**ПРОГРАММНОЕ СРАВНЕНИЕ АЛОРИТМОВ ШИФРОВАНИЯ RSA И ECC**

А.В. Коплик,

Студент группы 545-А НАУ «ХАИ»

[anatolykoplyk@gmail.com](mailto:anatolykoplyk@gmail.com)

**РЕЗЮМЕ**

Для удовлетворения потребностей пользователей криптографический алгоритм должен быть выбран на основе таких атрибутов как, безопасность и производительность. Одна из задач разработчика криптосистемы является взвешивание преимущества и недостатки и выбора алгоритмических инструментов, которые лучше соответствуют проблеме, которая будет решена.

RSA является самым популярным открытым ключом сегодня, но долгосрочные тенденции, такие, как распространение небольших, более простых устройств и увеличения потребности в безопасности сделают постоянное использование RSA более сложным с течением времени. Следовательно, криптография на эллиптических кривых (ECC) является подходящей альтернативой.

Эта работа сосредоточена на атрибуте производительности открытого ключа криптосистемы. Алгоритмы, которые я сравнивал это RSA , ECC. Я внедрил эти алгоритмы в .NET для того, чтобы выполнять тесты программного обеспечения, чтобы мы могли разобраться в относительной производительности каждого алгоритма и его соответствующими параметрами. Программное обеспечение на основе испытания проводятся для получения общего анализа генерации ключей, шифрования и дешифрования сообщения. Реализации в .NET в оконном интерфейсе. Каждый алгоритм тестируется на генерацию ключей и шифрование / дешифрование обычных, но больших файлы.

**КАТЕГОРИИ И ТЕМЫ**

Сетевая безопасность

**ОБЩИЕ ТЕРМИНЫ**

Алгоритмы, Безопасность.

**КЛЮЧЕВЫЕ СЛОВА**

Криптографические алгоритмы, криптография на эллиптических кривых, RSA, Шифрование / дешифрование, с открытым ключом.

**1 ВВЕДЕНИЕ**

Шифрование с открытым ключом основано на создание математических головоломок, которые трудно решить без определенных знаний о том, как они были созданы. Создатель хранит эту тайну знаний (секретный ключ) и публикует головоломки (открытый ключ).

Класс включает в себя головоломки решения уравнения ab = c для b при известных а и с. Такие уравнения с вещественными или комплексными числами легко решить с помощью логарифмов. Тем не менее, в большой конечной группе, поиск решений таких уравнений довольно сложен и известен как задача дискретного логарифмирования.

Данная работа посвящена аспектам реализации 2 открытых ключей криптографические алгоритмы: RSA и ECC и оценить алгоритмы в отношении времени, необходимого для генерации ключей, шифрования и дешифрования, а также цифровой подписи.

Рон Ривест, Ади Шамир и Леон Адлеман изобрели RSA систему в 1977 году. Он основан на предположении, что легко умножить два простых чисел, но трудно разделить результат снова на два простых числа.

Нил Коблиц изобрел криптографию на эллиптических кривых в 1987 году. Криптография на эллиптических кривых работает с точками на кривой. Безопасность этого типа шифрования с открытым ключом зависит от эллиптической кривой задачи дискретного логарифмирования. Основным преимуществом эллиптическую кривую криптография, что ключи могут быть гораздо меньше. Рекомендованный размер ключа составляет порядка 160 битов, а не 1024 бит как для RSA.

**2 АЛГОРИТМЫ ОТКРЫТЫХ КЛЮЧЕЙ**

В этом разделе мы обсудим проблемы генерации двух видов открытых ключей алгоритмами RSA и ECC.

**2.1 Алгоритм открытого ключа: RSA**

В криптографии, RSA является алгоритмом шифрования с открытым ключом. Это был первым алгоритм, как известно, пригодный для подписания, а также шифрования, а также одним из первых больших успехов в криптографии с открытыми ключами. RSA по-прежнему широко используется в электронной коммерции протоколов и, как полагают, безопасный при достаточно длинных ключах для современных реализаций.

**Операция**: RSA включает в себя открытый и закрытый ключи. Открытый ключ может быть известно каждому и используется для шифрования сообщений. Сообщения, зашифрованные с помощью открытого ключа, могут быть расшифрованы только с помощью закрытого ключа. **Открытый ключ** состоит из модуля N и открытой (или шифровальной) экспоненты e.

**Закрытый ключ** состоит из модуля N и закрытой (или дешифровальной) экспоненты D, которые должны храниться в секрете.

**Шифрование сообщений**: Алиса передает свой открытый ключ (n и е) Бобу и сохраняет секретный ключ секрет. Затем Боб хочет послать **M** сообщение Алисе. Сначала он превращает **М** в число m < n c использованием согласованного обратимого протокола схемы. Затем он вычисляет зашифрованный текст c, соответствующий:

c = me mod nc

Это может быть сделано быстро методом путем возведения возведение в квадрат. Затем Боб передает c Алисе.

**Расшифровка сообщений:** Алиса может восстановить m из c, используя ее

закрытый ключ d в следующем порядке:

m = cd mod n

Учитывая m, она может восстановить исходное сообщение **M**.

Процедуру расшифровки работает, потому что

cd ≡ (me)d ≡ med (mod n).

**2.2 Алгоритм открытого ключа: ECC**

Криптография на эллиптических кривых (ECC) представляет собой подход к криптографии с открытым ключом на основе алгебраической структуры эллиптических кривых над конечным полем. Эллиптические кривые часто используется в нескольких алгоритмах целочисленной факторизации, которые находят применение в криптографии, такие как, например, разложение Ленстра эллиптических кривых, но это использование эллиптических кривых обычно не называют "Криптографией на эллиптических кривых".

Шифрование с открытым ключом основано на создание математических головоломок, которые трудно решить без определенных знаний о том, как они были созданы. Создатель хранит эту тайну (секретный ключ) и публикует головоломки (открытый ключ). Головоломки могут быть использованы для шифрования сообщений таким образом, что только создатель может расшифровывать. Ранние системы с открытым ключом, такие, как Алгоритм RSA, использовали произведения двух больших простых чисел, как головоломку: пользователь выбирает два больших случайных простых чисел, как ее частный ключ, и публикует свой продукт как свой открытый ключ. Трудность факторизации гарантирует, что никто другой не может получить секретный ключ (т.е. двух простых множителей) из публичного ключа. Однако из-за последних достижений в факторизации, открытые ключи RSA должны теперь быть длиной тысячи бит, чтобы обеспечить надлежащую безопасность.

Другой класс головоломки предполагает решение уравнения ab = c для b когда а и с известными. Такие уравнения с реальными или комплексными числами легко решить с помощью логарифмов. Тем не менее, в большой группе конечных чисел, поиск решений таких уравнений вполне трудным и известно как проблема дискретного логарифма.

Эллиптической кривой называется плоская кривая, которая определяется уравнением формы:

*y2 = x3 + ax + b*.

Множество точек на такой кривой (то есть, все решения уравнение вместе с бесконечно удаленной точке) можно показать, чтобы сформировать в виде абелевой группы (с точкой на бесконечности как единица). Если координаты X и Y выбираются из большого конечного поля, решения образуют конечную абелевую группу. Проблема дискретного логарифма для таких групп эллиптических кривых, как полагают, более трудное, чем соответствующие задачи (мультипликативный группа ненулевых элементов) базовым конечным полем. Таким образом, ключи в эллиптических кривых могут быть выбраны намного короче для сопоставимого уровня безопасности.

**3 РЕАЛИЗАЦИЯ**

Не трудно найти криптографические алгоритмы. Доступно множество вариантов. Разработчик будет иметь для анализа алгоритм ее пригодности к его системе. Для этого он должен реализовать криптографические алгоритмы и выбрать более эффективный.

Так что в этом проекте я реализовал стандартные криптографические алгоритмы и сравнил их.

Программное обеспечение на основе испытания проводятся для получения общего анализа генерации ключей, шифрования и дешифрования сообщения.

Реализация в .NET формах. Каждый алгоритм проверяется для выполнения

следующие функции с помощью обычных, но больших файлов.

* шифрование с генерацией открытых и закрытых ключей
* дешифрование
* цифровая подпись
* проверка цифровой подписи

**3.1 Алгоритм RSA**

* Каждый субъект создает RSA открытый ключ и соответствующий закрытый ключ.
* Каждая сущность должна выбрать два больших простых чисел и создать открытый ключ и закрытый ключ. Выбранные два простые числа должны быть большими и того же размера.
* Тогда мы рассчитаем их продукт и выберите случайное число. Это случайное число и продукт станет открытым ключом.
* Вычислить уникальный номер, используя расширенный алгоритм Евклида. Это число и случайное число должно быть сравнимо по модулю с произведением простых чисел. Это уникальный номер будет закрытым ключом.

Теперь открытый ключ и закрытый ключи генерируются. Каждый объект будет опубликовать открытый ключ для всех и будет держать закрытый ключ в тайне.

Если любая сущность А хочет отправить сообщение для сущности B, то А должна использовать открытый ключ B , чтобы зашифровать сообщение и отправить зашифрованное сообщение для B. B в свою очередь, будет использовать свой закрытый ключ для расшифровки зашифрованного сообщения. Любая сущность, желающая отправить что-то другой сущности должна использовать для шифрования своих сообщений открытые ключи другой сущности. А получающая сущность в свою очередь будет расшифровывать его с помощью своих закрытых ключей.

*3.1.1 ПРОЦЕСС ШИФРОВАНИЯ И ДЕШИФРОВАНИЯ*

* Сообщение будет текст или изображения. Нам нужно найти его бинарное представление и разделить сообщение на блоки таким образом, что двоичные цифры должны быть в диапазоне от 0 до произведения простых чисел минус 1. Теперь каждый такой блок двоичных сообщений будет m и сущность B будет применять алгоритм шифрования на нем. Таким образом, каждый такой блок будет зашифрован и будет создан файл зашифрованного сообщения, который будет отправлен сущности А.
* Сущность А принимает зашифрованный файл, находит его двоичное представление и делит его на блоки. Она применяет алгоритм расшифровки по каждому блоку. В конечном итоге это создает расшифрованный файл, который будет ничем иным, как фактическое сообщение.

**3.2 Алгоритм ECC**

* Сначала мы возьмем кривую в форме y2 = x3 + ах + b. где a и b являются параметры кривой
* Затем выберем простое число.
* Использование точки добавления и точки удвоение вычислим точек на кривой.
* Выберите точку генерации из тех точек, порядок которых должен быть больше.
* Затем возьмите случайное число меньше порядка точки генерации как приватное число для каждой сущности. Это будет секретный ключ.
* Этот объект будет генерировать свой открытый ключ путем умножения числа генерации с приватным числом и опубликует точку.

*3.2.1 ПРОЦЕСС ШИФРОВАНИЯ И ДЕШИФРОВАНИЯ*

Первой задачей в этой системе заключается в кодировании m текста сообщения для передачи в качестве х-у точки Pm. Это будет точка Pm , которая будет зашифрована как зашифрованный текст, а затем расшифрована. Как и в случае системы с обменом ключами, шифрование / дешифрование системы требует точку G и эллиптическую группу Ep (a, b ) в качестве параметров.

* Каждый пользователь А выбирает закрытый ключ nA и генерирует открытый ключ PA = nA X G.
* Чтобы зашифровать и отправить сообщение Pm к B, A выбирает случайное положительное целое число x и производит зашифрованный текст Cm состоящий в пары точек
* Сm = { xG , Pm + xPB } . (A использует PB - открытый ключ Боба чтобы зашифровать сообщение)
* Для расшифровки зашифрованного текста , B умножает первую точку в паре на секретный ключ B и вычитает результат от из второй точки:

Pm + xPB - nB (xG) = Pm +x ( nBG ) – nB(xG) = Pm

Aлиса замаскировала сообщение Pm , добавив xPB к нему. Никто, кроме Aлисы не знает значение х, так что даже если PB представляет собой открытый ключ, никто может снять маску xPB . Тем не менее,Aлиса также включает в себя "ключ ", которого достаточно, чтобы снять маску, если знать секретный ключ nB. Для злоумышленника, чтобы восстановить сообщение, необходимо вычислить х для G и xG ,что является трудной задачей.

**4 РЕЗУЛЬТАТЫ РЕАЛИЗАЦИИ**

Я протестировал реализации RSA и ECC отдельно.

Испытания проводятся для сравнения времени, необходимого для шифрования, дешифрования, цифровой подписи и ее проверки, а также для дешифрования в обоих алгоритмах.

Результаты показаны на следующих таблицах и графиках.

Таблицы и графики для RSA:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Шифрование, секунд | | | | | | Дешифрация, секунд | | | | | |
| Ключ, бит | | | | | | Ключ, бит | | | | | |
| RSA | | | ECC | | | RSA | | | ECC | | |
| 384 | 512 | 1024 | 256 | 384 | 521 | 384 | 512 | 1024 | 256 | 384 | 521 |
| Размер файла, кБ | 25 | 2,60 | 3,27 | 34,14 | 24,37 | 49,56 | 109,64 | 0,46 | 0,53 | 1,08 | 0,15 | 0,14 | 0,15 |
| 50 | 5,32 | 6,68 | 73,74 | 48,46 | 100,64 | 222,28 | 0,92 | 1,07 | 2,21 | 0,28 | 0,29 | 0,29 |
| 75 | 7,74 | 9,89 | 118,10 | 73,81 | 150,71 | 332,74 | 1,40 | 1,59 | 3,30 | 0,45 | 0,42 | 0,43 |
| 100 | 10,61 | 13,15 | 146,31 | 97,81 | 207,38 | 445,42 | 1,87 | 2,15 | 4,39 | 0,57 | 0,62 | 0,57 |

**5 ВЫВОДЫ**

В современном цифровом мире существует огромный рост в использовании сети Интернет. Придет день, когда почти все работы могут быть осуществимы с Интернета. На одного программиста есть сотни хакеров. Таким образом, доля второго будет достаточной, чтобы уничтожить мир Интернета. Поэтому нам нужны мощные и быстрые алгоритмы шифрования, которые обеспечивают работу Интернет. Исследуя реализацию RSA и ECC алгоритмов, можно заключить, что операции в RSA сравнительно быстрее, чем ECC. У RSA генерации ключей шифрования быстрее, в то время как расшифровка медленнее. С другой стороны, при генерации ключей и шифрование ECC медленнее, в то время как расшифровка быстрее.

Отсюда вывод – RSA быстрее, но нужно сказать, что уровень безопасности, которую обеспечивает ECC – намного выше.