3. Public-key Cryptography - RSA & Diffie-Hellman K.E

3.1 TỔNG QUAN

Phương pháp mã hóa đối xứng dù đã phát triển từ cổ điển đến hiện đại song vẫn tồn tại 2 điểm yếu:

- Vấn đề trao đổi khóa bí mật giữa người gửi và người nhận: Cần phải có một kênh an toàn để trao đổi khóa sao cho khóa phải được giữ bí mật và chỉ có người gửi và người nhận biết. Vấn đề này sẽ khó khả thi, việc thiết lập một kênh an toàn như vậy sẽ tốn kém về mặt chi phí và chậm trễ về thời gian.
- Tính bí mật của khóa: không có cơ sở quy trách nhiệm nếu khóa bị tiết lộ.

Vào năm 1976, *Whitfield Diffie* và *Martin Hellman* đã tìm ra một phương pháp mã hóa khác có thể giải quyết được hai vấn đề trên, đó là mã hóa khóa công khai *(public key cryptography)* hay còn gọi là mã hóa bất đối xứng *(asymetric cryptography)*. Đây có thể xem là một bước đột phá quan trọng nhất trong lĩnh vực mã hóa.

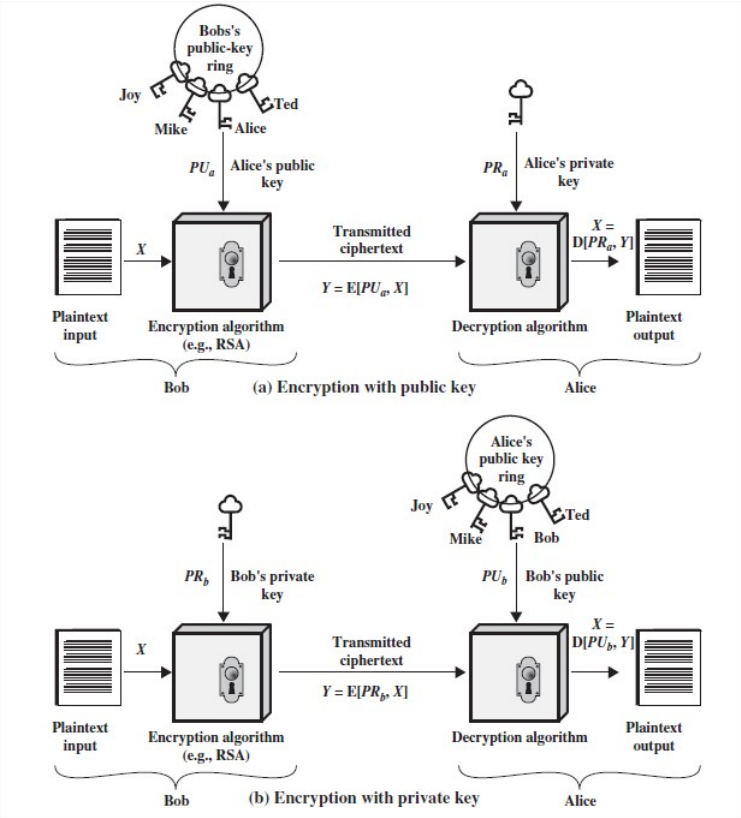
Để khắc phục điểm yếu của mã hóa đối xứng người ta tập trung vào nghiên cứu theo hướng: *có phương pháp nào để việc mã hóa và giải mã dùng hai khóa khác nhau?* Có nghĩa là **C = E(P, K1) và P = D(C, K2)**. Như vậy, chúng ta sẽ có 2 phương án áp dụng ứng với 2 mục tiêu:

1. **Tính bảo mật:** Người nhận (Bob) giữ bí mật khóa K2, còn khóa K1 thì công khai cho tất cả mọi người biết. Alice muốn gởi dữ liệu cho Bob thì dùng khóa K1 để mã hóa. Bob dùng K2 để giải mã. Điều này bảo đảm tính bảo mật của quá trình truyền dữ liệu.
2. **Tính chứng thực & không từ chối:** người gửi (Alice) giữ bí mật khóa K1, còn khóa K2 thì công khai cho tất cả mọi người biết. Alice muốn gởi dữ liệu cho Bob thì dùng khóa K1 để mã hóa. Bob dùng K2 để giải mã. Do đó phương án này không đảm bảo tính bảo mật. Tuy nhiên lại có tính chất quan trọng là đảm bảo tính chứng thực và tính không từ chối. Vì chỉ có duy nhất Alice biết được khóa K1, nên nếu Bob dùng K2 để giải mã ra bản tin, thì điều đó có nghĩa là Alice là người gửi bản mã.

Để tránh nhầm lẫn với khóa bí mật của mã đối xứng, khóa bí mật trong mô hình trên gọi là khóa riêng *(private key)* và thường ký hiệu là K_R, khóa công khai *(public key)* thường ký hiệu là K_U.

- Bản rõ được ký hiệu là M, bản mã hóa ký hiệu là C
- Phương án 1 viết lại thành: **C = E(M, K_U) và M = D(C, K_R)**
- Phương án 2 viết lại thành: **C = E(M, K_R) và M = D(C, K_U)**

Phải có một mối quan hệ nào đó giữa K_U và K_R mới có thể tiến hành giải mã hóa và giải mã được, nghĩa là $K_R = f(K_U)$. Tuy nhiên một yêu cầu rất quan trọng là việc tính $K_R = f(K_U)$ phải là bất khả thi về mặt thời gian. Nếu nguyên tắc này bị vi phạm thì việc giữ bí mật khóa K_R không còn ý nghĩa vì từ khóa công khai K_U có thể tính được K_R.



Hình 3.1: Ví dụ về 2 loại mã hóa với RSA

Có nhiều phương pháp mã hóa thuộc loại mã hóa khóa công khai như *Knapsack, RSA, Elgaman, phương pháp đường cong elliptic ECC,....*
Bài thực hành này sẽ tập trung vào việc tìm hiểu; thực hành về phương pháp mã hóa khóa công khai tiêu biểu - RSA và phương pháp trao đổi khóa Diffie-Hellman.

3.1.1 Kiến thức nền tảng

Để tìm hiểu và thực hành tốt, sinh viên cần nắm được kiến thức về:
- Lý thuyết số:
  + Phép chia modulo (chia lấy phần dư)
  + Ước số, ước số chung lớn nhất *gcd(a,b)*
  + Số nguyên tố, số nguyên tố cùng nhau.
  + Phép logarit rời rạc *(xác định x khi biết y,a và n trong phép lũy thừa modulo $y = a^x mod n$)*
- Phương pháp mã hóa RSA
- Phương pháp trao đổi khóa Diffie-Hellman

Những nội dung trên, sinh viên có thể tham khảo tại:
Chapter 8: More Number Theory & Chapter 9: Public-Key Cryptography and RSA
W. Stallings, *Cryptography and network security: Principles and practice, 6th ed.* Boston, MA, United States: Prentice Hall, 2013

3.1.2 Môi trường - Công cụ

1. Máy tính Windows và IDE với ngôn ngữ lập trình tùy ý (C#, Java, Web,Python,...)
2. Phần mềm CrypTool 2.1

3.2 NỘI DUNG THỰC HÀNH

Lab 3 gồm 1 buổi thực hành với thời lượng 5 tiết/buổi theo kế hoạch 6 buổi thực hành.

3.2.1 Lý thuyết số

Trước khi bắt đầu với thuật toán mã hóa khóa công khai, chúng ta sẽ ôn tập và thực hành một số yêu cầu về lý thuyết số như xác định số nguyên tố, ước chung lớn nhất, modulo và số e.

Task 3.1 Viết chương trình tương ứng để thực hiện các yêu cầu sau:

1. Số nguyên tố:
   + Phát sinh số nguyên tố ngẫu nhiên có độ dài 8 bit, 32 bit, 64 bit
   + Xác định 10 số nguyên tố lớn nhất nhỏ hơn 10 số nguyên tố **Mersenne** đầu tiên.
   + Kiểm tra một số nguyên bất kỳ nhỏ hơn 2^{89}-1 có phải số nguyên tố không.
2. Xác định Ước chung lớn nhất (gcd) của 2 số lớn nhất sinh viên có thể xử lý.
3. Tính giá trị của phép lũy thừa modulo $a^x \, mod \, p$ có thể áp dụng cho các trường hợp lũy thừa lớn (>40), ví dụ $7^{40} mod \, 19$

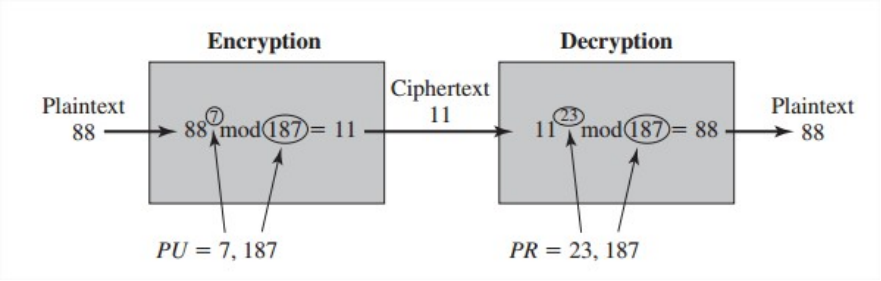
■

Gợi ý 3.1.
- Về việc kiểm tra số nguyên tố lớn, sinh viên có thể tìm hiểu và tham khảo thuật toán Miller-Rabin.
- Việc xác định ước số chung lớn nhất có thể sử dụng thuật toán Euclid.
- Trong phép tính lũy thừa modulo có tính chất bình phương liên tiếp:
$(a \times b) \equiv [(a mod \, n) \times (b mod \, n)] mod \, n$
Có thể áp dụng tính chất trên vào việc xác định giá trị của phép lũy thừa modulo.

3.2.2 Mã hóa RSA

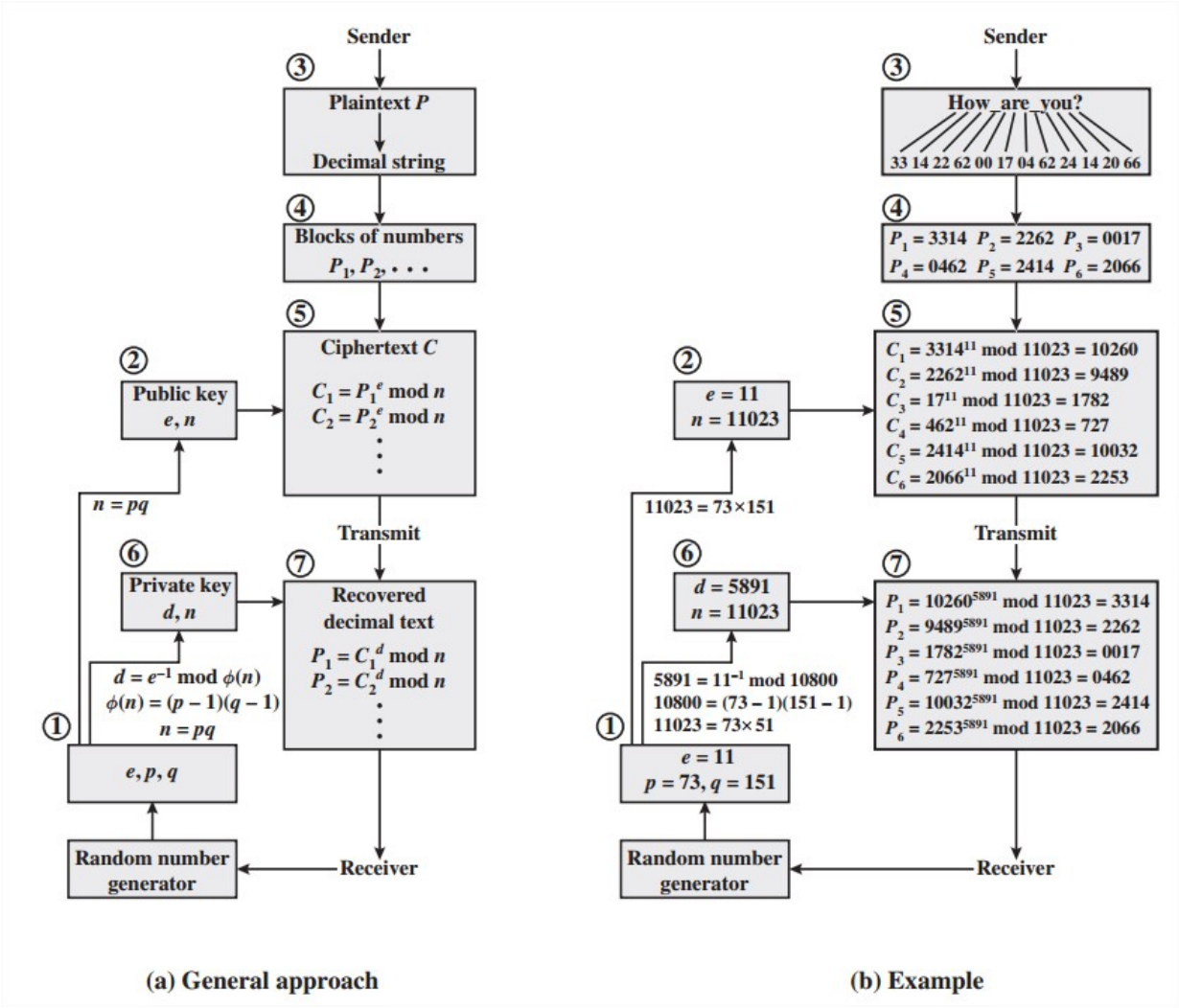
Phương pháp RSA là một phương pháp mã hóa khóa công khai. RSA được xây dựng bởi các tác giả Ron Rivest, Adi Shamir và Len Adleman tại học viện MIT vào năm 1977 và ngày nay đang được sử dụng rộng rãi. Về mặt tổng quát RSA là một phương pháp mã hóa theo khối.
Để thực hiện mã hóa và giải mã, RSA dùng phép lũy thừa modulo của lý thuyết số. Các bước thực hiện như sau:

1. Chọn hai số nguyên tố lớn p và q và tính N = pq, $p \# q$
2. Tính $n = (p-1)(q-1)$
3. Tìm một số e nguyên tố cùng nhau với n
4. Tìm một số d sao cho $e.d \equiv 1 \, mod \, n$ *(d là nghịch đảo của e trong phép modulo n)*
5. Hủy bỏ n, p và q. Chọn khóa công khai K_U là cặp (e, N), khóa riêng K_R là cặp (d, N)
6. Việc mã hóa thực hiện theo công thức:
   + Theo phương án 1, mã hóa bảo mật: $C = E(M, K_U) = M^e mod \, N$
   + Theo phương án 2, mã hóa chứng thực: $C = E(M, K_R) = M^d mod \, N$
7. Việc giải mã thực hiện theo công thức:
   + Theo phương án 1, mã hóa bảo mật: $M = D(C, K_R) = C^d mod \, N$
   + Theo phương án 2, mã hóa chứng thực:$M = D(C, K_U) = C^e mod \, N$



Hình 3.2: Ví dụ minh họa mã hóa, giải mã với RSA

Trong trường hợp cần mã hóa văn bản với nhiều ký tự, cần quy ước và chia thành các khối để tiến hành mã hóa với RSA.



Hình 3.3: Ví dụ về mã hóa khối văn bản với RSA

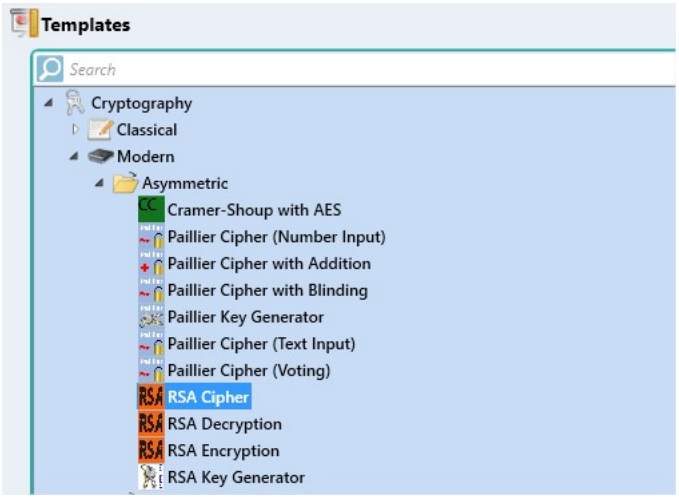
Task 3.2 Để làm quen với RSA, sinh viên sử dụng CrypTool 2 để thực hiện các phép mã hóa với RSA, mô tả nguyên tắc hoạt động và kiểm tra với dữ liệu mẫu của các quá trình:

1. Xác định khóa công khai P_U và khóa riêng P_R khi

+ $p_1 = 11, q_1 = 17, e_1 = 7$
+ $p_2 = 2007999387284232211615121 9, q_2 = 6767171457517362421707 89, e_2 = 17$
(Kiểm tra các giá trị trên là số nguyên tố hay không trước khi tính khóa)

2. Sử dụng khóa trong trường hợp p_1, q_1, e_1 trên để thực hiện mã hóa và giải mã thông điệp M=5 trong 2 trường hợp *mã hóa bảo mật* và *mã hóa chứng thực*.
3. Sử dụng lần lượt các khóa trên để thực hiện mã hóa thông điệp sau:
The University of Information Technology
Xác định Ciphertext tương ứng dưới dạng Base64.
4. Xác định Plaintext tương ứng của các Ciphertext sau, biết rằng chúng được mã hóa bởi 1 trong 2 bộ khóa trên.
   (a) raUcesUlOkx/8ZhgodMoo0Uu18sC20yXlQFevSu7W/
   FDxIy0YRHMyXcHdD9PBvIT2aUft5fCQEGomiVVPv4I
   (b) C8 7F 57 0F C4 F6 99 CE C2 40 20 C6 F5 42 21 AB AB 2C E0 C3
   (c) Z2BUSkJcg0w4XEpgm0JcMExEQmBlVH6dYEp
   NTHpMHptMQ7NgTHlgQrNMQ2BKTQ==
   (d) 00101000 00010100 11111111 10110111 00101110 11001010
   11101100 01100111 10111111 00111111 01101000 11001111 00110000
   10010100 01010100 11110101 01001100 11101110 11101111 01011011 00000100

Gợi ý 3.2. *Sinh viên có thể sử dụng các mô hình RSA được cung cấp hoặc tự xây dựng mô hình mã hóa tại CrypTool 2*



Hình 3.4: Các template về RSA trong CrypTool 2

Mở rộng 3.1 Tự xây dựng ứng dụng để thực hiện mã hóa RSA đáp ứng các yêu cầu cơ bản như:

- Sinh khóa với input là các giá trị p,q,e hoặc sinh khóa ngẫu nhiên là một số nguyên lớn.
- Sử dụng các khóa tương ứng để mã hóa/giải mã thông điệp. Thông điệp có thể là giá trị số hoặc chuỗi ký tự.

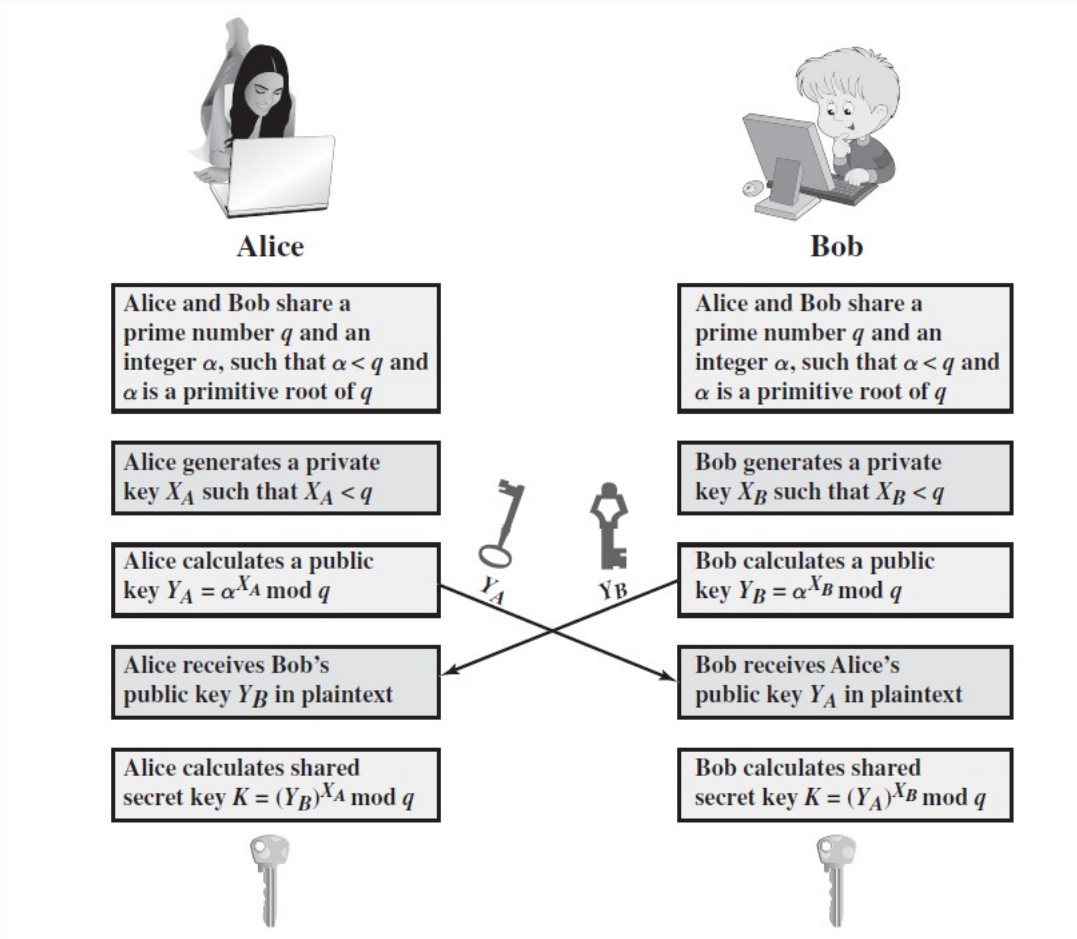
Kiểm tra ứng dụng với một số dữ liệu mẫu.

Gợi ý 3.3. *Cần giải quyết một số vấn đề khi xây dựng ứng dụng như: Phát sinh/ kiểm tra số nguyên tố lớn(có thể sử dụng thuật toán Miller-Rabin); Tìm giá trị ước chung lớn nhất (có thể sử dụng thuật toán Euclid); thuật toán bình phương liên tiếp để tính các giá trị a^x modn lớn,...*

Mở rộng 3.2 Tìm hiểu về mã hóa RSA và các gói hỗ trợ trong các ngôn ngữ lập trình *(nếu có)* để xây dựng chương trình mã hóa/giải mã file đơn giản trên máy tính. Khóa công khai và khóa riêng có thể được sinh ra ngẫu nhiên trong chương trình hoặc được nhập vào từ file khác.

3.2.3 Phương pháp trao đổi khóa Diffie-Hellman

Phương pháp trao đổi khóa Diffie-Hellman dùng để thiết lập một khóa bí mật giữa bên gửi và bên nhận mà không cần sử dụng mã hóa công khai (như RSA). Phương pháp này sử dụng hàm 1 chiều làm hàm logarit rời rạc và không có ý nghĩa về mặt mã hóa như RSA.



Hình 3.5: Quá trình trao đổi khóa Diffie Hellman

Quá trình trao đổi khóa Diffie-Hellman có thể được mô tả như sau:

1. Trước tiên Alice và Bob sẽ thống nhất sử dụng chung một số nguyên tố p và một số g nhỏ hơn p và là *primitive root* của p (nghĩa là phép toán $g^x mod\ p$ khả nghịch). Hai số p và g không cần giữ bí mật.
2. Sau đó Alice chọn một số a và giữ bí mật. Bob cũng chọn một số b và giữ bí mật số b.
3. Tiếp theo Alice tính và gửi $g^a mod p$ cho Bob, Bob tính và gửi $g^b mod p$ cho Alice.
4. Trên cơ sở đó Alice tính:
   $(g^b)^a mod\ p = g^{ab} mod\ p$
   Bob tính:
   $(g^a)^b mod\ p = g^{ab} mod\ p$
   Do đó Alice và Bob có chung giá trị $g^{ab} mod\ p$. Giá trị này có thể dùng làm khóa cho phép

mã hóa đối xứng.

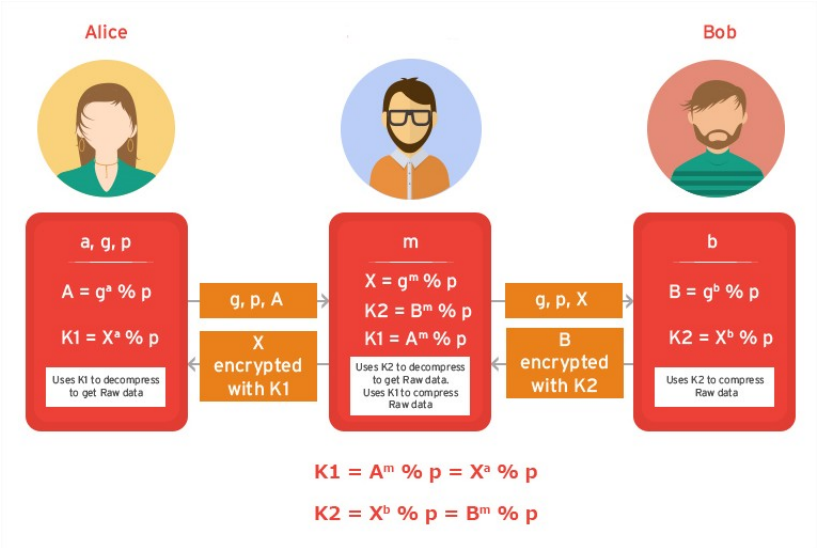
Task 3.3 Xây dựng ứng dụng **"Private Chat"** theo dạng Client-Server sử dụng giao thức Diffie-Hellman cho quá trình trao đổi khóa bí mật giữa Client và Server.
Sau khi đã trao đổi, sử dụng khóa bí mật trên và áp dụng một thuật toán mã hóa đối xứng bất kỳ để thực hiện trao đổi nội dung giữa Client và Server. *Ví dụ: Client gửi sử dụng khóa bí mật để mã hóa thông điệp gửi server, server nhận được thông điệp mã hóa và sử dụng khóa bí mật để giải mã thông điệp, hiển thị thông điệp ban đầu.*
Ngộn ngữ lập trình và kỹ thuật, loại mã hóa áp dụng do sinh viên lựa chọn, sao cho vẫn đảm bảo sử dụng Diffie-Hellman cho trao đổi khóa và dùng khóa đó cho quá trình mã hóa thông điệp. ■

Gợi ý 3.4. *Để thực hiện ứng dụng Client-Server và trao đổi thông điệp qua lại, sinh viên có thể nghiên cứu và sử dụng **Socket** cho ngôn ngữ mình đang sử dụng (C#, Java, Python,..). Ứng dụng hoàn chỉnh sẽ gồm ứng dụng phía server và ứng dụng phía client để có thể giao tiếp với nhau (có thể trên cùng máy hoặc giữa 2 máy trong cùng mạng nội bộ).*
Tham khảo ví dụ: `https://www.youtube.com/watch?v=B9AN3PDoAB8`

Mở rộng 3.3 Thực hiện ứng dụng chia sẻ file riêng tư giữa Client và Server:
- Sử dụng giao thức Diffie-Hellman để trao đổi khóa giữa Client và Server.
- Sử dụng khóa đó và áp dụng giao thức DES/AES để thực hiện mã hóa file khi gửi từ Client đến server. Khi Server nhận được sẽ tiến hành giải mã file gốc và có thể lưu lại trên máy.

Task 3.4 Trong giao thức trao đổi khóa Diffie-Hellman *(DH)*, nếu kẻ tấn công có các giá trị $g, p, g^a mod\ p, g^b mod\ p$ thì khóa bí mật đang trao đổi có bị lộ không? Tại sao?
DH có hoàn toàn an toàn không? Mô tả hình thức tấn công và cách hạn chế *(nếu có)*. ■

Gợi ý 3.5. *Ví dụ về một loại tấn công giao thức trao đổi khóa Diffie-Hellman:*



Hình 3.6: Một tấn công trao đổi khóa Diffie-Hellman

3.3 YÊU CẦU - ĐÁNH GIÁ

3.3.1 Yêu cầu

Sinh viên thực hiện đầy đủ các nội dung trong phần thực hành theo nhóm (tối đa **2** sinh viên/nhóm) hoặc cá nhân *(đăng ký từ buổi 1)* và báo cáo kết quả thực hiện như sau:

- File báo cáo .pdf kết quả thực hiện các nội dung thực hành (theo mẫu).
- Đối với các ứng dụng, đặt vào thư mục có tên tương ứng với Task *(source code và file thực thi)*

Báo cáo trình bày trực quan việc thực hành có kèm hình ảnh và chú thích rõ ràng theo cách tiếp cận của bản thân, không sao chép lẫn nhau. Nếu có tham khảo cần ghi nguồn.

Thời gian hoàn thành Lab 3:
Trong tối đa **12** ngày từ buổi thực hành tại lớp.

Lưu ý 3.3.1 Đặt tất cả báo cáo và thư mục liên quan vào 1 tập tin nén (.zip,.rar) có tên:

[MSSV1]-[Tên SV1]_[MSSV2]-[Tên SV2]_MMH_LabX

Ví dụ: 16520901-Nhut_16521516-Lam_MMH_Lab3. Kích thước tối đa: 10MB.
Nộp báo cáo theo thời gian quy định tại website môn học.

3.3.2 Đánh giá

- Sinh viên hiểu và hoàn thành tốt nội dung thực hành, đúng hạn: 80%
- Sinh viên trình bày cụ thể, rõ ràng kết quả thực hiện: 5-10%
- Khuyến khích sinh viên có tìm hiểu, thực hiện các nội dung mở rộng: 15-20% hoặc cộng trực tiếp vào điểm tổng kết thực hành nếu hoàn thành xuất sắc. *Nội dung mở rộng khuyến nghị thực hiện với sinh viên lớp ANTN.*
- Sao chép bài của sinh viên khác, tham khảo không ghi nguồn: -50% đến -100%.
- Nộp bài trễ: -20% mỗi ngày nộp trễ.

Lưu ý 3.3.2 Báo cáo nộp trễ, sao chép sẽ được xử lý tùy theo mức độ. Các nội dung báo cáo có thể được vấn đáp để đánh giá kết quả tại buổi thực hành tiếp theo, sinh viên vắng thực hành không có lý do thì nhóm sinh viên tương ứng sẽ được trừ tối thiểu 30% điểm bài thực hành đó.

3.4 TÀI LIỆU THAM KHẢO

[1] W. Stallings, *Cryptography and network security: Principles and practice, 6th ed*. Boston, MA, United States: Prentice Hall, 2013. *Chapter 8 -9*
[2] Asecurity Site, Available: `http://asecuritysite.com/`.
[3] Bernhard Esslinger, *Learning and Experiencing Cryptography with CrypTool and SageMath, 12th ed*, CrypTool Project. Available: `https://www.cryptool.org/en/ctp-documentation`.