Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Фізико-технічний інститут

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Вивчення криптосистеми RSA та алгоритму електронного підпису; ознайомлення з методами генерації параметрів для асиметричних криптосистем

Виконали:
студентки гр. ФБ-83
Григор’єва Ольга,
Мазуренко Віола

Перевірив:
Чорний О.М.

Київ – 2020

Мета та основні завдання роботи: Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.

Завдання:

1. Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч, використання готових реалізацій тестів не дозволяється.

2. За допомогою цієї функції згенерувати дві пари простих чисел p, q і 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $p1q1 \le pq$; p і q – прості числа для побудови ключів абонента А, 1 p і q1 – абонента В.

3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p,q) та відкритий ключ (n,e) . За допомогою цієї функції побудувати схеми RSA для абонентів А і В – тобто, створити та зберегти для подальшого використання відкриті ключі (e,n) , (,) 1 n1 e та секретні d і d1 .

4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і В. Кожна з операцій (шифрування, розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті ключові дані, які необхідні для її виконання. За допомогою датчика випадкових чисел вибрати відкрите повідомлення М і знайти криптограму для абонентів А и В, перевірити правильність розшифрування. Скласти для А і В повідомлення з цифровим підписом і перевірити його.

5. За допомогою раніше написаних на попередніх етапах програм організувати роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA. Протоколи роботи кожного учасника (відправника та приймаючого) повинні бути реалізовані у вигляді окремих процедур, на вхід до яких повинні подаватись лише ті ключові дані, які необхідні для виконання. Перевірити роботу програм для випадково обраного ключа $n.< k <0$

Результат роботи:

GENERATED for user A :

p = 0xf54f2e6a3be94ef8e1b72bc15ad27bc6c71c18ac282624bd3e3d29d45ceef447

q = 0x2d1673031313df29eb9a3cb94ca3c3dc4b5e8ee614271621d293a0b2de526cc7

(e,n,d) for user A looks like this:

(0x161658f77d1cba3617737e2d970b1d502b69affd48bf272fb6823d484743d2de0e11413045c6728a44c2afb8da7cf993651f913da2924c29f9ce38c6e52fd7ed ,
0x2b346e2c2db85841847b8a8d66350dc3f3c5f30ef686920d64f780d52652f66ed5ece03b6b037e91c113b24090ac16beb1c2cc652e3327b837655106dd8bd731 ,
0x299929a4b4e00f06b9b9c7992691d015898c9c395e905cbe7fb860e565ae217c678c03a4bb5d94ac82dadd96ebe1609af95356f4444fe95c163b7c63e195a77d)

Message looks like: 0xb5c8539f9b8d4c675b6dd59d2447049c

Encrypted message:
0x2924fed149a6c30375ac06021e62ab0c9afe0645e87fbacd5d3fa8708bceb5d75e7df847e4eab6d8eda27f3a4c6759d36cf176d7ad01d6d1155fd762c31a130a

Decrypted message: 0xb5c8539f9b8d4c675b6dd59d2447049c

Signature (0xb5c8539f9b8d4c675b6dd59d2447049c ~
0x2254f20c3cd0305848c5036efc44830b57e2be194c40b37fb3042ecdcab2cb8d83f4b8e2011ffc62a3e27ccf2c4c33e4721556c528bb348a4f3798a0b66272a3) generated for message

Message verification done

==========

GENERATED for user B :

p = 0x97f42f9383980bcf7fce66b1f06fde8e6cbe2eb8f50a0070f4260bed712b6327

q = 0x41770d79589d726be75e09821008b6e463c0bf4faed0213760b6d966bed08a37

(e,n,d) for user B looks like this:

(0x232f182c9602be86b0072abb2ae351339625d1081befa37e87ee34b32790b1c52e4f2027b5614ea04874ab252e0b6c982ed1ee94f101b522c419f73907f135f ,
0x26dbaa95ff6a261bc4a7a36273a1b6890942807fc530be482ac379c7811fa2bc9aaa4015641488ec6df0cdc357f5dc2f1393540b40abab95dd94466635755361 ,
0x203c4c7f861a396533806ed28090dd8d0ba7b00a2023fd00d567d34bb608592a6a3d7e1aacccce1f6037c8de6607a52c19cb6d3ec9df2e74256080ed17e1d363)

Message looks like: 0x8314b6263828f52016ccd91ee6ce769c

Encrypted message:
0x19156ec4263b3fc5d3fcd6b7ab820f5c31ac9dcade7152931038b0e1dd63166635add39454ea21de8c3f8b7ddb38762120ac73193ad37f9af201145117673258

Decrypted message: 0x8314b6263828f52016ccd91ee6ce769c

Signature (0x8314b6263828f52016ccd91ee6ce769c ~ 0xf062fafcedf4bcbc82854c83a51ba3a9cde976c1f7ab920e7811c9818c8c30273d705054aa7c2f9806763dde8de1a9a4d2ebc07fb4eb9a6ebfb9dbb0fcb45a5) generated for message

Message verification done

==========

LET`S ORGANISE COMMUNICATION BETWEEN A AND B

REGENERATING USER`S A DATA: n1<n

GENERATED for user A :

p = 0x48721d765221df054edb0e43d6fc037f976afe9291ae88689b854bd78174ce1b

q = 0x3a55e0457f57a2edea0623985d1baf6d885a086173ad0386a944db2847b77c43

(e,n,d) for user A looks like this:

(0x993615149442c7d8b046d026d0a5541f6281c8f869685abc45e09ced7474de6a7876c9e7f2461c6b691ca1a8972558430152a6c253ddfd38f5ee6e9295ae3cd , 0x108228081b77d409475f7abc7e25d3b739a00aad3495deca47c750c25ab4a94083af0dfa97451947a35fbfbc05da903efa8c647479e06aaaf95178fc47b40511 , 0x5cae224f2e2017f6200d654794d8d28d965278ccfcb5d70cf11516e3fccedfdc0d8a4ed2866cae5f2942926cdce78fdd8a4ca30f0f1f76e5aaeaaa643c3b199)

Message looks like: 0x88a459e7919cfee91f9df18787616205

Encrypted message:
0xb821b4429bdfe984f23d96a333c638a457f92bb5a78c20479b1fe5b048848da993b8357acf6a109bdb825d63ec0ce09cd65c2ae4bf170d7095b18f8cc6b56bb

Decrypted message: 0x88a459e7919cfee91f9df18787616205

Signature (0x88a459e7919cfee91f9df18787616205 ~ 0xc36f796de514aa326b91f19798cd438962ed5a19f2e76b84019d9be4f2edde247ddff5c222aee40fe3a92a6f8e584c62b94967cce01584c0e7baae3c0abc11f) generated for message

Message verification done

==========

LET`S ORGANISE COMMUNICATION BETWEEN A AND B

USER A PICKS UP THIS KEY:
0xe1011cf748006f1f71ddef88b7bbe67c4388c00152264a3313e36b4a1ec5d8fdeb00995b0b1a93ad05ac80433907124b860e1b5a938ef8650e13ac30a55be08

SHARED KEY BETWEEN A AND B ESTABLISHED SUCCESSFULLY!!!

0xe1011cf748006f1f71ddef88b7bbe67c4388c00152264a3313e36b4a1ec5d8fdeb00995b0b1a93ad05ac80433907124b860e1b5a938ef8650e13ac30a55be08

Enter public key for server: first e1:

10001

then n1:

0x9AF2A4ADDEA01E38E783837C78CF62E88A0B9012771E4C257FC3F6EC8EAACB3A2E3410E170D46F25EDC4C91B10BEC23321798185D950BA4DF581E64CA7FA4DB7

LET`S ORGANISE COMMUNICATION BETWEEN A AND SERVER

USER A PICKS UP THIS KEY: 0x309 / 777

0x9f6cbf1d6206984a730bf191b2c0bf9d26ee92d1c66c7773f4c225e92d2d51ca4185509b38de3c65b4a45584d034743fbc9e303fcf0cc8047a74d94dadc77f4

Key for server =
0x50106daf549742ed67d7527accdd7c7933ae0f02f776e5c767b62a48d491aefe5113de583b1e033dfc3b3b65cfe82fc3c8149fde48d577b45b78b8951f3213e2

Signature for server =
0x6b244817bbe1c603f8080586cb8ae12ad8ec5abf047d81732fabf5fc5748d1d71866658a73812cc97650990c0a6d6cd4fe05b7335db6819eb11e24b6de520377

User`s A e =
0x161658f77d1cba3617737e2d970b1d502b69affd48bf272fb6823d484743d2de0e11413045c6728a44c2afb8da7cf993651f913da2924c29f9ce38c6e52fd7ed

User`s A n =
0x2b346e2c2db85841847b8a8d66350dc3f3c5f30ef686920d64f780d52652f66ed5ece03b6b037e91c113b24090ac16beb1c2cc652e3327b837655106dd8bd731