

Tên: Nguyễn Việt Thanh Duy

MSSV: 19127378

Báo cáo phương pháp tấn công RSA

Thuật toán phân tích Fermat

1. Tài liệu tham khảo:

Phương pháp được nêu trong bài báo khoa học: Performance Analysis of Fermat Factorization Algorithms ([link](#)).

2. Cơ sở toán học:

- Trong thực tế, ta cần chọn p, q có cùng độ dài bit để tạo được 1 mã RSA mạnh, tuy nhiên nếu p, q quá gần nhau thì lại tạo ra lỗ hổng bảo mật khi mà attacker có thể dễ dàng factorize n .
- Điều kiện sử dụng thuật toán: $p - q < \sqrt[4]{n}$
- Thuật toán dựa trên hằng đẳng thức: $x^2 - y^2 = (x - y)(x + y)$
- Ta có:

$$n = pq = \left(\frac{q-p}{2}\right)^2 - \left(\frac{p+q}{2}\right)^2 = x^2 - y^2$$

- Vậy n có thể được phân tích thừa số nguyên tố như sau: $n = (x-y)(x+y)$.
- Như vậy nếu ta có thể tìm được (x, y) ta có thể suy ra dc (p, q) :

$$p = x + y$$

$$q = x - y$$

- Thuật toán phân tích Fermat có thể giúp ta làm được điều đó, mã giả:

Algorithm FF (Fermat's Factorization)

Input: n is a positive odd number.

Output: p and q are two prime numbers such that $n = p q$.

Begin

1. $x = \lfloor \sqrt{n} \rfloor + 1$

2. $y = x^2 - n$

3. While (y is not a perfect square) do

4. $x = x + 1$

5. $y = x^2 - n$

6. End while

7. $p = x + \sqrt{y}$

8. $q = x - \sqrt{y}$

End.

3. Thực nghiệm mô phỏng tấn công:

❖ Chương trình mô phỏng(được viết bằng ngôn ngữ Python) bao gồm các phần:

- Xây dựng mã RSA với $|p-q|$ nhỏ để dễ dàng mô phỏng tấn công.
- Tấn công RSA bằng phương pháp phân tích Fermat.

❖ Kết quả một số lần chạy thử chương trình:

- TN1: p, q là số nguyên tố 32 bit

```
PS C:\Users\BIN\Desktop\Cryp\lab05> python -u "c:\Users\BIN\Desktop\Cryp\Bonus\Source code.py"
Cho m = 19127378
Ma hoa m bang RSA:
e = 4737242479
d = 3145199619658514847
n = 20897929801601982211
c = 3724650044171855717
Tan cong RSA:
p = 4698568169
q = 4447723019
d = 3145199619658514847
m = 19127378
```

- TN2: p, q là số nguyên tố 64 bit

```
PS C:\Users\BIN\Desktop\Cryp\lab05> python -u "c:\Users\BIN\Desktop\Cryp\Bonus\Source code.py"
Cho m = 19127378
Ma hoa m bang RSA:
e = 25676790046854447517
d = 445107169363655999446040860617099009853
n = 487240947911532641355492487412599434791
c = 358084613914687278860576588993871821247
Tan cong RSA:
p = 22073535011672521901
q = 22073535011672521891
d = 445107169363655999446040860617099009853
m = 19127378
```

- TN3: p, q là số nguyên tố 128 bit

```
PS C:\Users\BIN\Desktop\Cryp\lab05> python -u "c:\Users\BIN\Desktop\Cryp\Bonus\Source code.py"
Cho m = 19127378
Ma hoa m bang RSA:
e = 410504062136562850857454983246611281241
d = 241706870502913751075473766403450816809295491099382221314344223024997982708041
n = 412158815553765759389738983313142703390604543775354629471411301070329777635407
c = 351290535404392572590607220452187358258294341112436628466968805660656350107179
Tan cong RSA:
p = 641995962256590644509303178813018068397
q = 641995962256590644509303178813018068331
d = 241706870502913751075473766403450816809295491099382221314344223024997982708041
m = 19127378
```

- TN4: p, q là số nguyên tố 512 bit

```

PS C:\Users\ADMIN\Desktop\crypt\lab05> python -c "from Crypto.PublicKey import RSA; p = 19127378; q = 19127378; n = p*q; e = 65537; d = 2252140629593000053201410101937070305124080506077988400170540000991162344747639939548066600127940001291444866111385995736293884005210129960380099; k = 2638225448386999010295570634154401542629541671355442116383240409404065413450654329123100547613383511675782538338057559077238086751195395212551206644731030900772221553047518018015690673053099977982398889504779117183963012385660884848501801178116608809488951337593012613484710651760257302578415599; m = 571683552955905358674617468380167219426255509579867465722081215096754177191490075138889928884847748891026171842554626756867365011091413007009185152658415674161134149507289958135359000772833566438867203881272306113754475412298618505270884421180721215361181154667272804861983; c = 92269729306561052770000227314261642483796340183123692308114224282708974896081418243662996367152587738081304287983807773501726487768825497248277757679638834107114464763827481714712674240212537187640516520547405812662941594656038613157150778853606524465615421562977837867277193485; tin cong RSA; p = 1998248122818051622050257992357181491436136340896778004111614621259996412129866612053953437504858297199819041127070483584211425631156655521313175825201; q = 19981248122818051622050257992357181491436136340896778004111614621259996412129866612053953437504858297199819041127070483584211425631156655521313175825213; e = 2638225448386999010295570634154401542629541671355442116383240409404065413450654329123100547613383511675782538338057559077238086751195395212551206644731030900772221553047518018015690673053099977982398889504779117183963012385660884848501801178116608809488951337593012613484710651760257302578415599; m = 19127378"

```

- TN5: p,q là số nguyên tố 1024 bit

```

PS C:\Users\ADMIN> python -c "from Crypto.PublicKey import RSA; p = 19127378; q = 19127378; n = p*q; e = 65537; d = 2252140629593000053201410101937070305124080506077988400170540000991162344747639939548066600127940001291444866111385995736293884005210129960380099; k = 2638225448386999010295570634154401542629541671355442116383240409404065413450654329123100547613383511675782538338057559077238086751195395212551206644731030900772221553047518018015690673053099977982398889504779117183963012385660884848501801178116608809488951337593012613484710651760257302578415599; m = 571683552955905358674617468380167219426255509579867465722081215096754177191490075138889928884847748891026171842554626756867365011091413007009185152658415674161134149507289958135359000772833566438867203881272306113754475412298618505270884421180721215361181154667272804861983; c = 92269729306561052770000227314261642483796340183123692308114224282708974896081418243662996367152587738081304287983807773501726487768825497248277757679638834107114464763827481714712674240212537187640516520547405812662941594656038613157150778853606524465615421562977837867277193485; tin cong RSA; p = 1998248122818051622050257992357181491436136340896778004111614621259996412129866612053953437504858297199819041127070483584211425631156655521313175825201; q = 19981248122818051622050257992357181491436136340896778004111614621259996412129866612053953437504858297199819041127070483584211425631156655521313175825213; e = 2638225448386999010295570634154401542629541671355442116383240409404065413450654329123100547613383511675782538338057559077238086751195395212551206644731030900772221553047518018015690673053099977982398889504779117183963012385660884848501801178116608809488951337593012613484710651760257302578415599; m = 19127378"

```

❖ Kết luận:

- Chỉ cần $|p - q|$ đủ nhỏ thì thuật toán chạy rất tốt, kể cả với số lớn.
- Để tránh cho RSA không bị tấn công phân tích Fermat thì trước khi sử dụng 2 hai số nguyên tố p và q (được sinh ngẫu nhiên), ta nên kiểm tra $|p - q|$ không được quá nhỏ.