TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

NHẬP MÔN AN TOÀN THÔNG TIN – IT4015

Chủ đề: Xác thực thông điệp

Mã lớp học: 115655

Thành viên: Nguyễn Ngọc Đức - 20173025
Nguyễn Đức Giang – 20173080
Giáp Ngọc Hiếu – 20173112
Đỗ Minh Vũ - 20173471

Giảng viên hướng dẫn: PGS.TS Nguyễn Linh Giang

Hà Nội, tháng 06 năm 2020

Chương 1. TỔNG QUAN VỀ XÁC THỰC ĐIỆN TỬ

2.1. VẤN ĐỀ XÁC THỰC ĐIỆN TỬ

2.1.1. Khái niệm xác thực

2.1.1.1. Xác thực theo nghĩa thông thường

Xác thực là một chứng thực một cái gì đó (hoặc một người nào đó) đáng tin cậy, có nghĩa là, những lời khai báo do người đó đưa ra hoặc về vật đó là sự thật.

Xác thực một đối tượng còn có nghĩa là công nhận nguồn gốc (provenance) của đối tượng, trong khi, xác thực một người thường bao gồm việc thẩm tra nhận dạng họ. Việc xác thực thường phụ thuộc vào một hoặc nhiều nhân tố xác thực (authentication factors) để minh chứng cụ thể.

2.1.1.2. Xác thực điện tử

Xác thực trong an ninh máy tính là một quy trình nhằm cố gắng xác minh nhận dạng số (digital identity) của phần truyền gửi thông tin (sender) trong giao thông liên lạc chẳng hạn như một yêu cầu đăng nhập. Phần gửi cần phải xác thực có thể là một người dùng một máy tính, bản thân một máy tính hoặc một chương trình máy tính (computer program).

Ngược lại sự tin cậy mù quáng (blind credential) hoàn toàn không thiết lập sự đòi hỏi nhận dạng, song chỉ thiết lập quyền hoặc địa vị hẹp hòi của người dùng hoặc của chương trình ứng dụng mà thôi.

Trong một mạng lưới tín nhiệm, việc "xác thực" là một cách để đảm bảo rằng người dùng chính là người mà họ nói họ là, và người dùng hiện đang thi hành những chức năng trong một hệ thống, trên thực tế, chính là người đã được ủy quyền để làm những việc đó.

2.1.2. Phân loại xác thực điện tử

2.1.2.1. Xác thực dữ liệu

1). Xác thực thông điệp (Message Authentication)

2). Xác thực giao dịch (Transaction Authentication)

3). Xác thực khóa (Key Authentication)

4). Xác thực nguồn gốc dữ liệu (Source của Data)

5). Xác thực bảo đảm toàn vẹn dữ liệu (Data Integrity)

2.1.2.2. Xác thực thực thể

1). Xác thực dựa vào thực thể: Biết cái gì (Something Known)

2). Xác thực dựa vào thực thể: Sở hữu cái gì (Something Possessed)

3). Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent)

2.2. XÁC THỰC DỮ LIỆU

2.2.1. Xác thực thông điệp

1). Khái niệm

Xác thực thông điệp hay ***Xác thực tính nguyên bản*** của dữ liệu (Data Origin Authentication) là một kiểu ***xác thực đảm bảo một thực thể*** được chứng thực là **nguồn gốc thực sự** tạo ra dữ liệu này ở một thời điểm nào đó.

Xác thực thông điệp bao hàm cả **tính toàn vẹn dữ liệu**, nhưng ***không đảm bảo*** tính duy nhất và sự phù hợp về thời gian của nó.

2.2.2. Xác thực giao dịch

1). Khái niệm

Xác thực giao dịch là ***Xác thực thông điệp*** cộng thêm việc **đảm bảo tính duy nhất** (Uniqueness) và sự phù hợp về **thời gian** (Timeliness) của nó.

Xác thực giao dịch liên quan đến việc sử dụng các tham số thời gian (**TVB-** Time Variant Parameters).

Transaction Authentication = Message Authentication + TVB

Xác thực giao dịch "mạnh hơn" Xác thực thông điệp.

2) Ví dụ

Một thông điệp gửi đi có thể đã bị chặn và phát lại (tương tự như việc đổi tiền bằng một bản sao của Séc). Để ngăn chặn tình huống này, người gửi và người nhận có thể gắn vào thông điệp ***nhãn thời gian*** hoặc ***số thông điệp.***

Số thông điệp là một con số được gắn vào thông điệp. Nó có thể chỉ dùng một lần duy nhất, giá trị không lặp lại, hoặc dùng dưới dạng dãy số tuần tự (Sequence Numbers).

Thám mã không có cách nào để biết được các bit của số này nằm ở vị trí nào trong thông điệp, hoặc không thể biết cách thay đổi các bit để tạo ra dạng mã hóa của số tiếp sau, hoặc không thể biết cách thay đổi các bit này mà không làm gián đoạn việc giải mã phần còn lại của thông báo.

Số thông báo này khó thể bị thay thế, thay đổi hoặc giả mạo. Người nhận phải duy trì việc đếm các số thông báo đã nhận được. Nếu hai người sử dụng một tập các số thì người nhận có thể biết được có thông báo nào trước thông báo hiện thời đã bị mất hoặc bị chậm trễ, vì số được mã hóa của thông báo hiện thời phải lớn hơn số được mã hóa của thông báo trước.

Nếu người gửi có nhiều thông báo thì có thể số thông báo sẽ quá dài. Vì thế, người ta thường đặt lại bộ đếm số thông báo trước khi nó đạt tới giá trị lớn nào đó. Lúc này tất cả bên thu phải được thông báo rằng, số thông báo được gửi tiếp theo sẽ được đặt lại về một số nhỏ (chẳng hạn là 0).

Nhãn thời gian (TimeStamp) là dấu hiệu về thời gian và ngày tháng lấy từ đồng hồ hệ thống hoặc đồng hồ địa phương. Bên gửi: gửi dữ liệu gắn TimeStamp đi. Bên nhận: nhận được dữ liệu, tiến hành lấy TimeStamp tại thời điểm hiện thời, trừ đi TimeStamp nhận được. Dữ liệu nhận được sẽ được chấp nhận nếu:

Độ lệch giữa 2 TimeStamp nằm trong khoảng chấp nhận được.

Không có thông báo nào có cùng TimeStamp được nhận trước đó từ cùng một người gửi. Điều này được thực hiện bằng cách bên nhận lưu giữ danh sách các TimeStamp từ người gửi để kiểm tra hoặc ghi lại TimeStamp gần nhất và chỉ chấp nhận TimeStamp có giá trị lớn hơn.

Như vậy, bên nhận phải đồng bộ và bảo mật về thời gian rất chặt chẽ với bên gửi, ngoài ra phải lưu giữ các TimeStamp.

2.2.3. Xác thực khóa

+ Xác thực không tường minh khóa (Implicit Key Authentication):

Một bên được đảm bảo rằng chỉ có bên thứ hai (và có thể có thêm các bên tin cậy-Trusted Parties) là ***có thể*** truy cập được khóa mật.

+ Khẳng định (Xác nhận) khóa (Key Confirmation):

Một bên được đảm bảo rằng bên thứ hai ***chắc chắn*** đã sở hữu khóa mật.

+ Xác thực tường minh khóa (Explicit Key Authentication)

Bao gồm cả 2 yếu tố trên, nó chứng tỏ được định danh của bên có khóa đã cho.

Chú ý:

Xác thực khóa tập trung vào định danh bên thứ hai có thể truy cập khóa hơn là giá trị của khóa. Khẳng định khóa lại tập trung vào giá trị của khóa.

Ta gọi ngắn gọn Explicit Key Authentication là Key Authentication.

Chú ý:

Xác thực dữ liệu đã bao gồm tính toán vẹn dữ liệu. Ngược lại thì không.

+ Đảm bảo xác thực nguồn gốc dữ liệu➔ phải đảm bảo tính toàn vẹn dữ liệu.

+ Đảm bảo tính toàn vẹn dữ liệu //➔ đảm bảo xác thực nguồn gốc dữ liệu

2.2.4. Xác thực nguồn gốc dữ liệu

Công cụ: Dùng chữ ký số, hàm băm, thủy vân ký.

2.2.5. Xác thực bảo đảm toàn vẹn dữ liệu

Công cụ: Dùng chữ ký số, hàm băm, thủy vân ký, mã xác thực.

2.3. XÁC THỰC THỰC THỂ

Xác thực thực thể (hay Định danh thực thể) là xác thực định danh của một đối tượng tham gia giao thức truyền tin.

Thực thể hay đối tượng có thể là người dùng, thiết bị đầu cuối,…

Tức là: Một thực thể được xác thực bằng định danh của nó đối với thực thể thứ hai trong một giao thức, và bên thứ hai đã thực sự tham gia vào giao thức.

2.3.1. Xác thực dựa vào thực thể: Biết cái gì (Something Known)

Chẳng hạn, mật khẩu, mật khẩu ngữ (pass phrase) hoặc số định danh cá nhân (personal identification number - PIN)

Định danh cá nhân (PIN- Personal Identifier Number) thường găn với **Something Possessed** để tăng tính bảo mật.

Chú ý:

"Biết cái gì" được dùng trong ***Giao thức định danh,*** đó là ***cơ chế hỏi – đáp*** (Challenge-Response):

Một thực thể (Claimant) chứng tỏ định danh của nó đối với thực thể khác (Verifier) bằng các biểu lộ hiểu biết về một thông tin mật liên quan nào đó cho Verifier, mà không bộc lộ bí mật của nó cho Verifier trong suốt giao thức.

Cơ chế đó gọi là ***"Chứng minh không tiết lộ thông tin"*.**

Trong cơ chế hỏi – đáp thường dùng một người được uỷ quyền có tín nhiệm TA (Trusted Autherity) để tạo các tham số chung, các thuật toán ký, kiểm tra chữ ký và các chuỗi định danh, dấu xác nhận cho các bên tham gia.

2.3.1.1 .Xác thực dựa trên User name và Password

Sự kết hợp của tên người dùng và mật khẩu là cách xác thực cơ bản nhất. Với kiểu xác thực này, chứng từ ủy nhiệm người dùng được đối chiếu với chứng từ được lưu trữ trên cơ sở dữ liệu hệ thống, nếu trùng khớp tên người dùng và mật khẩu, thì người dùng được xác thực và nếu không người dùng bị cấm truy cập. Phương thức này không an toàn lắm vì chứng từ xác nhận người dùng được gửi đi xác thực trong tình trạng "plain text", tức là không được mã hóa và có thể bị tóm trên đường truyền.

2.3.1.2. Giao thức Chứng thực bắt tay thách thức - Challenge Handshake Authentication Protocol (CHAP)

Giao thức Chứng thực "Bắt tay Thách thức" cũng là mô hình xác thực dựa trên tên người dùng/ mật khẩu. Khi người dùng cố gắng đăng nhập, server đảm nhiệm vai trò xác thực sẽ gửi một thông điệp thử thách (challenge message) trở lại máy tính người dùng. Lúc này máy tính người dùng sẽ phản hồi lại tên người dùng và mật khẩu được mã hóa. Server xác thực sẽ so sánh phiên bản xác thực người dùng được lưu giữ với phiên bản mã hóa vừa nhận. Nếu trùng khớp, ngươi dùng sẽ được xác thực. Bản thân mật khẩu không bao giờ được gửi qua mạng. Phương thức CHAP thường được sử dụng khi người dùng đăng nhập vào các "remote servers" của công ty chẳng hạn như RAS server. Dữ liệu chứa mật khẩu được mã hóa gọi là mật khẩu băm (hash password).

2.3.2. Xác thực dựa vào thực thể: Sở hữu cái gì (Something Possessed)

Ví dụ như sở hữu **khóa bí mật để ký điện tử**

Ví dụ như sở hữu thẻ từ (Magnetic-striped Card), thẻ tín dụng (Credit Card), thẻ thông minh (Smart Card), chứng minh thư (ID card), chứng chỉ an ninh (security token), chứng chỉ phần mềm (software token) hoặc điện thoại di động (cell phone)

2.3.2.1. Phương pháp xác thực Kerberos (Kerberos authentication)

Là phương pháp dùng một Server trung tâm để kiểm tra việc xác thực người dùng và cấp phát thẻ thông hành (service tickets) để người dùng có thể truy cập vào tài nguyên. Kerberos là một phương thức rất an toàn trong xác thực bởi vì dùng cấp độ mã hóa rất mạnh. Kerberos cũng dựa trên độ chính xác của thời gian xác thực giữa Server và máy khách, do đó cần đảm bảo có một "time server" hoặc "authenticating servers" được đồng bộ thời gian từ các "Internet time server". Kerberos là nền tảng xác thực chính của nhiều OS như Unix, Windows.

2.3.2.2. Phương pháp Tokens

Là phương tiện vật lý như các thẻ thông minh (smart cards) hoặc thẻ đeo của nhân viên (ID badges) chứa thông tin xác thực. Tokens có thể lưu trữ số nhận dạng cá nhân - personal identification numbers (PINs), thông tin về người dùng, hoặc mật khẩu.

Các thông tin trên token chỉ có thể được đọc và xử lý bởi các thiết bị đặc dụng, ví dụ như thẻ smart card được đọc bởi đầu đọc smart card gắn trên máy tính, sau đó thông tin này được gửi đến "authenticating server". Tokens chứa chuỗi text hoặc giá trị số duy nhất thông thương mỗi giá trị này chỉ sử dụng một lần.

Ví dụ về Smart cards:

Smart cards là ví dụ điển hình về xác thực tokens (token - based authentication). Một smart card là một thẻ nhựa có gắn một chip máy tính lưu trữ các loại thông tin điện tử khác nhau. Nội dung thông tin của card được đọc với một thiết bị đặc biệt.

2.3.3. Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent)

Chẳng hạn, vết lăn tay hoặc mẫu hình võng mạc mắt, chuỗi DNA (có đủ loại định nghĩa về cái nào là cái thích hợp và đầy đủ), mẫu hình về giọng nói (cũng có vài định nghĩa ở đây), sự xác minh chữ ký, tín hiệu sinh điện đặc hữu do cơ thể sống tạo sinh (unique bio-electric signals), hoặc những biệt danh sinh trắc (biometric identifier).

2.3.3.1. Phương pháp Biometrics (phương pháp nhận dạng sinh trắc học)

Mô hình xác thực dựa trên đặc điểm sinh học của từng cá nhân:

+ Quét dấu vân tay (fingerprint scanner)

+ Quét võng mạc mắt (retinal scanner)

+ Nhận dạng giọng nói (voice-recognition)

+ Nhận dạng khuôn mặt (facerecognition)

Vì nhận dạng sinh trắc học hiện rất tốn kém chi phí khi triển khai nên chưa được sử dụng rộng rãi như các phương thức xác thực khác.

Trong lịch sử, vết lăn tay là một phương pháp xác minh đáng tin nhất, song trong những vụ kiện tòa án (court cases) gần đây ở Mỹ và ở nhiều nơi khác, người ta đã có nhiều nghi ngờ về tính đáng tin cậy của dấu lăn tay. Những phương pháp sinh trắc khác được coi là khả quan hơn (quét võng mạng mắt và quét vết lăn tay là vài ví dụ), song có những bằng chứng chỉ ra rằng những phương pháp này, trên thực tế, dễ bị giả mạo.

Chương 2. PHƯƠNG PHÁP XÁC THỰC THÔNG ĐIỆP

3.1. XÁC THỰC THÔNG ĐIỆP BẰNG CHỮ KÝ SỐ

3.1.1. Ý tưởng chính của phương pháp xác thực bằng chữ ký số

1/. An gửi cho Thu cặp tin (X, Y), trong đó X là bản tin, Y là chữ ký số của bản tin X. Tức là $Y = Sig_k(X)$, Sig_k là thuật toán ký với khóa k.

2/. Khi nhận được (X, Y), Thu tiến hành kiểm tra chữ ký bằng thuật toán Ver (X,Y). Nếu $Ver_k(X, Y)$ = đúng thì Thu chắc chắn rằng X được bảo toàn.

Có hai khả năng:

+ An sử dụng chữ ký khôi phục được thông điệp gốc (chữ ký RSA)

+ An sử dụng chữ ký không khôi phục được thông điệp gốc (chữ ký ELGAMAL, chữ ký DSS).

Ta lấy chữ ký RSA và chữ ký ELGAMAL làm ví dụ cho hai khả năng trên.

3.1.2. Phương pháp chữ ký điện tử RSA

3.1.2.1. Sơ đồ chữ ký

1/. Sơ đồ (đề xuất năm 1978)

**** Tạo cặp khóa (bí mật, công khai)* (a, b):**

Chọn bí mật số nguyên tố lớn p, q, tính n = p * q, công khai n, đặt $P = C = Z_n$

Tính bí mật $\phi(n) = (p-1).(q-1)$. Chọn khóa công khai $b < \phi(n)$, nguyên tố với $\phi(n)$.

Khóa bí mật a là phần tử nghịch đảo của b theo mod $\phi(n)$: $a*b \equiv 1 \pmod{\phi(n)}$.

Tập khóa (bí mật, công khai) $K = \{(a, b)/\ a,b \in Z_n , a*b \equiv 1(\text{mod } \phi(n))\}$.

**** Ký số:*** Chữ ký trên $x \in P$ là $y = Sig_k(x) = x^a \pmod n$, $y \in A$. (R1)

**** Kiểm tra chữ ký:*** $Ver_k(x, y)$ = đúng $\Leftrightarrow x \equiv y^b \pmod n$. (R2)

2/. Chú ý:

- Việc ký chẳng qua là mã hóa, việc kiểm thử lại chính là việc giải mã:
- Việc "ký số" vào x tương ứng với việc "mã hóa" tài liệu x.
- Kiểm thử chữ ký chính là việc giải mã "chữ ký", để kiểm tra xem tài liệu đã giải mã có đúng là tài liệu trước khi ký không. Thuật toán và khóa kiểm thử "chữ ký" là công khai, ai cũng có thể kiểm thử chữ ký được.

3.1.2.2. Ví dụ

1/. An muốn gửi cho Thu bản rõ x = 2. An tiến hành ký trên x = 2 như sau:

* Tạo cặp khóa (bí mật, công khai) (a, b):

Chọn bí mật số nguyên tố p = 3, q = 5, tính n = p*q = 3*5 = 15, công khai n.

Đặt P = C = Z_n . Tính bí mật $\phi(n) = (p-1).(q-1) = 2*4 = 8$.

Chọn khóa công khai $b = 3 < \phi(n)$, nguyên tố với $\phi(n) = 8$.

Khóa bí mật a = 3, là phần tử nghịch đảo của b theo mod $\phi(n)$: $a*b \equiv 1 \pmod{\phi(n)}$.

* Ký số: Chữ ký trên $x = 2 \in P$ là

$y = Sig_k(x) = x^a \pmod n = 2^3 \pmod{15} = 8, \quad y \in A.$

An sẽ gửi cho Thu cả bản rõ x = 2 và chữ ký y = 8

2/. Thu sau khi nhận được bản rõ x = 2 và chữ ký y = 8 sẽ thực hiện kiểm tra chữ ký

* Kiểm tra chữ ký: $Ver_k(x, y) =$ đúng $\Leftrightarrow x \equiv y^b \pmod n$

$\Leftrightarrow 2 \equiv 8^b \pmod{15}$.

Nếu kết quả $Ver_k(x, y)$ đúng bằng 2 như bản rõ thì Thu có thể xác định chữ ký là đúng của bản rõ An gửi.

3.1.3. Phương pháp chữ ký điện tử ElGamal

Chữ ký điện tử ElGamal được giới thiệu vào năm 1985. Sau đó, Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) đã sửa đổi bổ sung phương pháp này thành chuẩn chữ ký điện tử (Digital Signature Standard– DSS).

3.1.3.1. Bài toán logarit rời rạc

Bài toán logarit rời rạc: Cho số nguyên tố p, gọi $\alpha \in Z_p$ là phần tử sinh (generator) và $\beta \in Z_p^*$. Cần xác định số nguyên dương $a \in Z_{p-1}$ sao cho

$$\alpha^a \equiv \beta \pmod{p}$$

Khi đó, a được ký hiệu là $\log_\alpha \beta$

Trên thực tế, bài toán logarit rời rạc thuộc nhóm bài toán *NP*, nói cách khác, chưa có thuật toán thời gian đa thức nào giải quyết được vấn đề này. Với p có tối thiểu 150 chữ số và $p - 1$ có thừa số nguyên tố đủ lớn, phép toán lũy thừa modulo p có thể xem như là hàm 1 chiều hay việc giải bài toán logarit rời rạc trên Z_p xem như không thể thực hiện được.

3.1.3.2. Sơ đồ chữ ký

1/. Sơ đồ (Elgamal đề xuất năm 1985)

**** Tạo cặp khóa (bí mật, công khai)* (a, h):**

Chọn số nguyên tố p sao cho bài toán logrit rời rạc trong Z_p là "khó" giải.

Chọn phần tử nguyên thủy $g \in Z_p^*$. Đặt P= Z_p^*, A= $Z_p^* \times Z_{p-1}$.

Chọn khóa bí mật là $a \in Z_p^*$. Tính khóa công khai $h \equiv g^a \bmod p$.

Định nghĩa tập khóa: $K = \{(p, g, a, h): h \equiv g^a \bmod p\}$.

Các giá trị p, g, h được công khai, phải giữ bí mật a.

**** Ký số:*** Dùng 2 khóa ký: khóa a và khóa ngẫu nhiên bí mật $r \in Z_{p-1}^*$.

(Vì $r \in Z_{p-1}^*$ nên nguyên tố cùng p -1, do đó tồn tại $r^{-1} \bmod (p-1)$).

Chữ ký trên $x \in P$ là $y = Sig_k(x, r) = (\gamma, \delta)$, $y \in A$ (E1)

Trong đó $\gamma \in Z_p^*$, $\delta \in Z_{p-1}$:

$\gamma = g^r \bmod p$ và $\delta = (x - a*\gamma)* r^{-1} \bmod (p-1)$

**** Kiểm tra chữ ký:***

$Ver_k(x, \gamma\ \delta)$ = đúng $\Leftrightarrow$ $h^{\gamma} * \gamma^{\delta} \equiv g^{x} \bmod p$. (E2)

2/. Chú ý: Nếu chữ ký được tính đúng, kiểm thử sẽ thành công vì

$$h^{\gamma} * \gamma^{\delta} \equiv g^{a\gamma} * g^{r*\delta} \bmod p \equiv g^{(a\gamma + r*\delta)} \bmod p \equiv g^{x} \bmod p.$$

Do $\delta = (x - a * \gamma) * r^{-1} \bmod (p-1)$ nên

$(a * \gamma + r * \delta) \equiv (a * \gamma + r\,(x - a * \gamma) * r^{-1}) \equiv (a * \gamma + x - a * \gamma) \equiv x \bmod (p-1)$.

3.1.3.3. Ví dụ

1/. An gửi cho Thu bản rõ x = 112

**** Tạo cặp khóa (bí mật, công khai)* (a, h):**

Chọn số nguyên tố p = 463. Đặt P = Z_p^*, A = $Z_p^* \times Z_{p-1}$.

Chọn phần tử nguyên thủy g = 2 $\in Z_p^*$.

Chọn khóa bí mật là a = 211 $\in Z_p^*$.

Tính khóa công khai $h \equiv g^a \bmod p = 2^{211} \bmod 463 = 249$.

Định nghĩa tập khóa: $K = \{(p, g, a, h): h \equiv g^a \bmod p\}$.

Các giá trị p, g, h được công khai, phải giữ bí mật a.

**** Ký số:*** Chọn ngẫu nhiên bí mật r = 235 $\in Z_{p-1}^*$, Khóa ký là (a, r).

Vì r $\in Z_{p-1}^*$ nên nguyên tố cùng p-1, do đó tồn tại $r^{-1} \bmod (p-1)$. Cụ thể:

UCLN(r, p-1) = UCLN(235, 462) = 1, nên $r^{-1} \bmod (p-1) = 235^{-1} \bmod 462 = 289$.

Chữ ký trên dữ liệu x = 112 là (γ, δ) = (16, 108), trong đó:

$\gamma = g^r \bmod p = 2^{235} \bmod 463 = 16$

$\delta = (x - a*\gamma) * r^{-1} \bmod (p-1) = (112 - 211 * 16) * 289 \bmod 462 = 108$

2/. Thu nhận được và tiến hành xác thực

**** Kiểm tra chữ ký:*** $Ver_k(x, \gamma\ \delta)$ = đúng $\Leftrightarrow$ $h^{\gamma} * \gamma^{\delta} \equiv g^{x} \bmod p$.

$h^{\gamma} * \gamma^{\delta} = 249^{16} * 16^{108} \bmod 463 = 132$

$g^{x} \bmod p = 2^{112} \bmod 463 = 132$.

Hai giá trị đó bằng nhau, như vậy chữ ký là đúng.

3.2. XÁC THỰC THÔNG ĐIỆP BẰNG HÀM BĂM

3.2.1. Ý tưởng chính của phương pháp xác thực bằng hàm băm

1/. A gửi cho B cặp tin (X, Y), trong đó X là bản tin, Y là đại diện bản tin X, tức là Y= h (X), h là hàm băm.

2/. Khi nhận được (X, Y), B tính lại h(X) = Z.

Nếu Z = Y, thì B chắc chắn rằng X được bảo toàn, không bị sửa đổi trên đường truyền.

3.2.2. Hàm băm MD4

3.2.2.1. Khái niệm "Thông điệp đệm"

"Thông điệp đệm" (Message Padding) là xâu bit có độ dài chia hết cho 512.

"Thông điệp đệm" được lưu trong mảng M = M[1] M[2] … M[N-1].

Trong đó M[i] là xâu bít có độ dài 32 bit, gọi là *word*

N≡0 mod 16. (32 x 16 = 512)

M được xây dựng từ Bản tin gốc a bằng thuật toán:

1. **d** = 447 – (|**a**| mod 512). (= 512 nếu |**a**| mod 512 > 447)
2. Giả sử **l** là kí hiệu biểu diễn nhị phân của |a| mod 2^{64}, tl: | **l** | = 64
3. **M = a** || 1 || 0^d || **l**

* Độ dài của xâu **a** || 1 || 0^d **là** |**a**| + 1 + **d** = 448 mod 512.

* Độ dài của "Thông điệp đệm" **M** là:

448 mod 512 + | **l** | = 448 mod 512 + 64 = 512 mod 512

Chú ý: Vì **M = a** || 1 || 0^d || **l** nên

d = |M| - (|a| + 1 + l) =

512 **-(|a| + 1 + 64) =** 512 – (|**a**| + 65) = 447 – (|**a**| mod 512)

Ví dụ

Xâu đầu vào là **a** = "ABC", xây dựng M như sau:

a: = "ABC" = "01000001 01000010 01000011". (Chú ý: „A"= 65).

* Độ dài tính theo bit của xâu ***a :*** |***a***| = 24 bit

=> d = 447 – (|***a***| mod 512) = 423

|***a***| + 1 + d = 24 +1 + 423 = 448 mod 512

* Biểu diễn nhị phân của độ dài xâu **a** là ***l:***

l = |***a***| mod 2^{64} = 24 mod 2^{64} = 24 = 16 + 8 = $(\underbrace{00\ldots\ldots00}_{59\text{ số}}11000)_2$

➔ Độ dài của l là |***l***| = |$\underbrace{00\ldots\ldots00}_{59\text{ số}}$11000| = 59 +5 = 64

M = **a** || 1 || 0^d || ***l***

➔ M = 01000001 01000010 01000011 || 1 || 00……00 || 00……0011000

423 số 59 số

M = M[0] M[1] … M[N-1], N ≡ 0 mod 16

M[0] = 01000001 01000010 01000011 10000000

M[1] = M[2] = ….. =M[13] = M[14] = $\underbrace{00\ldots..00}_{32\text{ số}}$

M[15] = 00000000 00000000 00000000 00011000

Trong việc xây dựng M, ta gắn số **l** đơn lẻ vào sau **a,** sau đó thêm tiếp các số 0 vào đủ để độ dài của M đồng dư với 448 modulo 512. Cuối cùng nối thêm 64 bit (chính là |***l***|) chứa biểu diễn nhị phân về độ dài ban đầu của ***x*** (được rút gọn theo modulo 2^{64} nếu cần).

Xâu kết quả M có độ dài chia hết cho 512. Vì thế khi chặt M thành các *word* 32 bit, số *word* nhận được là N sẽ chia hết cho 16.

Mục đích việc tạo ra mảng M – "thông điệp đệm" – là để các hàm băm xử lý trên từng khối (block) 512 bit, tức là 16 *word,* cùng một lúc.

3.2.2.2. Thuật toán

INPUT : Thông điệp là một xâu ***a*** có độ dài ***b*** bit.

OUTPUT: Bản băm có độ dài cố định 128 bit đại diện cho thông điệp gốc a.

1) Tóm tắt thuật toán

***Bước 1:* Khởi tạo các thanh ghi**

Có 4 thanh ghi để tính toán nhằm đưa ra các đoạn mã : A, B, C, D. Bản tóm lược của thông điệp được xây dựng như sự kết nối của các thanh ghi. Mỗi thanh ghi có độ dài 32 bit. Các thanh ghi này được khởi tạo giá trị hecxa.

word A:= 67 45 23 01 *word* B: = ef cd ab 89

word C:= 98 ba dc fe *word* D:= 10 32 54 76

Bước 2:

Xử lý thông điệp ***a*** trong 16 khối *word*, nghĩa là xử lý cùng một lúc 16 *word* = 512 bit.

Chia mảng M thành các khối 512 bit, đưa từng khối 512 bit vào mảng T[j]. Mỗi lần xử lý một khối 512 bit. Lặp lại N/16 lần.

2). Thuật toán MD4

Bước 1/. A:= **67 45 23 01** B:= **ef cd ab 89**

C:= **98 ba dc fe** D:= **10 32 54 76**

Bước 2/. **FOR** i:= 0 **TO** N/16 - 1 **DO**

for j:= 0 **to** 15 **do T**[j] = M[16 i + j];

AA :=A; BB :=B;

CC :=C; DD :=D;

Mỗi lần xử lý 16 từ, mỗi từ 32 bit, tl: 512 bit

Bước 3/. **Vòng 1**

Vòng 2

Vòng 3

Bước 4/. A = A + AA; B = B + BB; C = C + CC; D = D + DD;

Gán giá trị cho 4 biến AA, BB, CC, DD bằng 4 thanh ghi A, B, C, D tương ứng.

3). Các phép tính và các hàm dùng trong Thuật toán MD4

**** Các phép toán logic được sử dụng trong ba vòng.***

$X \wedge Y$ là phép toán AND theo bit giữa X và Y

$X \vee Y$ là phép toán OR theo bit giữa X và Y

$X \oplus Y$ là phép toán XOR theo bit giữa X và Y

$\neg X$ chỉ phép bù của X

$X + Y$ là phép cộng theo modulo 2^{32}

$X <<< s$ là phép dịch vòng trái đi s vị trí ($0 \leq s \leq 31$)

**** Ba hàm F, G, H dùng tương ứng trong vòng 1, 2, 3***

Mỗi hàm này là một hàm boolean tính theo bit.

$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$

$G(X, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$

$H(X, Y, Z) = X \oplus Y \oplus Z$

Ba vòng trong MD4 là hoàn toàn khác nhau. Mỗi vòng gồm một trong 16 *word* trong T được xử lý. Các phép toán được thực hiện trong ba vòng tạo ra các giá trị mới trong bốn *thanh ghi*. Cuối cùng, bốn *thanh ghi* được cập nhật ở bước 4 bằng cách cộng ngược các giá trị lưu trước đó ở bước 2, bước3. Phép cộng này được xác định là cộng các số nguyên dương, được rút gọn theo modulo 2^{32}.

4). Ba vòng "băm"

Vòng 1

1. A = (A + F (B, C, D) + T[0]) <<< 3
2. D = (D + F (A, B, C) + T[1]) <<< 7
3. C = (C + F (D, A, B) + T[2]) <<< 11
4. B = (B + F (C, D, A) + T[3]) <<< 19
5. A = (A + F (B, C, D) + T[4]) <<< 3
6. D = (D + F (A, B, C) + T[5]) <<< 7
7. C = (C + F (D, A, B) + T[6]) <<< 11
8. B = (B + F (C, D, A) + T[7]) <<< 19
9. A = (A + F (B, C, D) + T[8]) <<< 3
10. D = (D + F (A, B, C) + T[9]) <<< 7
11. C = (C + F (D, A, B) + T[10]) <<< 11
12. B = (B + F (C, D, A) + T[11]) <<< 19
13. A = (A + F (B, C, D) + T[12]) <<< 3
14. D = (D + F (A, B, C) + T[13]) <<< 7
15. C = (C + F (D, A, B) + T[14]) <<< 11
16. B = (B + F (C, D, A) + T[15]) <<< 19

Kết quả "băm" **a** *sau khi được xử lý qua vòng 1.*

1. 64B3DA82	5. 3D5E5934	9. 59798D5E	13. 7551AAC6
2. 34D8EB03	6. 489D5140	10 D206302D	14. 789B984F
3. B7BCB118	7. CCD14D6C	11. 753D6134	15. F55A1F31
4. 6D91B115	8. 454D14D6C	12. F52AED08	16. ABA71E22

Vòng 2

1. A = (A + G (B, C, D) + T[0] + 5A827999) <<< 3
2. D = (D + G (A, B, C) + T[4] + 5A827999) <<< 5
3. C = (C + G (D, A, B) + T[8] + 5A827999) <<< 9
4. B = (B + G (C, D, A) + T[12] + 5A827999) <<< 13
5. A = (A + G (B, C, D) + T[1] + 5A827999) <<< 3
6. D = (D + G (A, B, C) + T[5] + 5A827999) <<< 5
7. C = (C + G (D, A, B) + T[9] + 5A827999) <<< 9
8. B = (B + G (C, D, A) + T[13] + 5A827999) <<< 13
9. A = (A + G (B, C, D) + T[2] + 5A827999) <<< 3
10. D = (D + G (A, B, C) + T[6] + 5A827999) <<< 5
11. C = (C + G (D, A, B) + T[10] + 5A827999) <<< 9
12. B = (B + G (C, D, A) + T[14] + 5A827999) <<< 13
13. A = (A + G (B, C, D) + T[3] + 5A827999) <<< 3
14. D = (D + G (A, B, C) + T[7] + 5A827999) <<< 5
15. C = (C + G (D, A, B) + T[11] + 5A827999) <<< 9
16. B = (B + G (C, D, A) + T[15] + 5A827999) <<< 13

Giá trị 5A827999 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả "băm" **a** *sau khi được xử lý qua vòng 2.*

1. 558C2E28	5. 558C2E28	9. 31E9FE4A	13. B60A11E6
2. 5A0E08F9	6. 5A0E08F9	10. 6F68E462	14. 2DED6D8E
3. F6A9B390	7. F6A9B390	11. D745F88A	15. A2870B31
4. 7876BC8F	8. 7876BC8F	12. 7050BC10	16. 4384D178

Vòng 3

1. A = (A + H (B, C, D) + T[0] + 6ED9EBA1) <<< 3
2. D = (D + H (A, B, C) + T[8] + 6ED9EBA1) <<< 9
3. C = (C + H (D, A, B) + T[4] + 6ED9EBA1) <<< 11
4. B = (B + H (C, D, A) + T[12] + 6ED9EBA1) <<< 15
5. A = (A + H (B, C, D) + T[2] + 6ED9EBA1) <<< 3
6. D = (D + H (A, B, C) + T[10] + 6ED9EBA1) <<< 9
7. C = (C + H (D, A, B) + T[6] + 6ED9EBA1) <<< 11
8. B = (B + H (C, D, A) + T[14] + 6ED9EBA1) <<< 15
9. A = (A + H (B, C, D) + T[1] + 6ED9EBA1) <<< 3
10. D = (D + H (A, B, C) + T[9] + 6ED9EBA1) <<< 9
11. C = (C + H (D, A, B) + T[5] + 6ED9EBA1) <<< 11
12. B = (B + H (C, D, A) + T[13] + 6ED9EBA1) <<< 15
13. A = (A + H (B, C, D) + T[3] + 6ED9EBA1) <<< 3
14. D = (D + H (A, B, C) + T[11] + 6ED9EBA1) <<< 9
15. C = (C + H (D, A, B) + T[7] + 6ED9EBA1) <<< 11
16. B = (B + H (C, D, A) + T[15] + 6ED9EBA1) <<< 15

Giá trị 6ED9EBA1 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả "băm" **a** *sau khi được xử lý qua vòng 3.*

1. 98A7C489	5. F3031C80	9. C02E826B	13. 03477E5E
2. E70B031C	6. 7D7A371B	10. F38DC78B	14. 77509F0A
3. A96B2FFA	7. 1C2487DE	11. E3C7F63B	15.FB3D792D
4. 58BE9F94	8. F7767709	12. 81AB00F	16. 23D73C06

5). Kết quả "băm"

Kết quả ra là đoạn mã có độ dài 128 bit, được thu gọn từ thông điệp **a** có độ dài ***b*** bit. Đoạn mã này thu được từ 4 thanh ghi A, B, C, D: bắt đầu từ byte thấp nhất của thanh ghi A cho đên byte cao nhất của thanh ghi D.

Với **a** = "ABC", Đại diện văn bản a là thông tin trên 4 thanh ghi liên tiếp: A, B, C, D, trong đó:

Thanh ghi A = 6A8CA15F

Thanh ghi B = 671E4A93

Thanh ghi C = 93F85626

Thanh ghi D = 3409907C

Chú ý : A = A + AA = 03477E5E

67452301

=6A8CA15F

3.2.2.3. Ví dụ

Để hiểu được cách xác thực thông điệp bằng hàm băm, ta sẽ xem sét ví dụ sau:

An muốn gửi cho Thu xâu ***a*** = "ABC", An tiến hành tạo đại diện của xâu "ABC" bằng hàm băm MD4 và được giá trị Đại diện văn bản là 4 thanh ghi trên.

An tiến hành gửi cho Thu xâu "ABC" và bản Đại diện trên.

(ABC, 6A8CA15F, 671E4A93, 93F85626, 3409907C)

An ⟶ Thu

Trong đó "ABC" là xâu đầu vào a

6A8CA15F, 671E4A93, 93F85626, 3409907C lần lượt là giá trị của các thanh ghi A, B, C, D.

Thu sau khi nhận được thông điệp của An, sẽ tiến hành xác thực bằng cách cũng tạo đại diện của xâu ***a*** bằng cùng phương pháp MD4. Nếu kết quả ra là bốn thanh ghi có giá trị trùng khớp với Đại diện mà An đã gửi kèm thì bản rõ mà An gửi là đúng, nếu không trùng lặp thì Thu sẽ loại bỏ vì chứng tỏ đã có sự giả mạo hoặc thay thế của ai đó đối với bản rõ.

Giả sử Minh là kẻ gian muốn thay đổi thông tin của An để gửi lại cho Thu, có hai trường hợp xảy ra:

+ Minh biết được bản rõ ***a*** = "ABC" nhưng không nắm được cách An tạo đại diện nên sẽ gửi cho Thu một đại diện khác.

(ABC, 6A8CB15F, 671C4A93, 92F85626, 3409906C)

Minh ⟶ Thu

Thu sau khi nhận được bản tin giả mạo mà Minh gửi sẽ tiến hành xác thực bằng cách băm xâu "ABC" bằng hàm băm MD4, sau đó tiến hành so sánh với các thanh ghi đại diện gửi kèm. Thu có thể thấy rõ sự sai lệch ở từng thanh ghi và đưa ra kết luận thông điệp mà An gửi đã bị thay đổi.

+ Minh không biết được bản rõ ***a*** = "ABC" nhưng lại tình cờ biết được đại diện của ***a*** là 4 thanh ghi A, B, C, D. Minh sẽ gửi Thu một thông điệp với xâu ***a'*** = "MINH" và đại diện trên, mong muốn được Thu chấp nhận.

(MINH, 6A8CA15F, 671E4A93, 93F85626, 3409907C)

Minh ⟶ Thu

Thu sau khi nhận được bản tin giả mạo mà Minh gửi sẽ tiến hành xác thực bằng cách băm xâu "MINH" bằng hàm băm MD4, sau đó tiến hành so sánh với các thanh ghi đại diện gửi kèm. Thu có thể thấy sự sai khác giữa 2 đại diện và đưa ra kết luận thông điệp mà An gửi đã bị thay đổi

3.2.3. Hàm băm MD5

3.2.3.1. Giới thiệu MD5

Hàm băm MD4 (Message Digest 4) được Giáo sư Rivest đề nghị vào năm 1990. Vào năm sau, phiên bản cải tiến MD5 của thuật toán này ra đời. Cùng với hàm băm SHS, đây là ba phương pháp có ưu điểm tốc độ xử lý nhanh nên thích hợp áp dụng trong thực tế đối với các thông điệp dài.

Thông điệp ban đầu x sẽ được mở rộng thành dãy bit X có độ dài là bội số của 512. Một bit 1 được thêm vào sau dãy bit x, tiếp đến là dãy gồm d bit 0 và cuối cùng là dãy 64 bit l biểu diễn độ dài của thông điệp x. Dãy gồm d bit 0 được thêm vào sao cho dãy X có độ dài là bội số 512. Quy trình này được thể hiện:

Thuật toán 3.2.3.1 *Thuật toán xây dựng dãy bit X từ dãy bit x*

$d = (447 - |x|) \bmod 512$

Gọi dãy 64 bit l là biểu diễn nhị phân của giá trị $|x| \bmod 2^{64}$.

$X = x \,||\, 1 \,||\, 0^d \,||\, l$

Đơn vị xử lý trong MD5 là các từ 32 - bit nên dãy X sẽ được biểu diễn thành dãy các từ $X[i]$ 32 bit: $X = X[0]\, X[1] \ldots X[N–1]$ với N là bội số của 16.

Thuật toán 3.2.3.2 *Hàm băm MD5*

```
A = 0x67452301;
B = 0xefcdab89;
C = 0x98badcfe;
D = 0x10325476;
for i = 0        to N/16 –1
for j = 0              to 15
M[j] = X[16i-j]
end for
AA = A
BB = B
```

CC = C

DD = D

Round1

Round2

Round3

Round4

A = A+AA

B = B+BB

C = C+CC

D = D+DD

end for

Đầu tiên, bốn biến *A, B, C, D* được khởi tạo. Những biến này được gọi là *chaining variables.*

Bốn chu kỳ (Round) biến đổi trong MD5 hoàn toàn khác nhau và lần lượt sử dụng các hàm *F, G, H* và *I*. Mỗi hàm có tham số *X, Y, Z* là các từ 32 bit và kết quả là một từ 32 bit.

$$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$$
$$G(X, Y, Z) = (X \wedge Z) \vee (Y \wedge (\neg Z))$$
$$H(X, Y, Z) = X \oplus Y \oplus Z \qquad (3.2.1)$$
$$I(X, Y, Z) = Y \oplus (X \vee (\neg Z))$$

Với quy ước:

X ^ Y là phép toán AND theo bit giữa X và Y

X ∨ Y là phép toán OR theo bit giữa X và Y

X ⊕ Y là phép toán XOR theo bit giữa X và Y

¬ X chỉ phép bù của X

X + Y là phép cộng theo modulo 2^{32}

X <<< ***s*** là phép dịch vòng trái đi ***s*** vị trí ($0 \leq s \leq 32$)

Định nghĩa các hàm:

FF (A, B, C, D, Mj, s, t_i):

A = B + ((A + F(B, C, D) + Mj + t_i) <<< s)

GG (A, B, C, D, Mj, s, t_i):

A = B + ((A + G(B, C, D) + Mj + t_i) <<< s)

HH (A, B, C, D, Mj, s, t_i):

A = B + ((A + H(B, C, D) + Mj + t_i) <<< s)

II (A, B, C, D, Mj, s, t_i):

A = B + ((A + I(B, C, D) + Mj + t_i) <<< s)

Với Mj là *M*[*j*] và hằng số t_i xác định theo công thức:

$ti = \lfloor 2^{32} \mid \sin(i) \mid \rfloor$, *i* tính bằng radian.

Bảng 3.2.3.1. Bốn chu kỳ biến đổi trong MD5.

Chu kỳ 1	Chu kỳ 2
FF(A, B, C, D, M0 , 7, 0xd76aa478)	GG(A, B, C, D, M1 , 5, 0xf61e2562)
FF(D, A, B, C, M1 ,12, 0xe8c7b756)	GG(D, A, B, C, M6 , 9, 0xc040b340)
FF(C, D, A, B, M2 ,17, 0x242070db)	GG(C, D, A, B, M11, 14, 0x265e5a51)
FF(B, C, D, A, M3 ,22, 0xclbdceee)	GG(B, C, D, A, M0 , 20, 0xe9b6c7aa)
FF(A, B, C, D, M4 , 7, 0xf57c0faf)	GG(A, B, C, D, M5 , 5, 0xd62fl05d)
FF(D, A, B, C, M5 ,12, 0x4787c62a)	GG(D, A, B, C, M10, 9, 0x02441453)
FF(C, D, A, B, M6 , 17, 0xa8304613)	GG(C, D, A, B, M15, 14, 0xd8ale681)
FF(B, C, D, A, M7 ,22, 0xfd469501)	GG(B, C, D, A, M4 , 20, 0xeid3fbc8)
FF(A, B, C, D, M8 , 7, 0x698098d8)	GG(A, B, C, D, M9 , 5,0x21elcde6)
FF(D, A, B, C, M9 ,12, 0x8b44f7af)	GG(D, A, B, C, M14, 9, 0xc33707d6)
FF(C, D, A, B, M10, 17, 0xffff5bbl)	GG(C, D, A, B, M3 , 14, 0xf4d50d87)
FF(B, C, D, A, M11, 22, 0x895cd7be)	GG(B, C, D, A, M8 , 20, 0x455al4ed)
FF(A, B, C, D, M12, 7, 0x6b901122)	GG(A, B, C, D, M13, 5, 0xa9e3e905)
FF(D, A, B, C, M13, 12, 0xfd987193)	GG(D, A, B, C, M2 , 9,0xfcefa3f8)
FF(C, D, A, B, M14, 17, 0xa679438e)	GG(C, D, A, B, M7, 14,0x676f02d9)
FF(B, C, D, A, M15, 22, 0x49b40821)	GG(B, C, D, A, M12, 20,0x8d2a4c8a)

Chu kỳ 3	Chu kỳ 4
HH(A, B, C, D, M5 , 4, 0xfffa3942)	II(A, B, C, D, M0 , 6, 0xf4292244)
HH(D, A, B, C, M8 , 11, 0x8771f6811	II(D, A, B, C, M7 , 10, 0x432aff97)
HH(C, D, A, B, M11, 16, 0x6d9d6122)	II(C, D, A, B, M14, 15, 0xab9423a7)
HH(B, C, D, A, M14, 23, 0xfde5380c)	II(B, C, D, A, M5 ,21, 0xfc93a039)
HH(A, B, C, D, M1 , 4, 0xa4beea44)	II(A, B, C, D, M12, 6, 0x655b59c3)
HH(D, A, B, C, M4 , 11, 0x4bdecfa9)	II(D, A, B, C, M3 ,10, 0x8f0ccc92)
HH(C, D, A, B, M7 , 16, 0xf6bb4b60)	II(C, D, A, B, M10, 15, 0xffeff47d)
HH(B, C, D, A, M10, 23, 0xbebfbc70)	II(B, C, D, A, M1 , 21, 0x85845ddl)
HH(A, B, C, D, M13, 4, 0x289biec6)	II(A, B, C, D, M8 , 6, 0x6fa87e4f)
HH(D, A, B, C, M0 , 11, 0xeaal27fa)	II(D, A, B, C, M15, 10, 0xfe2ce6e0)
HH(C, D, A, B, M3 , 16, 0xd4ef3085)	II(C, D, A, B, M6 , 15, 0xa3014314)
HH(B, C, D, A, M6 , 23, 0x04881d05)	II(B, C, D, A, M13, 21, 0x4e0811al)
HH(A, B, C, D, M9 , 4, 0xd9d4d039)	II(A, B, C, D, M4 , 6, 0xf7537e82)
HH(D, A, B, C, M12, 11, 0xe6db99e5)	II(D, A, B, C, M11, 10, 0xbd3af235)
HH(C, D, A, B, M15, 16, 0xlfa27cf8)	II(C, D, A, B, M2 , 15, 0x2ad7d2bb)
HH(B, C, D, A, M2 , 23, 0xc4ac5665)	II(B, C, D, A, M9 , 21, 0xeb86d391)

3.2.3.2. Nhận xét

Hàm băm MD5 có những ưu điểm cải tiến so với phương pháp MD4 [45]:

+ MD4 chỉ có ba chu kỳ biến đổI, trong khi MD5 được bổ sung thêm chu kỳ thứ tư giúp tăng mức độ an toàn.

+ Mỗi thao tác trong từng chu kỳ biến đổi của MD5 sử dụng các hằng số t_i phân biệt, trong khi MD4 sử dụng hằng số chung cho mọi thao tác trong cùng chu kỳ biến đổi (Trong MD4, hằng số t_i sử dụng trong mỗi chu kỳ lần lượt là 0, 0x5a827999, 0x6ed9eba1).

+ Hàm G ở chu kỳ hai của MD4: $G(X, Y, Z) = ((X \quad X) \vee (X \quad Z) \vee (Y \; Z))$, được thay thế bằng $((X \quad Z) \veebar (Y \; Z))$ nhằm giảm tính đối xứng.

+ Mỗi bước biến đổi trong từng chu kỳ chịu ảnh hưởng kết quả của bước biến đổi trước đó nhằm tăng nhanh tốc độ của hiệu ứng lan truyền (avalanche).

+ Các hệ số dịch chuyển xoay vòng trong mỗi chu kỳ được tối ưu hóa nhằm tăng tốc độ hiệu ứng lan truyền. Ngoài ra, mỗi chu kỳ sử dụng bốn hệ số dịch chuyển khác nhau.

3.2.4. Hàm băm Secure Hash Standard (SHS)

Hàm băm SHS do NIST và NSA xây dựng được công bố trên Federal Register vào ngày 31 tháng 1 năm 1992 và sau đó chính thức trở thành phương pháp chuẩn từ ngày 13 tháng 5 năm 1993.

Nhìn chung, SHS được xây dựng trên cùng cơ sở với phương pháp MD4 và MD5. Tuy nhiên, phương pháp SHS lại áp dụng trên hệ thống big-endian thay vì little-endian như phương pháp MD4 và MD5. Ngoài ra, thông điệp rút gọn kết quả của hàm băm SHS có độ dài 160 bit (nên phương pháp này thường được sử dụng kết hợp với thuật toán ký DSS).

Tương tự MD5, thông điệp nguồn x sẽ được chuyển thành một dãy bit có độ dài là bội số của 512. Từng nhóm gồm 16 từ-32 bit $X[0]$, $X[1]$,..., $X[15]$ sẽ được mở rộng thành 80 từ-32 bit $W[0]$, $W[1]$, ..., $W[79]$ theo công thức:

$$W[t] = \begin{cases} X[t], & 0 \le t \le 15 \\ X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16], & 16 \le t \le 79 \end{cases}$$

(3. 2. 2)

Trong phiên bản cải tiến của SHS, công thức trên được thay bằng:

$$W[t] = \begin{cases} X[t], & 0 \le t \le 15 \\ (X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16]) <<< 1, & 16 \le t \le 79 \end{cases}$$

(3. 2. 3)

Tương tự MD5, hàm băm SHS sử dụng bốn chu kỳ biến đổi, trong đó, mỗi chu kỳ gồm 20 bước biến đổi liên tiếp nhau. Chúng ta có thể xem như SHS bao gồm 80 bước biến đổi liên tiếp nhau. Trong đoạn mã chương trình dưới đây, hàm $f[t]$ và hằng số $K[t]$ được định nghĩa như sau:

$$f[t](X,Y,Z)=\begin{cases}(X\wedge Y)\vee((\neg X)\wedge Z), & 0\le t\le 19\\ X\oplus Y\oplus Z, & 20\le t\le 39\\ (X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z), & 40\le t\le 59\\ X\oplus Y\oplus Z, & 60\le t\le 79\end{cases}$$

(3.2.4)

$$K[t]=\begin{cases}0x5a827999, 0\le t\le 19\\ 0x6ed9eba1, 20\le t\le 39\\ 0x8f1bbcdc, 40\le t\le 59\\ 0xca62c1d6, 60\le t\le 79\end{cases}$$

(3.2.5)

A = 0x67452301;

B = 0xefcdab89;

C = 0x98badcfe;

D = 0x10325476;

E = 0xc3d2elf0;

for i=0 **to** N/16 –1

for t=0 **to** 15 do

W[t] = X[16 * t-j]

end for

for t=16 **to** 79

W[t] =(W[t-3] xor W[t-8] xor W[t-14] xor W[t-16])<<<1

a = A

b = B

c = C

d = D

e = E

```
for t=0          to 79
TEMP = (a<<<5) + f[t] (b, c, d) + e + W[t] + K[t]
e    = d
d    = c
c    = b <<< 30
b    = a
a    = TEMP
end for
A = A + a
B = B + b
C = C + c
D = D+ d
E = E + e
end for
```

3.2.4.1. Nhận xét

Hàm băm SHS rất giống với MD4 nhưng thông điệp rút gọn được tạo ra có độ dài 160-bit. Cả 2 phương pháp này đều là sự cải tiến từ MD4. Dưới đây là một số đặc điểm so sánh giữa MD5 và SHS:

+ Tương tự như MD5, hàm băm SHS cũng bổ sung thêm chu kỳ biến đổi thứ tư để tăng mức độ an toàn. Tuy nhiên, chu kỳ thứ tư của SHS sử dụng lại hàm *f* của chu kỳ thứ 2.

+ 20 bước biến đổi trong cùng chu kỳ của hàm băm SHS sử dụng hằng số chung *K*[*t*] trong khi mỗi bước biến đổi của hàm băm MD5 lại dùng các hằng số khác nhau.

+ Trong hàm băm MD5, hàm *G* ở chu kỳ thứ hai của MD4:
G(*X*, *Y*, *Z*) = ((X $\wedge$ *Y*) $\vee$ (*X* $\wedge$ *Z*) $\vee$ (*Y* *Z*)) được thay thế bằng ((*X* *Z*) $\vee$(*Y* *Z*)) nhằm giảm tính đối xứng. Hàm băm SHS vẫn sử dụng hàm *G* như trong MD4.

+ Trong MD5 và SHS, mỗi bước biến đổi chịu ảnh hưởng bởi kết quả của bước biến đổi trước đó để tăng nhanh hiệu ứng lan truyền.

Hiện tại vẫn chưa có phương pháp tấn công nào có thể áp dụng được đối với hàm băm SHS. Ngoài ra, do thông điệp rút gọn của hàm băm SHS có độ dài 160 bit nên có độ an toàn cao hơn đối với phương pháp tấn công brute-force (kể cả phương pháp birthday attack) so với hàm băm MD5.

3.2.5. Hàm băm SHA

3.2.5.1. Ý tưởng của các thuật toán hàm băm SHA

Các thuật toán hàm băm SHA gồm 2 bước: tiền xử lý và tính toán giá trị băm.

+ Bước tiền xử lý bao gồm các thao tác:

- Mở rộng thông điệp
- Phân tích thông điệp đã mở rộng thành các khối *m* bit
- Khởi tạo giá trị băm ban đầu

+ Bước tính toán giá trị băm bao gồm các thao tác:

- Làm *N* lần các công việc sau:
  - Tạo bảng phân bố thông điệp (message schedule) từ khối thứ *i*.
  - Dùng bảng phân bố thông điệp cùng với các hàm, hằng số, các thao tác trên từ để tạo ra giá trị băm *i*.
- Sử dụng giá trị băm cuối cùng để tạo thông điệp rút gọn.

Thông điệp *M* được mở rộng trước khi thực hiện băm,nhằm đảm bảo thông điệp mở rộng có độ dài là bội số của 512 hoặc 1024 bit tùy thuộc vào thuật toán. Sau khi thông điệp đã mở rộng, thông điệp cần được phân tích thành *N* khối *m*-bit trước khi thực hiện băm.

Đối với SHA-1 và SHA-256, thông điệp mở rộng được phân tích thành *N* khối 512-bit $M^{(1)}$, $M^{(2)}$,..., $M^{(N)}$. Do đó 512 bit của khối dữ liệu đầu vào có thể được thể hiện bằng 16 từ 32-bit, $M_0(i)$ chứa 32 bit đầu của khối thông điệp i, $M_1{}^{(i)}$ chứa 32 bit kế tiếp,…

Đối với SHA-384 và SHA-512, thông điệp mở rộng được phân tích thành *N* khối 1024-bit $M^{(1)}$, $M^{(2)}$,..., $M^{(N)}$. Do đó 1024 bit của khối dữ liệu đầu vào có thể được thể hiện bằng 16 từ 64-bit, $M_0^{(i)}$ chứa 64 bit đầu của khối thông điệp *i*, $M_1^{(i)}$ chứa 64 bit kế tiếp,...

Với mỗi thuật toán băm an toàn, giá trị băm ban đầu $H^{(0)}$ phải được thiết lập. Kích thước và số lượng từ trong $H^{(0)}$ tùy thuộc vào kích thước thông điệp rút gọn.

Các cặp thuật toán SHA-224 và SHA-256; SHA-384 và SHA-512 có các thao tác thực hiện giống nhau, chỉ khác nhau về số lượng bit kết quả của thông điệp rút gọn. Nói cách khác, SHA-224 sử dụng 224 bit đầu tiên trong kết quả thông điệp rút gọn sau khi áp dụng thuật toán SHA256. Tương tự SHA-384 sử dụng 384 bit đầu tiên trong kết quả thông điệp rút gọn sau khi áp dụng thuật toán SHA-512.

3.2.5.2. Khung thuật toán chung của hàm băm SHA

Trong hàm băm SHA, chúng ta cần sử dụng thao tác quay phải một từ, ký hiệu là ROTR, và thao tác dịch phải một từ, ký hiệu là SHR.

Hình 3.2.5.1. *Khung thuật toán chung cho hàm băm SHA*

for $i = 1$ **to** N

for $t = 0$ **to** 15

$W_t = M_t(i)$

end for

for $t = 16$ **to** *scheduleRound*

$W_t = \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}$

end for

$a = H_0^{(i-1)}$

$b = H_1^{(i-1)}$

$c = H_2^{(i-1)}$

$d = H_3^{(i-1)}$

$e = H_4^{(i-1)}$

$f = H_5^{(i-1)}$

$g = H_6^{(i-1)}$

$h = H_7^{(i-1)}$

for $t = 0$ **to** 63

$T_1 = h + \sum_1(e) + \text{Ch}(e, f, g) + K_t + W_t$

$T_2 = \sum_0(a) + \text{Maj}(a, b, c)$

$h = g$

$g = f$

$f = e$

$e = d + T_1$

$d = c$

$c = b$

$b = a$

$a = T_1 + T_2$

end for

$H_0^{(i)} = a + H_0^{(i-1)}$

$H_1^{(i)} = b + H_1^{(i-1)}$

$H_2^{(i)} = c + H_2^{(i-1)}$

$H_3^{(i)} = d + H_3^{(i-1)}$

$H_4^{(i)} = e + H_4^{(i-1)}$

$H_5^{(i)} = f + H_5^{(i-1)}$

$H_6^{(i)} = g + H_6^{(i-1)}$

$H_7^{(i)} = h + H_7^{(i-1)}$

end for

Mỗi thuật toán có bảng hằng số phân bố thông điệp tương ứng. Kích thước bảng hằng số thông điệp (scheduleRound) của SHA-224 và SHA-256 là 64, kích thước bảng hằng số thông điệp của SHA-384 và SHA-512 là 80.

Trong hàm băm SHA-224 và SHA-256, chúng ta cần sử dụng các hàm:

$$\text{Ch}(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$\text{Maj}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_0(x) = \text{ROTR}^2(x) \oplus \text{ROTR}^{13}(x) \oplus \text{ROTR}^{22}(x)$$
$$\sum_1(x) = \text{ROTR}^6(x) \oplus \text{ROTR}^{11}(x) \oplus \text{ROTR}^{25}(x)$$
$$\sigma_0(x) = \text{ROTR}^7(x) \oplus \text{ROTR}^{18}(x) \oplus \text{SHR}^3(x)$$
$$\sigma_1(x) = \text{ROTR}^{17}(x) \oplus \text{ROTR}^{19}(x) \oplus \text{SHR}^{10}(x)$$

Trong hàm băm SHA-384 và SHA-512, chúng ta cần sử dụng các hàm sau:

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_0 (x) = ROTR^{28}(x) \oplus ROTR^{34}(x) \oplus ROTR^{29}(x)$$
$$\sum_1 (x) = ROTR^{14}(x) \oplus ROTR^{18}(x) \oplus ROTR^{41}(x)$$
$$\sigma_0 (x) = ROTR^{1}(x) \oplus ROTR^{8}(x) \oplus SHR^{7}(x)$$
$$\sigma_1 (x) = ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^{6}(x)$$

3.2.5.3. Nhận xét

Chuẩn SHS đặc tả 5 thuật toán băm an toàn SHA-1, SHA-224[3], SHA-256, SHA-84 và SHA-512. Bảng 3.2.5.2 thể hiện các tính chất cơ bản của bốn thuật toán băm an toàn.

Sự khác biệt chính của các thuật toán là số lượng bit bảo mật của dữ liệu được băm – điều này có ảnh hưởng trực tiếp đến chiều dài của thông điệp rút gọn. Khi một thuật toán băm được sử dụng kết hợp với thuật toán khác đòi hỏi phải cho kết quả số lượng bit tương ứng. Ví dụ, nếu một thông điệp được ký với thuật toán chữ ký điện tử cung cấp 128 bit thì thuật toán chữ ký đó có thể đòi hỏi sử dụng một thuật toán băm an toàn cung cấp 128 bit như SHA-256.

Ngoài ra, các thuật toán khác nhau về kích thước khối và kích thước từ được sử dụng.

Bảng 3.2.5.2. *Các tính chất của các thuật toán băm an toàn*

Thuật toán	Kích thước (bit)				Độ an toàn[4] (đơn vị: bit)
	Thông điệp	Khối	Từ	Thông điệp rút gọn	
SHA-1	$< 2^{64}$	512	32	160	80
SHA-224	$< 2^{64}$	512	32	224	112
SHA-256	$< 2^{64}$	512	32	256	128
SHA-384	$< 2^{128}$	1024	64	384	192
SHA-512	$< 2^{128}$	1024	64	512	256

3.3. XÁC THỰC THÔNG ĐIỆP BẰNG MÃ XÁC THỰC

3.3.1. Định nghĩa mã xác thực thông điệp

Thông điệp được xác thực bằng **mã xác thực thông điệp** đi theo bản mã. Mã xác thực thông điệp thực chất là một giá trị băm của bản rõ, hoặc bản mã được tính theo thuật toán khác hẳn với thuật toán mã hóa thông điệp, rồi được mã hóa bằng một khóa khác với khóa mã hóa thông điệp. Nếu mã này xác thực nội dung bản rõ, thì nó cũng gián tiếp xác thực khóa đúng.

Mục đích của phần này là phải có được khả năng xác thực ngay cả khi có một đối phương tích cực - Minh là người có thể quan sát các bản tin trong kênh. Mục đích này có thể đạt được bằng cách thiết lập một „"khóa riêng"" K bằng cách để An và Thu chung một khoá bí mật trước khi mỗi bản tin được gửi đi.

Trong mục này ta sẽ nghiên cứu đảm bảo xác thực, chứ không phải các mã đảm bảo độ mật. Trong mã này, khoá sẽ được dùng để tính một mã xác thực cho phép Thu kiểm tra được tính xác thực của thông báo mà anh ta nhận được. Một ứng dụng khác của mã xác thực là kiểm tra các số liệu trong một file lớn có bị can thiệp vào một cách hợp pháp hay không. Nhãn xác thực sẽ được lưu cùng với số liệu: khóa được dùng để tạo và kiểm tra dấu xác thực được lưu một cách tách bạch trong một "vùng" an toàn.

Về nhiều khía cạnh mã xác thực cũng tương tự như một sơ đồ chữ kí hoặc tương tự như một mã xác thực thông báo (MAC). Sự khác biệt chính là sự an toàn của một mã xác thực là không điều kiện trong khi sơ đồ chữ kí và MAC lại được nghiên cứu theo quan điểm độ an toàn tính toán. Cũng vậy, khi một xác thực (hoặc MAC) được dùng, một bản tin chỉ có thể được kiểm tra bởi người nhận hợp pháp. Trong khi đó bất cứ ai cũng có thể xác minh được chữ kí, bằng cách dùng một thuật toán xác minh công khai.

Định nghĩa 3.3.1

Mã xác thực là một bộ 4 (S, A, K, E) thoả mãn các điều kiện sau :

1. S là tập hữu hạn các trạng thái nguồn có thể.
2. A là tập hợp các nhãn xác thực có thể.
3. K là một tập hữu hạn các khoá có thể (không gian khoá).
4. Với mỗi $k \in K$ có một quy tắc xác thực $e_k : S \rightarrow A$

   Tập bản tin được xác định bằng $M = S \rightarrow A$

3.3.2. Ý tưởng chính của phương pháp xác thực bằng mã xác thực

Chú ý một trạng thái nguồn tương đương với một bản rõ. Một bản tin gồm bản rõ với một nhãn xác thực kèm theo, một cách chính xác hơn có thể coi đó là một bản tin đã được xác nhận. Một quy tắc xác thực không nhất thiết phải là hàm đơn ánh.

Để phát thông báo (đã được kí). An và Thu phải tuân theo giao thức sau. Trước tiên họ phải chọn một khoá ngẫu nhiên $k \in K$. Điều này được thực hiện một cách bí mật như trong hệ mật khoá bí mật. Sau đó giả sử rằng An muốn gửi một trạng thái nguồn $s \in S$ cho Thu trên kênh không an toàn, An sẽ tính $a = e_k(s)$ và gửi bản tin (s, a) cho Thu. Khi nhận được (s, a) Thu tính $a' = e_K(s)$. Nếu $a = a'$ thì Thu chấp nhận bản tin là xác thực, ngược lại Thu sẽ loại bỏ nó.

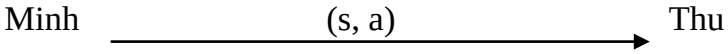
3.3.3. Phương pháp

Ta sẽ nghiên cứu hai kiểu tấn công khác nhau mà Minh có thể tiến hành. Trong cả hai loại này, Minh sẽ là "kẻ xâm nhập vào giữa cuộc".

Giả mạo

Minh đưa ra bản tin (s, a) vào kênh, và hi vọng nó sẽ được chấp nhận. Phương pháp này được mô tả trong hình 3.3.1.

Hình 3.3.1. Việc giả mạo bởi Minh



Thay thế

Minh quan sát một bản tin (s, a) trên kênh, sau đó anh ta biến đổi nó thành (s‟, a‟), trong đó s‟ = s và hi vọng được Thu chấp nhận như một bản tin xác thực. Bởi vậy anh ta tin sẽ lái được Thu đi tới trạng thái nguồn mới này. Phương pháp này được mô tả như hình 3.3.2

Hình 3.3.2. Phép thay thế của Minh

Ví dụ 3.3.1

Giả sử $K=A=Z$

và $K=Z_3 \times Z_3$

Với mỗi $(i, j) \in K$ và mỗi $s \in S$, xác định

$$e_k(s) = (i.s + j) \bmod 3$$

An gửi cho Thu bản rõ s và khóa k, đồng thời gửi ma trận xác thực (ma trận này tạo bằng tất cả các giá trị $e_k(s)$). Với mỗi khoá $k \in K$ và với mỗi $s \in S$ ta đặt nhãn xác thực $e_k(s)$ vào hàng k và cột s của một ma trận M kích thước K xác suất. Mảng M được mô tả trên hình 3.3.3

Hình 3.3.3. Ma trận xác thực

Khóa \ Bản rõ	0	1	2
(0,0)	0	0	0
(0,1)	1	1	1
(0,2)	2	2	2
(1,0)	0	1	2
(1,1)	1	2	0
(1,2)	2	0	1
(2,0)	0	1	2
(2,1)	1	0	2
(2,2)	2	1	0

Với khóa k = (i, j) = (0, 0) và bản rõ s = 0

Ta có mã xác thực $e_k(s)$ = (i.s + j) mod 3 = (0.0 + 0) mod 3= 0

Khi nhận được bản rõ s và khóa k (khóa bí mật chỉ An và Thu biết) Thu sẽ tiến hành lập ma trận xác thực theo công thức $e_k(s)$ = (i.s + j) mod 3 ((i, j) $\in$ *K* và *s* $\in$ *S*). Sau đó tiến hành so sánh với ma trận mà An đã gửi, nếu trùng khớp thì xác thực đúng, nếu tồn tại một nhãn e_k nào đó không khớp với ma trận xác thực An đã gửI chứng tỏ đã có sự giả mạo hoặc thay thế thông tin trên đường truyền.

**** Tấn công bằng phương pháp "Giả mạo"***

Trước tiên xét cách tấn công giả mạo, Minh sẽ chọn ra một trạng thái nguồn s và cố gắng phỏng đoán một nhãn xác thực "đúng"". Kí hiệu k_0 là khoá đang sử dụng (mà Minh không biết). Minh sẽ thành công trong việc đánh lừa Thu nếu anh ta phỏng đoán $a_0 = e_{k_0}(s)$. Tuy nhiên với bất kì s $\in$ S và a $\in$ A ta thấy rằng, chỉ có đúng 3 (chứ không phải là 9) quy tắc xác thực k $\in$ *K* sao cho $e_k(s)$ =a. Nói cách khác mỗi kí hiệu chỉ xuất hiện 3 lần trong mỗi cột của ma trận xác thực. Bởi vậy dẫn tới Pd_0=1/3.

**** Tấn công bằng phương pháp "Thay thế"***

Giả sử Minh đã quan sát được trên kênh 1 bản tin (0,0). Nhờ đó anh ta đã biết một thông tin nào đó về khoá, anh ta biết rằng :

$$k_0 \in \{(0, 0), (1, 0), (2, 0)\}$$

Bây giờ, giả sử Minh thay bản tin (0,0) bằng bản tin (1,1). Khi đó anh ta sẽ lừa bịp thành công khi và chỉ khi $k_0=(0, 1)$, xác suất để k_0 là khoá bằng 1/3 vì khoá nằm trong tập {(0, 0), (1, 0), (2, 0)}.

Có thể thực hiện một phân tích tương tự đối với bất kì một phép thay thế nào mà Minh tiến hành. Nói chung nếu Minh quan sát một bản tin (s, a) và thấy nó bằng một bản tin bất kì (s‟, a‟) trong đó $s' \neq s$ thì anh ta sẽ đánh lừa được Thu với xác suất 1/3. Ta có thể thấy rõ điều này như sau. Việc quan sát được (s, a) sẽ hạn chế khóa và một trong ba khả năng. Trong khi đó với một phép chọn (s‟, a‟) chỉ có một khoá chứ không phải ba khoá có thể theo quy tắc a là nhãn xác thực của s‟.

KẾT LUẬN

+ Việc đòi hỏi an toàn trong giao dịch cũng như trao đổi thông điệp được đặt lên hàng đầu vì vậy việc xác thực thông điệp là một vấn đề rất quan trọng trong giao dịch hiện nay, đặc biệt là trong giao dịch trực tuyến. Khi nhận được một thông điệp như thư, hợp đồng, đề nghị,…vấn đề đặt ra là làm sao để xác định được đúng đối tác giao dịch và nguồn gốc của thông điệp. Vì vậy báo cáo này nghiên cứu một số phương pháp xác thực thông điệp.

+ Kết quả chính của báo cáo là: Tìm hiểu và nghiên cứu qua tài liệu để hệ thống các vấn đề sau:

1/. Trình bày tổng quan về xác thực điện tử.

2/. Trình bày một số phương pháp xác thực thông điệp.