Network Security

Project 1 - Hacking the Cipher

Tools

Language: C/C++

Used Library : OpenSSL

Program flow (P.S. 咖啡色字為使用到的 functions)

1. 從.pub 檔讀入各個 Public Keys

```
RSA *PEM_read_RSA_PUBKEY(FILE *fp, RSA **x,
                         pem_password_cb *cb, void *u)
```

將檔案中各個 Public keys 讀入，並且存成 RSA 這個 Data structure，透過 RSA->n 即可取得 BIGNUM 形態的 n value .

2. 找尋兩個不互值的 n，並求出其 GCD

```
int BN_gcd(BIGNUM *r, BIGNUM *a, BIGNUM *b, BN_CTX *ctx)
```

取 12 個 Public Key 的 n，兩倆互相做 GCD .

3. 令他們的 GCD 為 p，利用 n / p 求出 q

```
int BN_div(BIGNUM *dv, BIGNUM *rem, const BIGNUM *a,
                          const BIGNUM *d, BN_CTX *ctx)
```

4. 求出各別的 Φ(n)

```
int BN_sub(BIGNUM *r, const BIGNUM *a, const BIGNUM *b)
```

```
int BN_mul(BIGNUM *r, BIGNUM *a, BIGNUM *b, BN_CTX *ctx)
```

利用 $\Phi(n) = (q - 1)(p - 1)$這個公式求出小於 n 與 n 互值的數的個數 .

5. 有了 Φ(n), Publics Key, 即可求出 Private Key

```
BIGNUM *BN_mod_inverse(BIGNUM *r, BIGNUM *a,
                       const BIGNUM *n, BN_CTX *ctx)
```

Private Key 為 Public Key 在模 Φ(n)下的乘法反元素，因此有了 Public Key 與 Φ(n)，即可推出 Private Key .

6. 將 Private Key 輸出成 PEM format (.pem 檔)

```
int PEM_write_RSAPrivateKey(FILE *fp, RSA *x,
        const EVP_CIPHER *enc, unsigned char *kstr,
                  int klen, pem_password_cb *cb, void *u)
```

Result

```
Ricky  publicKeys  ./main
Target Public Key Number : 3, 8

Common Factor :
D0B7C45918753A4C0A819AA2E77797124C24ACFF9A0F093F4A86641CFB73CC63BF564988ED9A86B0E6F716616C1042A7277EB4F81BA079526609AE74CBBB1C900B8D29E2A0B568E5881516704D6EF5
AD98DD23A86E47C3B727CC5BD078E022EF2F656CC240B8E8EB1F6602F52C74D880550A3AC6F2F444FAA5759686B0AD7C69

Public Key 3's q :
C18E34C5788F13D16F45946EF99689A3EC54361F3E8FC0AEFA3BA110A61BCF46DDD67C93FFE550FAEC58D78377AF75D15C603F71DCCFE5F4B934FF59FB708E5E7D63234C2FA486C384B528D105CEBE
97610262E059F182CDD0DED802B03709264DBD3F4F68E0F1276DBAF8139BD3FBF4F8FF9A7337C7B9D80B0260FD2FB8B84D
Public Key 8's q :
D87CCB4BC31EDE5EE7EA65480DE4CE42C6828D281FEC8126C517A922E28BF00E2E5BF7F0662D95AC69AA8C06DC7C7D1B6D0EDB2E5BD8E8D8DB939D55EC28A0AB6FBF352481E32F17C14F915E89A022
D17872C45FB94B93BE6F333320E2DE56A9332D48C93157A36163F7BD4FAC4CC133A880F9A523F4A298BE633657042B8B6D

Private Key 3 :
C47DD8200578A3512292B0069A1875140EDEEC55C7040674FA35EF5EB80CF87E590D1185903B5F38A5666980210D63DBC9425632E28493EBB503A89CE080DE5CF040819BDCC802BB733572F8F8E992
B27DEAC2E5E133E80657E0E485077C5465B09DEAE2D4B455C388C0A0DEBBBBBB4BA8B208A041294D22E63BA2C063F59F72B08F13754E12625B87D7F7E1E88ED5B5236C0F59D89A2E4193D982B8F333
28B117F71D0912ED17BA665B1D6FB8B37E53A79D622D533DEB386321DBC2A42F118384AC3D83A9965C303C6B357D99EC2BE249AFD715CC2ECCCE3E45832BCBD1FA1A162E56F130F22AECDD77D57B7C
EAC84298F5693310DD862C4A5764B39A659BD
Private Key 8 :
AA1DC41706D5B7D5555D68BB75FFDA71FCCD064E87B0C4C3512373523AC58C382C554E9B184101CCD50C169DCAE11501B8AEF342C720FD3A39DED2CDA2A5DA325FF45D7DB47C723056B824ACD1CAF2
88B32A96C9F0705E6DB7CDDAF452BF47DD97FA4DC60B40630297B9999C68ADF1112BEC1043EC450988AD334AB98BBB76B35E3808937FE7A12BF7A4DC9FD8B76FC365ECB238DCDC0BF15DA3D65F54CD
7CBA3797D2CA7B8F257AFAC79BD4332B9668B82A9102D61F6584BD1ECFD4BC408D73B1BF8D3FECF2A76BAD13C221296260115879649BE0042EB3061F81D6F0E0E6EADD225CC417712D867465AE488B
1A84C47700C21860B50D16FFC00DFBEFE0195D
```

Reference

- **Wiki OpenSSL – Manual: Bn**
  https://wiki.openssl.org/index.php/Manual:Bn(3)
- **Wiki OpenSSL – Manual: Rsa**
  https://wiki.openssl.org/index.php/Manual:Rsa(3)
- **Wiki OpenSSL – Manual: Pem**
  https://wiki.openssl.org/index.php/Manual:Pem(3)
- **Understanding Common Factor Attacks: An RSA-Cracking Puzzle**
  http://www.loyalty.org/~schoen/rsa/