Студент: Казміді І.Д.  
Група: ФБ21-мн  
  
Лабораторна робота №1

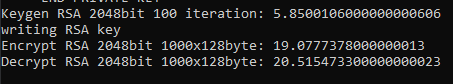
Підгрупа 2А

Порівняння криптографічних бібліотек OpenSSL, PyCryptoDome, RSA(pure python) для розробки криптографічної системи під ОС Windows.

Для порівняння швидкодії використовувалися наступні тести:  
Генерація 100 пар ключів RSA 2048bit  
Шифрування 128 байт даних 1000 разів  
Розшифрування 128 байт даних 1000 разів  
Умови тестування для всіх бібліотек однакові.

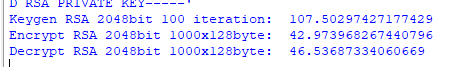
Була отримана наступна статистика:

|  |  |  |  |
| --- | --- | --- | --- |
|  | KeyGen | Encryption | Decryption |
| OpenSSL | 5.827420700000000231 | 18.984363800000000566 | 19.428972800000000376 |
| PyCryptoDome | 92.21123099327087 | 31.889445543289185 | 47.30909824371338 |
| RSA (pure python) | 452.77561831474304 | 5.596318960189819 | 10.960975646972656 |

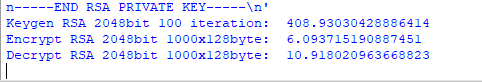
Одна з ітерацій перевірки OpenSSL:  


Для перевірки використовувалася готова збірка OpenSSL під Win64.

Одна з ітерацій перевірки PyCryptoDome:



Одна з ітерацій перевірки RSA(pure python):



Генерація ключів у даній бібліотеці може різнитися, в залежності від того, наскільки швидко буде згенеровано та перевірено прості числа. Через це у випадку генерації однієї пари ключів час генерації коливається від 1-2 секунд до 10 секунд.

Вибір бібліотеки для розробки криптографічної системи:

Якщо зважати на статистику, то можемо побачити, що найбільш збалансованою бібліотекою є OpenSSL, ця ж бібліотека має найбільший набір криптографічних функцій та алгоритмів.  
PyCryptoDome – повільніший ніж OpenSSL, набір функцій приблизно однаковий. Для застосунків, які не вимагають швидкодії криптографії – зручний варіант.  
RSA – бібліотека, яка реалізує функції лише для алгоритму RSA. Генерація ключів повільніша, оскільки методи перевірки великих чисел на простоту важко оптимізуються на Python, через що генерація однієї пари ключів може займати 2-10 секунд. Але дана бібліотека має найменший час шифрування та дешифрування.   
  
Вибір бібліотеки – OpenSSL, через наступні причини:  
- Задовільний рівень оптимізації   
- Велика кількість підтримуваних алгоритмів  
- Гнучкість застосування  
- Технічна підтримка з боку суспільства