САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Дьяченко Александр Сергеевич

Выпускная квалификационная работа бакалавра

Комбинированный алгоритм гомоморфного шифрования

Направление 010300

Фундаментальные информатика и информационные технологии

Научный руководитель,
кандидат физ.-мат. наук,
доцент
Мисенов Б. А.

Санкт-Петербург

2016

Оглавление

Введение

Проблема защиты информации, исключающего возможности прочтения ее посторонним лицом, волновала человеческий ум еще с давних времен. Так, еще на заре человеческой цивилизации утечка информации об убитом животном, могла оставить без пищи целое первобытное племя. Таким образом, с зарождением человеческой цивилизации было сформировано умение сообщать информацию одним людям так, чтобы другие не имели к ней доступа. В качестве примера, можно привести священные писания древнего Египта и древней Индии [1].

С начала XV века начинается использование примитивных методов относительно содержания шифруемых текстов, например, использовались методы кодирования стеганографии. Так, большинство изобретаемых методов шифрования текста сводилось к многоалфавитной подстановке или перестановке. Применялись такие методы шифрования, как, шифр Цезаря, состоящий в замене каждой буквы алфавита на другую букву, отстоящую от нее в алфавите на определенное число позиций. Впоследствии начинают появляться печатные работы, в которых были обобщены и сформулированы на тот момент алгоритмы шифрования является труд “Полиграфия” немецкого аббата Иоганна Трисемуса. В этом труде он описал два важных открытия: способ заполнения полибианского квадрата и шифрование пар букв.

В XIX веке было сформулировано главное требование к криптографическим системам, которое остается актуальным и по нынешнее время: секретность шифра должна быть основана на секретности ключа, но не алгоритма. Так же в этом веке механизировали роторные криптосистемы, которые позволили организовать более высокую криптостойкость [2]. Одной из первых подобных систем, стала механическая машина.

Практическое распространение роторные машины получили только в XX веке. В 1917 году Эдвардом Хеберном была разработана немецкая шифровальная машина, более известная как Enigma. Более активно она

использовалась во времена второй мировой войны. Так же были изобретены такие роторные устройства, как Sigaba, Typex и Orange. Первые криптоатаки стали возможны лишь в 40-х годах с появлением первых ЭВМ.

В 70-е годы с помощью производительных и компактных машин, стало возможным применять на практике блочные шифры, появилась возможность применять первые классы криптосистем. В 1978 году был принят стандарт шифрования DES, его автором был Хорст Фейстел, именно он описал модель блочных шифров на основе которой были построены другие, более стойкие криптосистемы. Так же в 70-е годы произошел инновационный прорыв в области криптографии, появились асимметричные криптосистемы, которые в свою очередь не требовали передачи секретного ключа между сторонами. С появлением асимметричной криптографии человечеству стало доступно сразу два новых направления, такие как, системы электронной цифровой подписи и электронных денег.

На рубеже XX века стали появляться новые направления в области криптографии: квантовая криптография, вероятностное шифрование и другие. Актуальной остается, и задача по совершенствованию симметричных криптосистем.

ГЛАВА 1. Постановка задачи, определение требований и обзор существующих методов

1.1 Общая постановка задачи

Цель работы состоит в разработке оптимального подхода к получению частично гомоморфного алгоритма шифрования.

Для достижения поставленной цели необходимо решить следующие задачи:

- провести анализ предметной области;
- исследовать существующие решения;
- сформулировать требования к своей системе решения поставленной задачи;
- сформировать алгоритм шифрования, удовлетворяющий предъявленным требованиям.

1.2 Основные определения

В данном разделе, используем λ для обозначения уровня криптостойкости системы. Определим гомоморфную криптосистему как $\xi = (KeyGen_\xi, Enc_\xi, Dec_\xi, Eval_\xi)$ – это есть соответственно четверка вероятностных полиномиальных шагов.

- Генерация ключа – алгоритм $(evk, sk) \leftarrow KeyGen_\xi(I^\lambda)$ принимает на вход параметр криптостойкости λ, выдает соответственно открытый ключ для вычисления evk и секретный ключ sk для шифрования и расшифрования.
- Шифрование – алгоритм $c \leftarrow \mathrm{Enc}_\xi(sk, m)$ принимает на вход секретный ключ sk и сообщение открытого текста в размере одного бита $m \in \{0,1\}$ и выдает соответствующий шифротекст c.

- Дешифрование – алгоритм $m^* \leftarrow \mathrm{Dec}_\xi\ (sk, c)$ принимает на вход секретный ключ sk, шифротекст c и выдает сообщение открытого текста $m^* \in \{0,1\}$.
- Гомоморфное вычисление – алгоритм $c_f \leftarrow \mathrm{Eval}\ (f, c_1, \ldots, c_l)$ получает на вход ключ для вычисления evk, функцию $f : \{0,1\}^l \rightarrow \{0,1\}$ и набор из l шифротекстов $c_1, \ldots, c_l$ и выдает шифротекст c_f.

1.3 Обзор существующих методов

1.3.1 Криптосистема RSA

Рассматриваемая криптосистема RSA [3] является одной из самых популярных, упоминается во многих источниках, связанных с безопасностью данных. Сама система использует открытый ключ. Также эта частично гомоморфная система использует односторонние функции, они же обладают свойствами:

- Если известно, то $f(x)$ вычислить просто.
- Если известно $y = f(x)$, то для вычисления x нет простого, то есть более эффективного пути.

Алгоритм криптографической системы RSA обладает сложностью, сравнимой со сложностью «задачи факторизации» произведения двух больших простых чисел. Шифрование подразумевает сложность, которую можно обозначить как возведение в степень большого числа. Дешифрование, приемлемое по длительности подразумевает вычисление функции Эйлера от указанного большого числа. Чтобы достичь такого требования, т.е. приемлемого времени, необходимо обладать информацией о разложимости этого числа на простые множители. В алгоритме RSA, каждый участник системы располагает как открытыми ключом (public key), так и закрытым ключом (private key). Причем каждый такой ключ состоит из пары целых чисел, и каждый участник системы в праве создавать как открытый, так и закрытый ключ. Кроме того, следует отметить правило, что закрытый ключ

необходимо держать в секрете и его нельзя никому разглашать, открытый же ключ можно публиковать и сообщать кому угодно.

Открытый и закрытый ключи каждого участника обмена сообщениями в криптосистеме RSA образуют «согласованную пару» в том смысле, что они являются взаимно обратными, то есть:

- для каждых допустимых пар открытого и закрытого ключей (p, s);
- существуют соответствующие функции шифрования $E_p(x)$ и расшифрования $D_s(x)$ такие, что для каждых сообщений $m \in M$, где M – множество допустимых сообщений, $m = D_s(E_p(m)) = E_p(D_s(m))$.

Пусть n – модуль RSA, а e – открытый ключ шифрования. Тогда функция шифрования будет иметь вид: $E(x) = x^e \bmod n$. Покажем гомоморфизм по умножению: $E(m_1) * E(m_2) = m_1^e * m_2^e \bmod n = (m_1 m_2)^e \bmod n = E(m_1 m_2)$.

1.3.2 Криптосистема Эль – Гамаля

Криптографическая схема Эль-Гамаля [4] основана на сложности в вычислении дискретных логарифмов в конечном поле. Можно сказать, что данный алгоритм относится к числу тех, которые принадлежат к числу вероятностного шифрования. Схему можно найти в широком использовании алгоритмов электронной подписи и непосредственно в шифровании.Так же можно отметить, что алгоритм обладает свойством ре-рандомизации. В 1985 году это схема была предложена Эль-Гамалем. В отличие от RSA алгоритм Эль-Гамаля не был запатентован и, поэтому, стал более дешевой альтернативой, так как не требовалась оплата взносов за лицензию. Считается, что алгоритм попадает под действие патента Диффи-Хеллмана.

Сообщение M должно быть меньше числа p. Сообщение шифруется следующим образом:

1. Выбирается сессионный ключ – случайное целое число такое, что $1 < k < p-1$.

2. Вычисляются числа $a = g^k \ mod\ p$ и $b = y^k$ M $mod\ p$.
3. Пара чисел (a, b) является шифротекстом.

Нетрудно заметить, что длина шифротекста в схеме Эль – Гамаля длиннее исходного сообщения M вдвое.

Дешифровка происходит следующим образом. Зная закрытый ключ x, исходное сообщение можно вычислить из шифротекста (a, b) по формуле: $M = b\ (a^x)^{-1}\ mod\ p$. При этом нетрудно проверить, что $(a^x)^{-1} \equiv g^{-kx}\ (mod\ p)$ и поэтому $b\ (a^x)^{-1} \equiv (y^k\ M)*g^{-xk} \equiv (g^{xk}\ M)*g^{-xk} \equiv M\ (mod\ p)$. Так же для практических вычислений больше подходит следующая формула: $M = b(a^x)^{-1}\ mod\ p = b * a^{(p-1-x)}\ mod\ p$.

1.3.3 Криптосистема Пэйе

Алгоритм Пэйе [5] был впервые предложен в 1999 году, ученым Паскалем Пэйе. Суть его подхода состоит в поиске корня остатка от деления по модулю с описанием сложности такого алгоритма. А также описана применимость алгоритма в криптографических целях и указаны некоторые нюансы обеспечения безопасности криптосистемы. Однако позднее, в ходе изучения данного алгоритма, было показано, что оригинальная схема криптосистемы, перестает быть уязвимой к атакам подобного рода. Далее покажем работу алгоритма Пэйе с точки зрения генерации ключей, шифрования и дешифрования.

Генерация ключей происходит следующим образом:

1. Выбираются два больших простых числа p и q, такие что
$$gcd\ (pq,\ (p\text{-}1)\ (q\text{-}1)\ ) = 1.$$
2. Вычисляется $n = pq$ и $\lambda = lcm\ (p\text{-}1,\ q\text{-}1)$.
3. Выбирается случайное целое число g, такое что $g \in Z^*_{n^2}$.
4. Вычисляется $\mu = (L\ (g^\lambda\ mod\ n^2))^{-1}\ mod\ n$, где $L(u) = \frac{u-1}{n}$.

Где открытым ключом является пара (n, g), а закрытым ключом пара (λ, μ).

Далее покажем шифрование, которое основано на следующих правилах:

1. Предположим, что необходимо зашифровать открытый текст m, $m \in Z_n$.
2. Выбираем случайное число r, $r \in Z^*_n$.
3. Вычисляем шифротекст $c = g^m * r^n \ mod \ n^2$.

Дешифровка:

1. Принимаем шифротекст $c \in Z^*_n{}^2$.
2. Вычисляем исходное сообщение $m = L\ (c^\lambda \ mod \ n^2) * \mu \ mod \ n$.

1.3.4 Криптосистема Гольдвассер – Микали

Криптографическая система с открытым ключом, которая была разработана в 1982 году [6]. Данная система является первой вероятностной схемой шифрования с открытым ключом. Однако эта криптосистема проявила себя неэффективной, так как шифротекст может быть более длинным, чем шифруемое сообщение. Алгоритм шифрует текст сообщения бит за битом, причем сложность работы, связанная с поиском отдельного зашифрованного бита в тексте c, должна принадлежать множеству QR_N.

С точки зрения генерации ключей, шифрования и дешифрования, причем будем использовать такие понятия как, «участник 1» и «участник 2», которые и будут обмениваться сообщениями.

Для того чтобы сгенерировать ключ, «участнику 1» необходимо выполнить следующие операции:

1. Выбрать два случайных числа p и q, которые будут удовлетворять условию $|p| = |q|$ бит.
2. Вычислить значение $N = pq$.
3. Далее извлекаем случайное целое число y, которое будет удовлетворять условию $(\frac{y}{p}) = (\frac{y}{q}) = -1$.
4. Описать пару $(N, y,)$ в качестве открытого ключа, а пару (p, q), использовать как закрытый ключ.

Чтобы послать «участнику 1» зашифрованное сообщение $m = b_1 b_2 \ldots b_l$, «участник 2» выполняет следующие операции:

for (i = 1, 2…, l)

{

$x \leftarrow_U Z^*_N$;

if $(b_i = = 0)$ $c_i \leftarrow x^2 \pmod N$;

else $c_i \leftarrow yx^2 \pmod N$;

}

В данном фрагменте «участник 1» посылает сообщение «участнику 2» $E_N(m) \leftarrow (c_1, c_2, \ldots, c_l)$.

Получив кортеж $(c_1, c_2, \ldots, c_l)$ «участник 1» выполняет следующую процедуру:

for (i = 1, 2…, *l)*

{

if ($c_i \in QR_N$) $b_i \leftarrow 0$;

$b_i \leftarrow 1$;

}

set $m \leftarrow (b_1, b_2, \ldots, b_l)$.

1.3.5 Криптосистема Меркла – Хеллмана

Криптосистема Меркла – Хеллмана [7] была основана в 1978 году. Эта система является одной из первых с открытым ключом. В ее основу была положена «задача о рюкзаке», но она оказалась криптографически нестойкой и в следствии этого не обрела должной популярности. Далее мы рассмотрим математическое описание алгоритма.

Генерация ключа:

Для того чтобы зашифровать n – битное сообщение, выберем возрастающую последовательность из n ненулевых натуральных чисел такую, что $w = (w_1, w_2, \ldots, w_n)$. Далее случайным образом выберем целые взаимно простые числа q и r, что $q > \sum_{i=1}^{n} w_i$. Число q здесь выбирается таким образом,

чтобы гарантировать уникальность шифротекста. Если оно будет хотя бы чуть меньше, то может возникнуть ситуация, где одному шифротексту может соответствовать несколько открытых текстов. Далее рассмотрим последовательность $\beta = (\beta_1, \beta_2, \ldots, \beta_n)$, где каждый элемент этой последовательности вычисляется по формуле $\beta_i = rw_i \ mod \ q$. Тогда как β и будет открытым ключом, а закрытый ключ образует последовательность (w, q, r).

Шифрование:

Для того, чтобы зашифровать сообщение, необходимо вычислить число c, которое и будет являться шифротекстом $c = \sum_{i=1}^{n} \alpha_i \beta_i$.

Дешифровка:

После того как полученный текст был зашифрован, получатель должен определить биты сообщения, которые удовлетворяли бы данной формуле $c = \sum_{i=1}^{n} \alpha_i \beta_i$. При этом, стоит заметить, что если бы β_i было выбрано случайным, то такая задача была бы неразрешима, таким образом Дешифрование будет происходить только в том случае, если закрытый ключ (w, q, r) известен. Далее получатель зашифрованного сообщения вычисляет

$ć \equiv cs \ (mod \ q)$. Из этого тождества следует, что $ć \equiv cs \equiv \sum_{i=1}^{n} \alpha_i \beta_i \ s \ (mod \ q)$, а из того , что $rs \ mod \ q = 1$ и $\beta_i = rwi \ mod \ q$. Тогда $ć \equiv \sum_{i=1}^{n} \alpha_i w_i \ (mod \ q)$, так как значение $\sum_{i=1}^{n} \alpha_i w_i$, находится в интервале [0, q-1]. Таким образом получателю сообщения необходимо определить α_i.

1.3.6 Алгоритм Полига – Хеллмана

Алгоритм Полига – Хеллмана [8] был опубликован в 1978 году, американскими математиками Рональдом Сильвером и Стивеном Полигом. Предложенный алгоритм является детерминированным и одной из особенность этого метода криптографии является то, что для простых чисел специального вида можно находить логарифм за полиномиальное время. Пусть задано сравнение вида $a^x \equiv b \ (mod \ p)$, а также известно разложение числа

p-1 на простые множители p-1 = $\prod_{i=1}^{k} q_i{}^{\alpha_i}$, где необходимо найти число x, $0 \leq x$ <p-1, которое удовлетворяло бы $a^x \equiv b \pmod p$. Суть в том, чтобы найти x по модулям $q_i{}^{\alpha_i}$ для всех i, для этого необходимо решить сравнение $(a^x)^{(p-1)/q} \equiv b^{(p-1)/q}$.

1.3.7 Криптосистема Бенало

В заключении первой главы, опишем еще один алгоритм шифрования. В данной криптосистеме открытым ключом является модуль m, тогда как функция шифрования сообщения x представлена в виде $\mathcal{E}(x) = g^x r^c \bmod m$, для случайного $r \in \{0, \ldots, m-1\}$. Тогда гомоморфное свойство этого алгоритма будет проявляться в виде: $\mathcal{E}(x_1) * \mathcal{E}(x_2) = (g^{x1} r_1{}^c)\,(g^{x2} r_2{}^c) = g^{x1+x2}(r_1 r_2)^c = \mathcal{E}(x_1 + x_2 \bmod c)$.

ГЛАВА 2. Предлагаемый метод частично гомоморфного шифрования

2.1 Введение в предлагаемую систему

Удивительно, но два десятилетия спустя после открытия схем шифрования с ассиметричным ключом в запасе криптографии все еще остается очень мало схем ассиметричного шифрования. Следовательно, поиск новых таких схем является непростой задачей. Более того, поиск таких схем зачастую выглядит безнадежным – большинство найденных схем либо оказываются взломанными, либо не переживают сравнения с RSA.

Схема, которая предлагается в работе, использует целое n являющееся произведением простых p и q. Однако, она во многом отличается от RSA. Рассмотрим основные отличия предлагаемой схемы:

1. данные шифруются при экспоненциальной сложности вместо возведения в степень;
2. используется специальная «лазейка» для расшифровки;
3. защищенность схемы строится некоторым другим способом;
4. обладает некоторыми «алгебраическими» свойствами, которые могут оказаться полезными в практических применениях.

Коротко прокомментируем эти различия: первое может дать большое преимущество в средах, с большим объемом оперативной памяти – в таких средах возможно большое ускорение операций экспоненциации. Второе – очевидно представляет интерес, в виду упомянутого вначале факта о небольшом количестве существующих схем. Без погружения в технические детали упомянем только, что «функция с лазейкой» получается путем помещения небольших простых множителей в q-1 и p-1. Чтобы понять, что значит третье отличие, нужно заметить, что если можно разложить на множители $mod\ n$, то криптосистему и RSA можно взломать. Однако, остается не совсем ясным можно ли принимать криптосистему RSA в качестве

эквивалента по устойчивости к взломам. По этой причине, гипотеза того, что RSA безопасно, стала обособленным предположением, формально более сильным чем указанное разложение. Предлагаемая криптосистема относится к другой гипотезе, также формально более сильной, чем разложение и известной как гипотеза большей остаточности. Это может помочь понять, как эти различные гипотезы соотносятся. И, наконец, поясним алгебраические свойства предлагаемой схемы с помощью примера: предположим, что кто-то хочет снять незначительное количество u с баланса m некоторого счета; также предположим, что баланс дан в зашифрованной форме E(m) и, что служащий, выполняющий операцию не имеет доступа к расшифровке. Криптосистема, которая предлагается в данной работе решает проблему, вычисляя E(m)/E(u) $mod\ n$, что является шифрованием нового баланса m-u.

Возможность выполнять алгебраические операции такие как сложение или вычитание, имея дело только с криптограммами, имеет потенциальное применение в нескольких контекстах:

1. в схемах голосования можно получить суммарный результат, не расшифровывая индивидуальные голоса;
2. в сфере водяных знаков, это позволяет добавить метку, к зашифрованным данным.

Тем не менее, в этих примерах часто требуется зашифровать данные из небольшого множества S и, как хорошо известно, детерминированные криптосистемы вроде RSA не имеют здесь большого успеха: чтобы расшифровать E(a) злоумышленник может просто сравнить зашифрованный текст со всеми имеющимися, и восстановить исходный текст. Чтобы преодолеть эту сложность, необходимо использовать вероятностное шифрование, в котором каждый открытый текст имеет много соответственных шифров, зависящих от некоторого случайного параметра, выбранного во время шифрования. Такой выбор должен сделать невозможным вскрытие исходного текста, даже если шифруются данные из очень малого множества. Это сильное требование носит название «семантической безопасности». Как

дальнейшее отличие от RSA, криптосистема, предложенная в данной работе, имеет очень естественную вероятностную версию, с доказанной семантической безопасностью.

Вероятностные гомоморфные схемы шифрования, предложенные до настоящего момента, имеют один серьезный недостаток – плохую пропускную способность. Требуется несколько килобит, чтобы зашифровать всего несколько битов.

Как было упомянуто предлагаемый подход является практико-ориентированным. При этом основания системы, генерация ключей, шифрование и дешифровка предлагаются с учетом эффективности. Также проводится анализ безопасности на неформальном уровне и выводятся минимальные параметры.

2.2 Основные положения и генерация ключа

Пусть σ бесквадратное нечетное B-гладкое число, где B – небольшое целое и пусть $n=pq$ RSA модуль, такой, что $\phi(n)$ делится на σ и является простым по отношению к $\phi(n)/\sigma$. Обычно, считается что B 10-битное целое и n имеет длину по меньшей мере 768 бит. Пусть g будет элементом, чей мультипликативный порядок по модулю n это большой множитель σ. Введем n и g, p и q и σ оставим в секрете. Сообщение m меньшее чем σ зашифровано $g^m \bmod n$; дешифрование производится, используя простые множители σ.

Генерация модуля выглядит следующим образом: выберем семейство p_i из k малых нечетных различных простых чисел, где k четное. Пусть $u = \prod_{i=1}^{\frac{k}{2}} p_i$, $\upsilon = \prod_{\frac{k}{2}+1}^{k} p_i$ и $\sigma = u\upsilon = \prod_{i=1}^{k} p_i$. Выберем два больших простых числа a и b таких что, $p = 2au + 1$ и $q = 2b\upsilon + 1$ простые и пусть $n = pq$.

Однако, время подобной генерации резко возрастает с увеличением модуля: a должно быть выбрано в правильном диапазоне, и протестировано на простоту, так же, как и $p = 2au+1$, до тех пор, пока оба теста одновременно не

дадут положительный результат. Вместо этого, предлагается сгенерировать a, b, v, u независимо от требований к простоте p, q, и использовать пару 24 – битных «настраиваемых простых» p' и q' (не используемых в процессе шифрования), таких, что, $p = 2aup' + 1$ и $q = 2b\upsilon q' + 1$ простые. Чтобы избежать интерференции с механизмом шифрования, рекомендуется убедиться, что НОК(p' q', σ) = 1 и $p' \neq q'$. На практике такой подход немного медленнее, чем у RSA генерация такого же по размеру ключа.

Чтобы выбрать g можно взять случайное число и убедиться, что оно имеет порядок ϕ(n)/4. Основная идея состоит в том, чтобы убедиться, что g не p_i-я степень для каждого $i \leq k$, проверяя, что $g^{\frac{\phi(n)}{pi}} \neq 1 mod\ n$. Вероятность успеха: $\pi = \prod_{i=1}^{k}(1 - \frac{1}{pi})$, чей логарифм: $\ln(\pi) \simeq -\sum_{i=1}^{k}\frac{1}{pi}$. Если p_i первыми k простыми, это может быть вычислено как $-\ln\ln k$ и результаты являются достаточно приемлемыми для общей вероятности в вид $\pi \cong 1/\ln k$. Другой метод состоит в выборе для каждого индекса $i \leq k$, случайного g_i до тех пор, пока оно не является степенью p_i-ого. С очень высокой вероятностью $g = \prod_{i=1}^{k} gi^{\sigma/pi}$ имеет порядок ϕ(n)/4.

2.3 Шифрование и дешифрование

Шифрование состоит из единичной модульной экспоненциации: сообщение m, меньшее чем σ, шифруется как $g^m\ mod\ n$. Важно отметить, что знание σ не требуется. Нижняя граница (предпочтительно степень двойки) является достаточной, но остается неясным, насколько важно для безопасности схемы хранить σ в секрете. Тем не менее, если σ секретно, необходимы некоторые меры предосторожности [9], [10].

Также, вообще говоря, нет причины, по которой p_i-ые должны быть простыми. Схема будет работать, с учетом соответствующих изменений, если p_i-ые взаимно просты. Следовательно, они могут быть выбраны как степени простых, что поможет увеличить вариативность схемы.

Дешифрование основывается на китайской теореме об остатках [11]. Пусть p_i-ые, $1 \leq i \leq k$ являются простыми множителями σ. Алгоритм вычисляет $m_i = m \bmod p_i$, и получает результат используя китайскую теорему об остатках. Чтобы найти m_i, имея зашифрованный текст $c = g^m \bmod n$, алгоритм вычисляет $c_i = c^{\frac{\phi(n)}{pi}} \bmod n$, что является в точности $g^{mi\frac{\phi(n)}{pi}} \bmod n$. Это вытекает из следующих несложных вычислений, где $y_i = \frac{(m-mi)}{pi} : c_i = c^{\frac{\phi(n)}{pi}} = g^{\frac{m\phi(n)}{pi}} = g^{\frac{(mi+yipi)\phi(n)}{pi}} = g^{\frac{mi\phi(n)}{pi}} g^{yi\phi(n)} = g^{\frac{mi\phi(n)}{pi}} \bmod n$.

Сравнивая этот результат со всеми возможными степенями $g^{\frac{j\phi(n)}{pi}}$ можно найти верное значение m_i. Другими словами, это можно записать следующим псевдокодом:

for $j = 0$ to $p_i - 1$ until $c_i = g^{\frac{j\phi(n)}{pi}} \bmod n$.

Исходный текст m таким образом, может быть вычислен следующей процедурой:

for $i = 1$ to k
{
let $c_i = c^{\frac{\phi(n)}{pi}} \bmod n$
for $j = 0$ to $p_i - 1$
{ *if* $c_i == g^{\frac{j\phi(n)}{pi}} \bmod n$ *let* $m_i = j$}
}
x = ChinaReminder ($\{m_i\}$, $\{p_i\}$)

Основная операция, использующаяся в этом алгоритме – экспоненциация по модулю, сложности $\log^3(n)$ повторенная меньше, чем: $kp_k < \log(n)p_k \cong \log(n)k \log(k) < \log^2(n) \log\log(n)$ раз. Дешифрование соответственно занимает $\log^5(n) \log\log(n)$ битовых операций.

Это очевидно хуже, чем $\log^3(n)$ сложности RSA, но шифрование может быть оптимизированно если таблица запоминает все возможные значения

t[i, j] = $g^{j\phi(n)/p_i}$ для $1 \le i \le k$, $1 \le j \le i$: значение m_i для исходного сообщения m mod p_i находится по таблице, как только вычислено с $\frac{\phi(n)}{p_i} \bmod n$. Необязательно сохранять все $g^{\frac{j\phi(n)}{p_i}}$. Любая хэш-функция, которая различает $g^{\frac{j\phi(n)}{p_i}}$ и $g^{\frac{j'\phi(n)}{p_i}}$ для $j \neq j'$ подойдет. В практических терминах будет достаточно нескольких байт, например, приблизительно $2|p_i|$ бит из каждого t[i, j]. Также возможно использование хэш-функций, которые не различают значения $g^{\frac{j\phi(n)}{p_i}}$: необходимое значение находится по таблице хэшей $g^{\frac{2^l j\phi(n)}{p_i}}$, для $l = 1,2\ldots$ до тех пор, пока не пропадет неопределенность.

2.4 Пример

Генерация ключа для $k = 6$, $p = 21211 = 2 * 101 * 3 * 5 * 7 + 1$, $q = 928643 = 2* 191 * 11 * 13 * 17 + 1$, $n = 21211 * 928643 = 19697446673$ и $g = 131$ дают таблицу:

	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6
j = 0	0001	0001	0001	0001	0001	0001
j = 1	1996	6544	1967	6273	6043	0372
j = 2	9560	3339	4968	7876	4792	7757
j = 3		9400	1765	8720	0262	3397
j = 4		5479	6701	7994	0136	0702
j = 5			6488	8651	6291	4586
j = 6			2782	4691	0677	8135
j = 7				9489	1890	3902
j = 8				8537	6878	5930
j = 9				2312	2571	6399
j = 10				7707	7180	6592
j = 11					8291	9771
j = 12					0678	0609

j = 13						7337
j = 14						6892
j = 15						3370

Таблица 1. Результаты генерации ключа

Где элемент $\{i, j\}$ содержит $g^{\frac{j\phi(n)}{p_i}}$ *mod n mod* 10000.

Шифрование:

Сообщение $m = 202$, $c = g^m \bmod n = 131^{202} \bmod 19697446673 = 519690214$

Дешифрование:

Экспоненциацией получаем:

1. $c^{\frac{\phi(n)}{p_1}}$ *mod n mod* 10000 = 1966
2. $c^{\frac{\phi(n)}{p_2}}$ *mod n mod* 10000 = 3339
3. $c^{\frac{\phi(n)}{p_3}}$ *mod n mod* 10000 = 2782
4. $c^{\frac{\phi(n)}{p_4}}$ *mod n mod* 10000 = 7994
5. $c^{\frac{\phi(n)}{p_5}}$ *mod n mod* 10000 = 1890

Откуда, по таблице:

1. *m mod* 3 = *table* (1966) = 1
2. *m mod* 5 = *table* (3339) = 2
3. *m mod* 7 = *table* (2782) = 6
4. *m mod* 11 = *table* (7994) = 4
5. *m mod* 13 = *table* (1890) = 7

И, по китайской теореме об остатках: $m = 202$.

2.5 Предлагаемые параметры и анализ безопасности

Предлагается брать $\sigma > 2^{160}$ и $|n| = 768$ бит, что есть минимальный размер модуля. Если найдено разложение n, то становятся известны a и b, а также $\phi(n)$. Следовательно, схема взломана. Однако, схема не обязательно является

эквивалентной разложению. На самом деле стойкость схемы в большей мере зависит от наличия, так называемых оракулов, которые могут сказать, является ли случайное число *x* p_i-ой степенью *mod n* для *i*=1…*k*. Эта проблема известна как проблема высокой остаточности, и на текущий момент является неразрешимой. Рассматривая эту версию схемы в предложенной работе, нет доказательства эквивалентности этой проблемы криптостойкости текущей схемы, но и вероятного вектора атаки также не было найдено. Также, эффективные методы факторизации, такие как квадратичное сито и NFS не получают преимуществ от знания того, что *u* (или *υ*) делится на *p*-1 (или *q*-1).

Теперь рассмотрим размер σ. Чтобы избежать расчета дискретных логарифмов методом маленьких-гигантских шагов, нужно взять σ достаточно большим, 2^{160} это минимум. Это может быть достигнуто перестановкой первых 30 нечетных простых, которые дадут примерно $2^{160.45}$. Также можно выбрать 16 последовательных простых из 10 бит. Так как таких простых 75, то энтропия 53 бита. Есть дальнейшие сложности, когда σ известна. Отметим, что: $4ab = \frac{\phi(n)}{\prod_{i=1}^{k} p_i} = \frac{n-p-q+1}{\sigma}$, следовательно, $4ab$ отличается от *n*/σ только на $\epsilon = -\frac{p+q-1}{\sigma}$. Числитель имеет размер $\frac{|n|}{2}$, следовательно, если он не превосходит знаменатель на большое число бит, то ab известно и шифрование взломано.

Когда точное разложение множителей σ на *u* и υ известно, предыдущий анализ можно продвинуть дальше. Сокращая соотношение $n = (2au + 1)(2bv + 1)$ *mod u* находим, что $n = 2bv + 1 \mod u$ и можем вычислить, *d* = *b mod u*. Похожим образом, можно вычислить *c* = *a mod* υ. Пусть $a = rv + c$ и $b = su + d$, r и s, неизвестны, тогда используя тот факт, что $\sigma = uv$, получаем выражение: $n = (2rvu + 2cu + 1)(2suv + 2du + 1) = 4rs\sigma^2 + 2\sigma[r(2dv + 1) + s(2cu + 1)] + (2cu + 1)(2dv + 1)$, которое в свою очередь имеет форму: $n = 4rs\sigma^2 + 2\sigma(\alpha r + \beta s) + \gamma$.

С известными α, β, γ. Сокращая на $mod\ \sigma^2$, получаем $\delta = \alpha r + \beta s\ mod\ \sigma$, заметим, что пара r, s лежит на двумерной решетке, определенной: $L = \{(x, y)|\alpha x + \beta y = \delta\ mod\ \sigma\}$.

Также, легко увидеть, что α и β ограничены 2*σ, а γ ограниченна $4*\sigma^2$. Из этого получаем: $rs \leq \frac{n}{4\sigma^2} \leq rs + r + s + 1 = (r + 1)(s + 1)$.

Следовательно пара (r, s) лежит очень близко к границе кривой C с уравнением $xy = \frac{n}{4\sigma^2}$. Точнее, расстояние между парой (r, s) и кривой не превышает $\sqrt{2}$. Это определяет геометрическую область, включающую (r, s). Также, обычно, процедура генерации ключа накладывает ограничения, которые сокращают диапазон возможных параметров. По этой причине кривую можно заменить линией $x + y = \frac{\sqrt{n}}{2\sigma}$, чтобы вычислить площадь A. Что приводит к приближению: $O(\frac{\sqrt{n}}{\sigma})$. Количество точек на сетке L на этой области измеряется отношением размера A и определитель: $\frac{\sqrt{n}}{2\sigma}$. Для обеспечения безопасности необходимо убедиться, что это множество за границей поиска, что можно выразить следующим соотношением: $n \gg \sigma^4$. Заметим, что отношение $\frac{|n|}{\sigma}$ - скорость роста криптосхемы. Разумеется, необходимо сделать ее настолько низкой, насколько возможно. С другой стороны, имея ввиду все выше сказанное, необходимо чтобы соотношение $\frac{|n|}{4} - \sigma$ было достаточно большим. Асимптотически это достижимо если скорость роста больше 4. Для реальных приложений можно принять во внимание эвристическую границу, $\frac{|n|}{4} - \sigma \geq 128$, что сопоставимо с ранее предложенными минимальными параметрами. Большие параметры, дают незначительное улучшение в скорости роста.

2.7 Производительность

Несмотря на скорость роста криптосистема довольно эффективна: шифрование требует возведения постоянно 768 битного числа в 160-битную степень. Несколько пакетных и пред-обработочных приемов могут ускорить это вычисление, что является небольшим преимуществом над RSA.

Дешифрование несколько более сложно, так как требуется k экспоненциаций. Но их число может быть сокращено несколькими способами. Во-первых, во время вычисления $c^{\phi(n)/p_i}$ для каждого i возможно сохранить вначале $c' = c^{4ab} \; mod \; n$ и возвести c' в последовательные степени $\frac{\sigma}{p_i}$ таким образом, что последующие экспоненциации будут включать в себя 160-битные степени.

Проводя сбор данных по количеству модульных перемножений, полученных в результате работы схемы |n| = 768 и выбором шестнадцати 10-битных простых показывает, что общее количество модульных перемножений снижается до 2352: 912 для вычисления *c'* и 1440 для оставшегося. Однако, часть с «умножением» также может быть амортизирована. Результирующая нагрузка ниже, чем требуется для пары RSA шифровок с похожим модулем.

К сожалению, есть один недостаток в уменьшении значения *k*: в случае 30 простых необходимо хранить 1718 различных хэш-значений t[i, j]. Хэширование по двум байтам является достаточным, и результирует в общее требование по памяти равное 4 килобайтам. В случае использования 16 простых необходимо хэшировать по 3 байта, что ведет к потреблению памяти в 100 килобайт. Однако, как было упомянуто ранее размер хэш-таблицы может быть значительно сокращен по цене незначительного прироста в процессорном времени.

Другое ускорение может быть получено путем раздельного выполнения дешифровки $mod \; p$ и $mod \; q$, чтобы получить преимущество операндов меньшего размера. Только это сокращает нагрузку в четыре раза.

И, наконец, процесс дешифрования по сути происходит параллельно, так как каждое m_i может быть вычислено независимо от остальных.

Заключение

В данной работе был проведен анализ частично гомоморфных криптографических систем. В ходе анализа, был предложен ассиметричный алгоритм гомоморфного шифрования, который имеет практико-ориентированный подход. В результате изучения предметной области предложены методы генерации ключа, шифрования и дешифрования. Исследован вопрос безопасности и получены диапазоны параметров, при которых уровень безопасности криптосистемы остается приемлемым. Предлагаемая система сравнима по эффективности с криптосистемой RSA, в частности, по пропускной способности.

Список использованной литературы

1. Бауэр Ф. Расшифрованные секреты. Методы и принципы криптологии. М.: Мир, 2007, 550 с.
2. Жельников В. Становление науки криптологии // Криптография от папируса до компьютера. – М.: ABF, 1996, 335 с., ISBN 5-87484-054-0.
3. Rivest R. L., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems.,15
4. Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone. 11.5.2 The ElGamal signature scheme //Handbook of applied cryptography.,794
5. Варновский Н. П., Шокуров А. В. Гомоморфное шифрование Труды ИСП РАН, 2007, С. 27–36.
6. Мао В. Современная криптография: Теория и практика, М.: Вильямс, 2005, 768 с. ISBN 5-8459-0847-7
7. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си = Applied Cryptography. Protocols, Algorithms and Source Code in C., М.: Триумф, 2002., 816 с. , 3000 экз. — ISBN 5-89392-055-4.
8. J. Hoffstein, J. Pipher, J. H. Silverman. An Introduction to Mathematical Cryptography. - Springer, 2008., 524 с., ISBN 978-0-387-77993-5.
9. M. Rabin, Digitized signaturm and public-key functions as intractable as factorization, MIT/LCS/TR- 212, MIT Laboratory for Computer Science, 1979.
10. A. Shamir, MA for paranoids, CryptoBytes, vol. 1-3, pp. 1-4, 1995.
11. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии., М.: МЦНМО, 2003., 328 с.,1000 экз.,ISBN 5-94057-103-4.