ส่วนที่ 2

Task 2: Encrypting a Message ใช้คำสั่ง:

BN_hex2bn(&m, "4120746f7020736563726574657421"); // แปลงข้อความในรูปแบบ Hex เป็นเลขจำนวนใหญ่

BN_hex2bn(&n, "DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");

BN_hex2bn(&e, "010001"); // public key

BN_hex2bn(&d, "74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");

// priveate key

BN_mod_exp(res, m, e, n, ctx); // res = m^e mod n (เข้ารหัส)

printBN("2 en", res); // พิมพ์ข้อความที่เข้ารหัส

ทำการเข้ารหัสข้อความ m ด้วยคีย์สาธารณะ e และโมดูลัส n

BN_mod_exp(res, res, d, n, ctx); // res = res^d mod n (ถอดรหัส)

printBN("2 de", res); // พิมพ์ข้อความที่ถอดรหัส

ถอดรหัสข้อความที่เข้ารหัส (res) ด้วยคีย์ส่วนตัว d และโมดูลัส n

Task 3: Decrypting a Message ใช้คำสั่ง:

```
BIGNUM *c = BN_new();  // ข้อความที่ถูกเข้ารหัส

BN_hex2bn(&c,
"8C0F971DF2F3672B28811407E2DABBE1DA0FEBBBDFC7DCB67396567EA1E2493F"); //
ข้อความที่เข้ารหัส

   BN_mod_exp(c, c, d, n, ctx);         // c^d mod n (ถอดรหัส)

   printBN("3 de", c);                  // พิมพ์ข้อความที่ถอดรหัส
```

ถอดรหัสข้อความที่เข้ารหัส c ด้วยคีย์ส่วนตัว d และโมดูลัส n

Task 4: Signing a Message ใช้คำสั่ง:

BN_hex2bn(&m, "49206F776520796F752024323030302E"); // ข้อความ "I owe you $2000."

BN_mod_exp(res, m, d, n, ctx); // res = m^d mod n (สร้างลายเซ็น)

printBN("4 sign2000 = ", res); // พิมพ์ลายเซ็น

ลายเซ็นถูกสร้างขึ้นโดยใช้คีย์ส่วนตัว d ซึ่งเป็นค่าเฉพาะของผู้ส่ง ทำให้ข้อความที่มีลายเซ็นนี้ยืนยันได้ว่ามาจากเจ้าของคีย์ส่วนตัวตัวจริง

ผลลัพธ์ที่คาดหวัง: "4 sign2000 = [ลายเซ็นที่สร้างขึ้น]"

BN_hex2bn(&m, "49206F776520796F752024333030302E"); // ข้อความ "I owe you $3000."

BN_mod_exp(res, m, d, n, ctx); // res = m^d mod n (สร้างลายเซ็น)

printBN("4 sign3000 = ", res); // พิมพ์ลายเซ็น

ลายเซ็นนี้ถูกสร้างขึ้นโดยใช้คีย์ส่วนตัว d ซึ่งเป็นค่าเฉพาะของผู้ส่ง ทำให้ข้อความที่มีลายเซ็นนี้ยืนยันได้ว่ามาจากเจ้าของคีย์ส่วนตัวตัวจริง

ผลลัพธ์ที่คาดหวัง: "4 sign3000 = [ลายเซ็นที่สร้างขึ้น]"

Task 5: Verifying a Signature ใช้คำสั่ง:

```
BN_hex2bn(&m, "4C61756E63682061206D697373696C652E");
// ข้อความ "Launch a missile."
   BIGNUM *s = BN_new();    // ตัวแปรสำหรับลายเซ็น
   BN_hex2bn(&s,
"643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6802F");
// ลายเซ็นที่ต้องการตรวจสอบ
   BN_hex2bn(&n,
"AE1CD4DC432798D933779FBD46C6E1247F0CF1233595113AA51B450F18116115");
// n ใหม่


BN_mod_exp(res, s, e, n, ctx);                    // res = s^e mod n (ตรวจสอบลายเซ็น)
```

เป็นการถอดรหัสลายเซ็นที่เข้ารหัสไว้โดยใช้คีย์ส่วนตัว (Private Key) ของผู้ส่ง

ผลลัพธ์ res จะได้ค่าที่ควรเป็นข้อความเดิม (m) ถ้าหากลายเซ็นถูกต้อง

```
printBN("5 read signature with 2F as = ", res);        // พิมพ์ผลลัพธ์


if (BN_cmp(res, m) == 0)                                // เปรียบเทียบข้อความกับลายเซ็นที่ถอดรหัส
  {
     printf("Message match with signature\n");       // ถ้าเท่ากับ 0 ลายเซ็นถูกต้อง
  }
```

ใช้เปรียบเทียบข้อความที่ถอดรหัสจากลายเซ็น (res) กับข้อความต้นฉบับ (m)

ถ้าตรงกันลายเซ็นถูกต้ององ แต่ถ้าไม่ตรงกัน ลายเซ็นจะถือว่าไม่ถูกต้อง

ตรวจสอบลายเซ็นอื่น

```
BN_hex2bn(&s,
"643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6803F");
// ลายเซ็นที่เปลี่ยนแปลง
  BN_mod_exp(res, s, e, n, ctx);                    // res = s^e mod n (ตรวจสอบลายเซ็น)
  printBN("5 read signature with 2F as = ", res);   // พิมพ์ผลลัพธ์

  if (BN_cmp(res, m) == 0)                          // เปรียบเทียบข้อความกับลายเซ็นที่ถอดรหัส
  {
    printf("Message match with signature\n");       // ถ้าเท่ากับ 0 ลายเซ็นถูกต้อง
  }
  else
  {
    printf("Message does not match with signature\n");   // ถ้าไม่ใช่ 0 ลายเซ็นไม่ถูกต้อง
  }
```

Task 2: Encrypting a Message

ผลลัพธ์:

- ข้อความ m ถูกเข้ารหัสด้วยคีย์สาธารณะ e และโมดูลัส n จนได้ผลลัพธ์ res ซึ่งเป็น ciphertext
- ข้อความที่เข้ารหัสไม่สามารถเข้าใจได้ในรูปแบบ ASCII แต่ถอดรหัสกลับมาได้โดยใช้คีย์ส่วนตัว d

สิ่งที่สังเกตได้:

- การเข้ารหัสทำงานได้ถูกต้อง โดยค่าที่เข้ารหัส (res) เป็นข้อมูลที่ถูกเปลี่ยนแปลงไปจากข้อความต้นฉบับ (m) เพื่อป้องกันการเข้าถึงโดยบุคคลอื่น

คำตอบ: RSA ใช้กระบวนการนี้ในการป้องกันข้อความสำคัญที่ถูกส่งจากผู้ส่งไปยังผู้รับ โดยผู้ส่งใช้คีย์สาธารณะเพื่อเข้ารหัสข้อความ และผู้รับใช้คีย์ส่วนตัวเพื่อถอดรหัสข้อความ

Task 3: Decrypting a Message

ผลลัพธ์:

- Ciphertext c ถูกถอดรหัสกลับเป็นข้อความต้นฉบับ (m) โดยใช้คีย์ส่วนตัว d และโมดูลัส n(ไม่รู้ว่าต้องมีป่าว)
- ข้อความที่ถอดรหัสได้ตรงกับข้อความต้นฉบับ (m) ที่เข้ารหัสใน Task 2

สิ่งที่สังเกตได้:

- ใช้คีย์ส่วนตัวในการถอดรหัสเท่านั้น ทำให้มั่นใจได้ว่ามีเพียงผู้รับที่ถือคีย์ส่วนตัวจึงสามารถเข้าถึงข้อความได้

คำตอบ: กระบวนการนี้ยืนยันถึงความปลอดภัยในระบบการเข้ารหัส RSA เนื่องจากต้องใช้คีย์ส่วนตัวที่เป็นค่าเฉพาะในการถอดรหัสเท่านั้น

Task 4: Signing a Message

ผลลัพธ์:

- ลายเซ็นดิจิทัล (res) ถูกสร้างขึ้นโดยใช้คีย์ส่วนตัว d และข้อความต้นฉบับ m
- ผลลัพธ์ที่ได้เป็นข้อมูลที่สามารถนำไปตรวจสอบเพื่อยืนยันตัวตนของผู้ส่ง

สิ่งที่สังเกตได้:

- ลายเซ็นดิจิทัลมีความเฉพาะเจาะจง เนื่องจากการสร้างลายเซ็นใช้คีย์ส่วนตัวของผู้ส่ง ดังนั้นจึงไม่สามารถปลอมแปลงได้ง่าย

คำตอบ: การเซ็นข้อความด้วย RSA ช่วยรับประกันความน่าเชื่อถือของผู้ส่ง เพราะลายเซ็นยืนยันว่าข้อความมาจากเจ้าของคีย์ส่วนตัว และไม่ได้ถูกแก้ไข

Task 5: Verifying a Signature

ผลลัพธ์:

- ลายเซ็น (s) ถูกตรวจสอบโดยใช้คีย์สาธารณะ (e) และโมดูลัส (n) หากผลลัพธ์จากการถอดรหัสตรงกับข้อความต้นฉบับ (m) จึงยืนยันว่าลายเซ็นนั้นถูกต้อง
- ในกรณีที่ลายเซ็นถูกแก้ไข (643D6F...03F) ผลลัพธ์ของการตรวจสอบจะไม่ตรงกัน และลายเซ็นจะถือว่าไม่ถูกต้อง

สิ่งที่สังเกตได้:

- RSA มีความสามารถในการตรวจสอบลายเซ็นว่าถูกต้องหรือไม่ ด้วยการใช้คีย์สาธารณะ(e)และโมดูลัส(n)
- ลายเซ็นที่ไม่ตรงกับข้อความต้นฉบับจะถือว่าไม่สามารถรับรองได้

คำตอบ: การตรวจสอบลายเซ็นดิจิทัลด้วย RSA เป็นขั้นตอนสำคัญในระบบความปลอดภัย เช่น การส่งเอกสารหรือข้อความที่ต้องยืนยันตัวตนผู้ส่ง เพื่อป้องกันการแก้ไขข้อมูลระหว่างทาง

สรุป

อัลกอริทึม RSA มีบทบาทสำคัญในการเข้ารหัสและตรวจสอบลายเซ็นดิจิทัล โดย:

1. ใช้คีย์คู่ (Public Key และ Private Key) เพื่อสร้างความปลอดภัยให้กับข้อมูล
2. การเข้ารหัสทำให้ข้อความเป็นข้อมูลที่ยากต่อการอ่าน
3. การเซ็นข้อความช่วยยืนยันตัวตนผู้ส่งและป้องกันการปลอมแปลงข้อมูล