Task 1: Deriving the Private Key

เราสามารถคำนวณค่า private key ดังนี้ และนำไปใช้ในชุดคำสั่งภาษา C ได้

Choose two large prime numbers (p and q)

Calculate n = p*q and z = (p-1)(q-1)

Choose a number e where 1 < e < z

Calculate d = e-1mod(p-1)(q-1)

1.1 ชุดคำสั่งภาษา C

```
#include <stdio.h>

#include <openssl/bn.h>

#define NBITS 256

void printBN(char *msg, BIGNUM *a){

// ฟังก์ชั่นไว้รับชื่อตัวแปร และ BIGNUMที่ได้ สำหรับปริ้นท์

char * number_str = BN_bn2hex(a);

// แปลง BIGNUM ที่ส่งเข้ามาให้เป็น String

printf("%s %s\n", msg, number_str); //คำสั่งปริ้นท์ออกมาทาง Terminal

OPENSSL_free(number_str); // คืนMemmory

}

int main(){

// ประกาศตัวแปรไว้ใช้ใน Main ดังนี้

BN_CTX *ctx = BN_CTX_new();

BIGNUM *p = BN_new();

BIGNUM *q = BN_new();
```

```
BIGNUM *e = BN_new();

BIGNUM *d = BN_new();

BIGNUM *r1 = BN_new();

BIGNUM *r2 = BN_new();

BIGNUM *r3 = BN_new();

BIGNUM *one = BN_new();

// แปลง hexadecimal ที่ให้มา เป็น BIGNUM และเก็บไว้ที่ p

BN_hex2bn(&p, "F7E75FDC469067FFDC4E847C51F452DF");


// แปลง hexadecimal ที่ให้มา เป็น BIGNUM และเก็บไว้ที่ q

BN_hex2bn(&q, "E85CED54AF57E53E092113E62F436F4F");

// แปลง hexadecimal ที่ให้มา เป็น BIGNUM และเก็บไว้ที่ e (public key)

BN_hex2bn(&e, "0D88C3");

// เก็บ 1 ไว้ใน ตัวแปรBIGNUM ชื่อ one ไว้สำหรับคำนวณ (p-1)(q-1)

BN_dec2bn(&one,"1");


//คำนวณ ค่า p-1 และเก็บผลลัพธ์ไว้ที่ r1

BN_sub(r1, p, one);


//คำนวณ ค่า q-1 และเก็บผลลัพธ์ไว้ที่ r2

BN_sub(r2, q, one);
```

```
//คำนวณ ค่า r1*r2 และเก็บผลลัพธ์ไว้ที่ r3(totient(n))

BN_mul(r3, r1, r2, ctx);



// d = e-1mod(r3)

BN_mod_inverse(d, e, r3, ctx);

//print BN ส่ง ข้อความและ ค่าของ d ไปยังฟังก์ชั่น print

printBN("d = ",d);

return 0;

}
```

1.2 ผลลัพธ์ของการรันชุดคำสั่ง

หลังจาก Compile ชุดคำสั่ง และ รัน ./task-1 ได้ผลลัพธ์ดังนี้

```
rew@LAPTOP-F65IQ8R2:~/Lab1$ ./task-1
d =  3587A24598E5F2A21DB007D89D18CC50ABA5075BA19A33890FE7C28A9B496AEB
```

Private key d =

3587A24598E5F2A21DB007D89D18CC50ABA5075BA19A33890FE7C28A9B496AEB