**Лабораторная работа №3**

**Роман Харченко 13 группа**

**Алгоритм ECDSA и RSA**

Алгоритм ECDSA (Elliptic Curve DSA) является аналогом алгоритма цифровой подписи DSA (Digital Signature Algorithm), реализованным с помощью эллиптических групп. Рассмотрим эллиптическую группу EM(a,b) и генерирующую точку G с порядком q, причем q простое число.  
Пользователи A генерируют свои ключи: секретный nA и открытый PA, где PA= nAG. Для постановки цифровой подписи под сообщением m пользователь A:

1. На основе хеш-функции h() находит хеш-код h(m) от m. В качестве хеш-функции должна использоваться криптографически стойкая функция, например SHA-1.
2. Генерирует случайное число k, 1 < k < q – 1.
3. Вычисляет значение kG = (x1,y1) и r = x1 mod q. Если r = 0, возвращается на шаг 2.
4. Вычисляетhttps://lh5.googleusercontent.com/wxiP6kFQ7DrCi9oSyGfoS1epWUR033cDRYmQ8Sic_fpdoVc6-CjMK5Rh3dI0aHt0QPN8e3eMXI3goW1vFysCRaASgFrH9GZdzwhSPZwGoaBzadNFAboGZGHwR7JOuTscw06CxCM(mod q) . Если s = 0, то возвращается на шаг 2.
5. Подпись сообщения m – это пара целых чисел (r,s).

Для проверки цифровой подписи пользователь B использует ту же эллиптическую группу EM(a,b), генерирующую точку G, открытый ключ PA и хеш-функцию h.

На основе хеш-функции h определяем хеш-код h(m) от m.

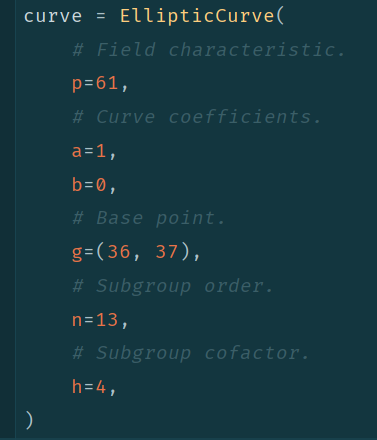
* Проверяем, принадлежат ли числа r и s интервалу от 1 до q – 1.
* Вычисляем w = m mod q.
* Вычисляем u1 = h(m)w mod q и u2 = rw mod q.
* Вычисляем u1G + u2PA = (x1\*, y1\*) и r\*=x1\*mod q.
* Равенство r\* = r удостоверяет подпись пользователя A.

Особое достоинство криптосистем на эллиптических кривых состоит в том, что по сравнению с системами на основе алгоритма RSA они обеспечивают существенно более высокую стойкость при равной трудоёмкости или, наоборот, существенно меньшую трудоёмкость при равной стойкости. В результате тот уровень стойкости, который достигается в RSA при использовании 1024-битных модулей, в системах на эллиптических кривых реализуется при размере модуля 160 бит, что обеспечивают более простую как программную, так и аппаратную реализацию.

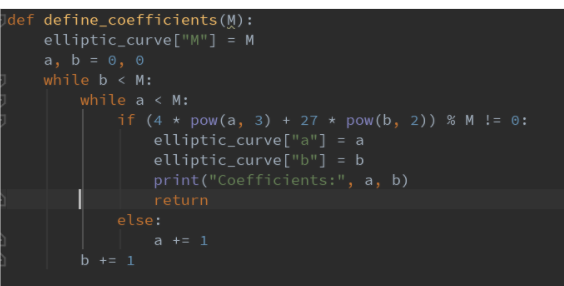
Для алгоритма ECDSA можно определить сложность приблизительно в O(n) (или O(2k/2) если рассматривать длину ключа) в случае алгоритма «маленьких и больших шагов». В случае алгоритма «Pollard’s p» временная сложность также составит O(n), но пространственная сложность будет только O(1).  Это по прежнему экспоненциальное время, но уже намного лучше атаки грубой силой.

Для алгоритма RSA Serpent(который был реализован в предыдущей лабораторной работе) можно указать временную сложность O(N · 2k). Сложность подделки сообщений, как и для алгоритмов с использованием эллиптический кривых составит O(2k/2).

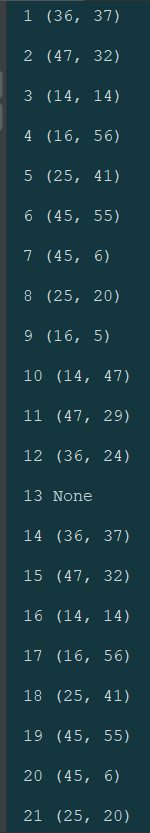
Подобранные параметры, исходя из варианта 4:



Расчёт значений a,b



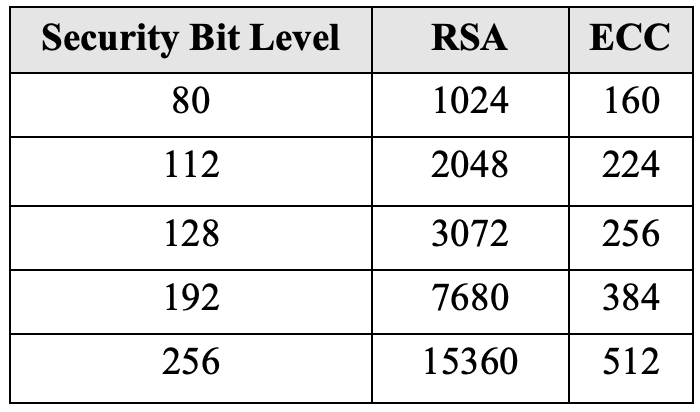
Циклические подгруппы, полученные программой:



Обмен сообщениями:

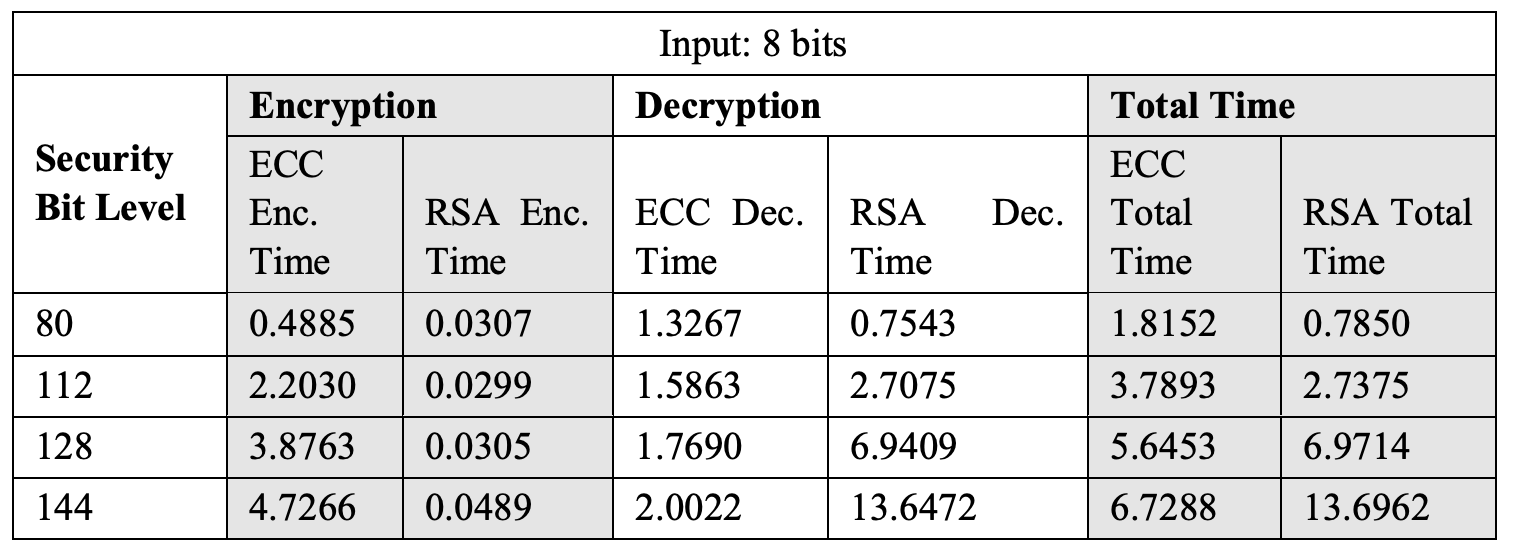


для того же уровня безопасности, в ECC можно использовать значительно меньшие параметры, чем в RSA. Например, для достижения уровня безопасности 112 бит RSA алгоритм требует размер ключа 2048 бит, в то время как ECC требует размер ключа 224 бит.

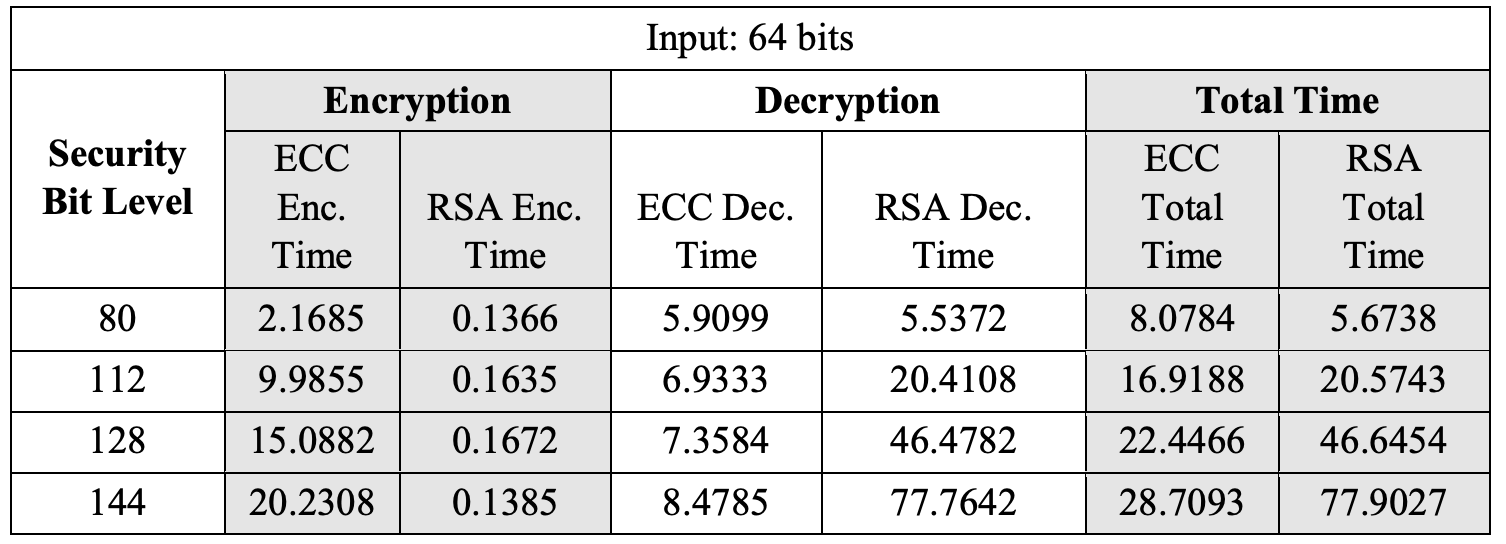


RSA очень эффективен в шифровании, но медленнее в дешифровании, а ECC медленнее в шифровании, но очень эффективен в дешифровании. В целом ECC более эффективен и безопасен, чем RSA.

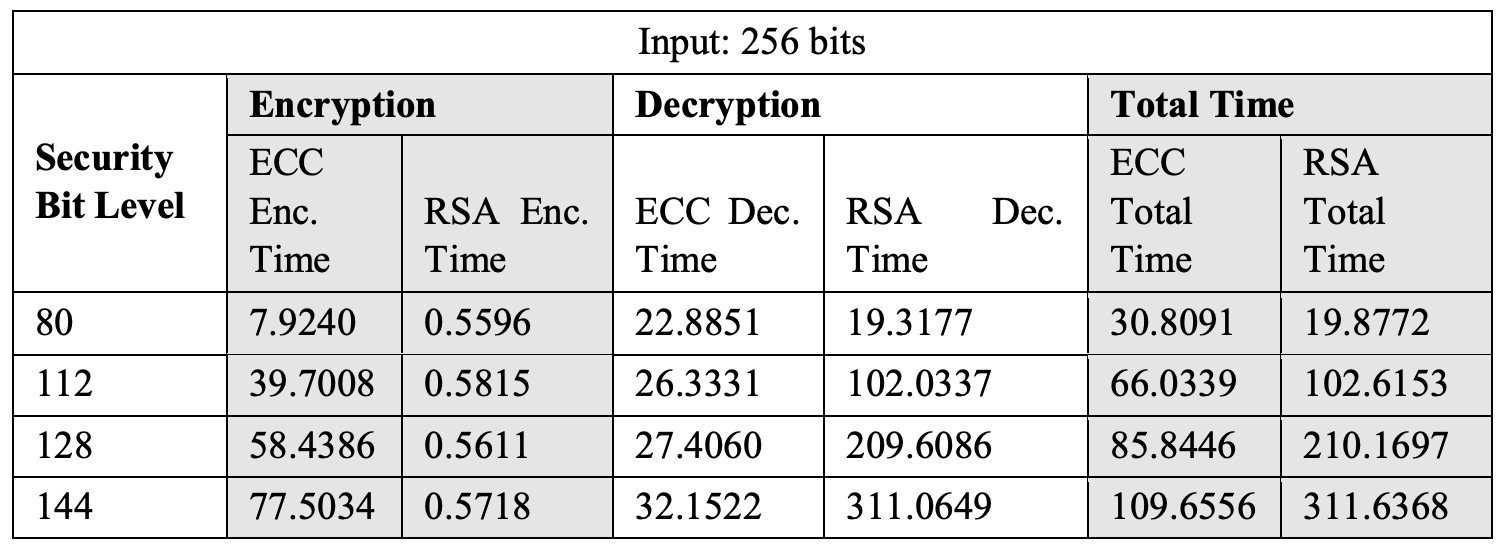
8 bits:



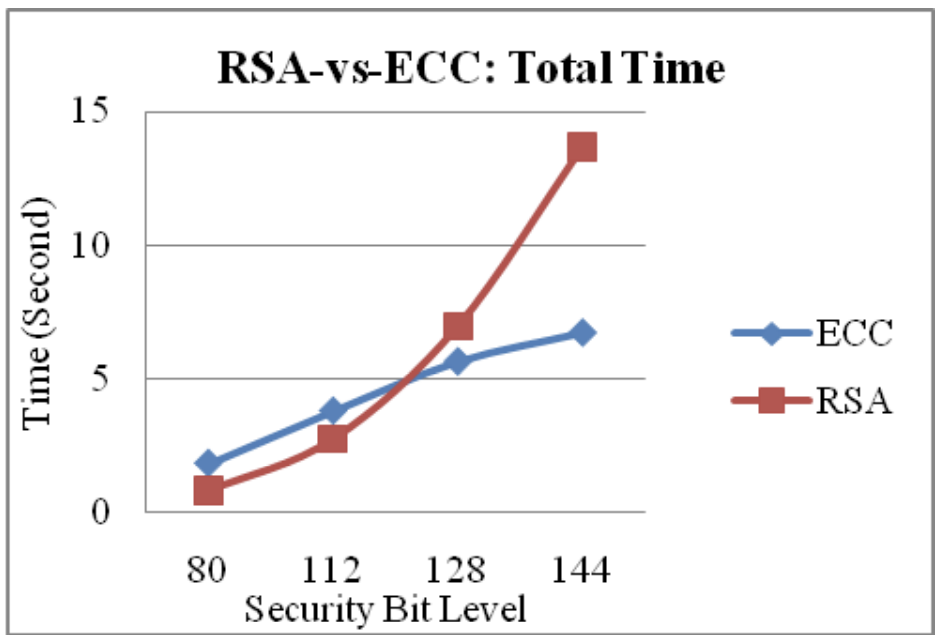
64 bits:



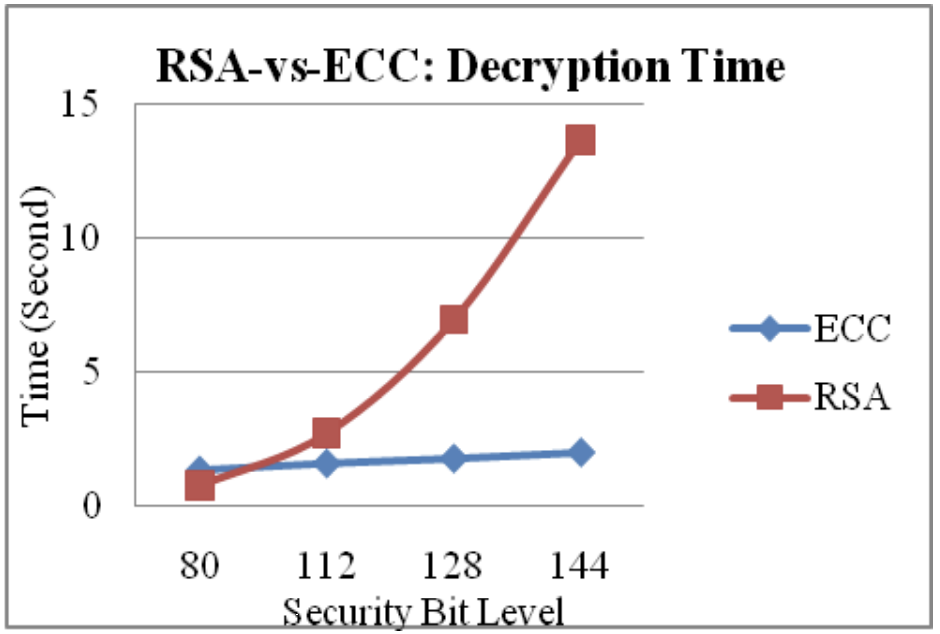
256 bits:



8 bits общее время:



8 bits расшифрование:



Для 64 и 256 bits ситуация аналогичная.

Если резюмировать то можно сказать, что ECC превосходит RSA по операционной эффективности и безопасность с меньшими параметрами.