является цифровая подпись, позволяющая помечать сообщения, тем самым подтверждая их подлинность.

Также были рассмотрены и сами атаки на протокол обмена ключами Диффи-Хеллмана. Главное внимание было уделено атаке посредника, как наиболее опасной из рассмотренных.

Помимо атаки посредника, были рассмотрены атаки, основанные на теории чисел. Эти атаки направлены на изменение параметров, которые

протокол обмена ключами Диффи-Хеллмана использует для создания секретного ключа. Однако не только атаки злоумышленников угрожают безопасности данных, но и неправильное использование протокола, плохой выбор параметров и недостатки реализации.

Была рассмотрена возможность использования протокола обмена ключами Диффи-Хеллмана более чем двумя участниками. Это позволяет неограниченному числу пользователей безопасно обмениваться секретными ключами с целью дальнейшего обмена данными.

Помимо этого, были рассмотрены задача Диффи-Хеллмана и задача дискретного логарифмирования. Криптографическая стойкость протокола обмена ключами Диффи-Хеллмана зависит от сложности проблемы дискретного логарифмирования.

В дополнение к рассмотрению теоретических аспектов, был реализован протокол обмена ключами Диффи-Хеллмана на двух языках программирования, Java и C++. Реализации были написаны максимально приближенно друг к другу. Проведена проверка реализаций на работоспособность. Обе реализации показали идентичные результаты, при использование одинаковых наборов входных данных.

Для сравнительного анализа реализаций протокола обмена ключами Диффи-Хеллмана. были добавлены функции подсчета времени, позволяющие выявить быстродействие обеих реализация. Для получения более точных результатов было проведено не менее 100 итераций вычисления общего секретного ключа протоколом Диффи-Хеллмана.

Практическое сравнение реализаций показало, что быстродействие реализации, написанной на языке программирования Java, превосходит аналогичную реализация на языке программирования C++. Такие результаты достигаются за счет встроенных методов Java, которые отсутствуют в C++.

Данные выводы показывают, что использование языка программирования Java выгоднее использовать при создании систем, обеспечивающих

безопасность передачи данных, которые задействуют протокол обмена ключами Диффи-Хеллмана.

Список используемой литературы

1. Касто, В. Просто криптография / В. Касто. — СПб.: ООО «Страта», 2014 — 208 с.

2. Герман, О.Н. Теоретико-числовые методы в криптографии: учебник / О.Н. Герман, Ю.В. Нестеренко. — М.: Академия, 2012. — 272 с.

3. Ишмухаметов, Ш.Т. Математические основы защиты информации: учеб. пособие / Ш.Т. Ишмухаметов, Р.Г. Рубцов. — Казань: Казанский федер. Ун-т, 2012. — 138 с.

4. Кнауб, Л.В. Теоретико-численные методы в криптографии: учеб. пособие / Е.А. Новиков, Ю.А. Шитов, Л.В. Кнауб. — Красноярск: Сиб. федер. ун-т, 2011. — 161 с.

5. Нестеренко, А.Ю. Теоретико-числовые методы в криптографии: учеб. пособие / А.Ю. Нестеренко. — Моск. гос. ин-т. электроники и математики, 2012 — 224 с.

6. Орлов, В.А. Теория чисел в криптографии: учеб. пособие / В.А. Орлов, Н.В. Медведев, Н.А. Шимко, А.Б. Домрачева. — М.: Изд-во МГТУ им. Н.Э. Баумана, 2011 — 223 с.

7. Романьков, В.А. Введение в криптографию. Курс лекций / В.А. Романьков. — М.: ФОРУМ, 2012. — 240 с.

8. Рябко, Б.Я. Основы современной криптографии и стеганографии: [монография] / А.Н. Фионов, Б.Я. Рябко. — 2-е изд. — М.: Горячая линия – Телеком, 2013. —Библиогр.: с. 225-229 (55 назв.)

9. Салий, В.Н. Криптографические методы и средства защиты информации: учеб. пособие / В. Н. Салий. — Саратов: Саратовский гос. ун-т имени Н.Г. Чернышевского, 2012. — 41 с.

10. Сердюк, А.И. Криптография. Разработка приложений для шифрования информации: метод. указания / О.Н. Яркова, А.И. Сердюк. — Оренбург: ГОУ ОГУ, 2012 — 98 с.

11. Ященко, В.В. Введение в криптографию: книга / В.В. Ященко. — 4-е изд., доп. М.: МЦНМО, 2012. — 348 с.

12. Adrian, D. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. 2014. [Электронный ресурс]. Режим доступа: https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf

13. Boni, S. Improving the Diffie-Hellman Key Exchange Algorithm by Proposing the Multiplicative Key Exchange Algorithm. / S. Boni, J. Bhatt, S. Bhat. 2015. [Электронный ресурс]. Режим доступа: http://www.ijcaonline.org/research/volume130/number15/boni-2015-ijca-907170.pdf

14. Bowman, J.C. Coding theory & cryptography. 2015. [Электронный ресурс]. Режим доступа: https://www.math.ualberta.ca/~bowman/m422/m422.pdf

15. Chaturvedi, A. A Secure Wireless Communication protocol using Diffie-Hellman Key Exchange. / A. Chaturvedi, N. Srivastava, V. Shukla. 2015. [Электронный ресурс]. Режим доступа: http://www.ijcaonline.org/research/volume126/number5/chaturvedi-2015-ijca-906060.pdf

16. Cozzens, M.J. The Mathematics of Encryption: An Elementary Introduction. 2013. [Электронный ресурс]. Режим доступа: https://books.google.ru/books?id=GbKyAAAAQBAJ&printsec=frontcover&dq=The+Mathematics+of+Encryption:+An+Elementary+Introduction&hl=ru&sa=X&ved=0ahUKEwif8-7t#v=onepage&q=The%20Mathematics%20of%20Encryption%3A%20An%20Elementary%20Introduction&f=false

17. Jenings, T.A. The mathematics of cryptography & data compression. 2012. [Электронный ресурс]. Режим доступа: https://www.carroll.edu/library/thesisArchive/Anderson%20J_2012final.pdf

18. Lai, D. Preventing Man-In-The-Middle Attack in Diffie-Hellman Key Exchange Protocol. / D. Lai, A. Khader.2015. [Электронный ресурс]. Режим доступа: https://www.researchgate.net/publication/280722113_Preventing_Man-In-The-Middle_Attack_in_Diffie-Hellman_Key_Exchange_Protocol

19. Lehtinen, S. Diffie-Hellman Key Exchange – From Mathematics to Real Life. 2011. [Электронный ресурс]. Режим доступа: https://tampub.uta.fi/bitstream/handle/10024/83062/gradu05484.pdf?sequence=1

20. Ruohonen, K. Mathematical cryptology. 2014. [Электронный ресурс]. Режим доступа: http://math.tut.fi/~ruohonen/MC.pdf

Приложение А

Листинг кода, реализаций протокола Диффи-Хеллмана.

Реализация на языке Java:

```
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import static java.lang.Math.random;
import java.util.*;
import java.util.Random;
import static java.lang.StrictMath.random;
import java.math.BigInteger;
import static jdk.nashorn.internal.objects.NativeMath.random;

public class MainClass
{
   public static int p1, g1, x1, y1;
   public static BigInteger p, g, x, y, R1, R2, K1, K2;
   public static BigInteger z = new BigInteger("2");
   public static long before;
   public static long after;
   public static long diff;
   public static long diffc;
   public static  Random random = new Random();
   public static void main(String[]args)throws IOException
   {
      try (PrintWriter out = new PrintWriter(new OutputStreamWriter(new
FileOutputStream("C:/fsd/asj.txt")), true)) {

         for(int i=0;i<100; i++)
         {
            mytim();
         }
          out.close();
      }
    }
//Функция, вычисляющая секретный ключ
   public static void prot()
   {
       p1 = random.nextInt(9999)+2;

       p = BigInteger.probablePrime(1024, random);
       do
       {
          g=BigInteger.probablePrime(1024, random);
```

```
        }
        while(g.compareTo(p.subtract(z))<=0 && g.compareTo(z)>=0);
        do
        {
           x=BigInteger.probablePrime(1024, random);
        }
        while(x.compareTo(p.subtract(z))<=0 && x.compareTo(BigInteger.ONE)>=0);
        do
        {
           y=BigInteger.probablePrime(1024, random);
        }
        while(y.compareTo(p.subtract(z))<=0 && y.compareTo(BigInteger.ONE)>=0);
        R1=g.modPow(x,p);
        R2=g.modPow(y,p);
        K1=R2.modPow(x,p);
        K2=R1.modPow(y,p);
        System.out.println(K1+" "+K2);
    }
//Функция подсчета времени выполнения
    public static long mytim()
    {
        before = System.currentTimeMillis();
        prot();
        after = System.currentTimeMillis();
        diff = after - before;
        return diff;
    }
    }
```

Реализация на языке C++:

```
#include<stdio.h>
#include<iostream>
#include <fstream>
#include<windows.h>
#include <iostream>
#include <ctime>
#include <cstdlib>
using namespace std;
long long int p, g, x, a, y, b;
double start;
double finish;
double result;

//Выбирает все простые числа до заданного.
long long int sundaram(long long int n)
{
        long long int *a = new long long int[n], i, j, k;
        memset(a, 0, sizeof(long long int) * n);
```

```
	for (i = 1; 3 * i + 1 < n; i++)
	{
		for (j = 1; (k = i + j + 2 * i*j) < n && j <= i; j++)
			a[k] = 1;
	}
	//Выбирает из списка простых чисел ближайшее к заданному.
	for (i = n - 1; i >= 1; i--)
		if (a[i] == 0)
		{
			return (2 * i + 1);
			break;
		}
	delete[] a;
}
//Функция возведения числа a в степень b по модулю mod
long int power(long long int a, long long int b, long long int mod)
{
	long long int t;
	if (b == 1)
		return a;
	t = power(a, b / 2, mod);
	if (b % 2 == 0)
		return (t*t) % mod;
	else
		return (((t*t) % mod)*a) % mod;
}
//Функция вычисления секретного ключа

void prot()
{
	p = rand() % 10000;
	p = sundaram(p);
	do
	{
		g = rand() % (p-2);
	} while (g>=p-2 && g<=2);
	do
	{
		x = rand() % (p-2);
	} while (x >= p - 2 && x <= 1);
	do
	{
		y = rand() % (p-2);
	} while (x >= p - 2 && x <= 1);
	a = power(g, x, p);
	b = power(g, y, p);
	cout << "key for the first person is : " << power(b, x, p) << endl;
	cout << "key for the second person is :" << power(a, y, p) << endl;
}
```

```
//Функция подсчета времени выполнения

double time()
{
	start = GetTickCount();
	prot();
	finish = GetTickCount();
	return(finish - start);
}
void main()
{
	srand((unsigned)time(NULL));
	std::ofstream out("C:/fsd/acs.txt", std::ios::app);
	for (int i = 0; i < 100; i++)
	{
		time();
	}
	out.close();
}
```