TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN HỆ THỐNG THÔNG TIN

Khuất Thị Ngọc Bích -- Lê Thị Trúc Lâm

Tìm hiểu và phát triển cơ chế bảo mật trên mã nguồn mở của mySQL

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN HỆ THỐNG THÔNG TIN

Khuất Thị Ngọc Bích -0112046
Lê Thị Trúc Lâm -0112101

Tìm hiểu và phát triển cơ chế bảo mật trên mã nguồn mở của mySQL

LUẬN VĂN CỬ NHÂN TIN HỌC

GIÁO VIÊN HƯỚNG DẪN:

Th.S: PHẠM THỊ BẠCH HUỆ

NIÊN KHOÁ: 2001-2005

GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

LỜI CẢM ƠN

Chúng em xin chân thành cám ơn Khoa Công Nghệ Thông Tin, trường Đại Học Khoa Học Tự Nhiên, Đại học Quốc gia Tp. Hồ Chí Minh đã tạo điều kiện thuận lợi cho chúng em thực hiện đề tài luận văn tốt nghiệp này.

Chúng em xin nói lên lòng biết ơn sâu sắc đối với ThS. Phạm Thị Bạch Huệ. Xin chân thành cám ơn Cô đã luôn quan tâm, tận tình hướng dẫn em trong quá trình học tập, nghiên cứu và thực hiện đề tài.

Em xin chân thành cám ơn quý Thầy Cô trong Khoa Công Nghệ Thông Tin đã tận tình giảng dạy, trang bị cho em những kiến thức quý báu, đã truyền thụ cho em những kiến thức, kinh nghiệm, đã quan tâm dìu dắt và giúp đỡ em trong quá trình học tập cũng như trong lúc thực hiện đề tài này.

Chúng con luôn nhớ mãi công ơn của Ông Bà, Cha Mẹ đã luôn thương yêu, lo lắng, chăm sóc và nuôi dạy con thành người.

Cuối cùng chúng em xin gửi lời cám ơn đến các anh chị, các bạn đã quan tâm động viên và giúp đỡ chúng em trong quá trình thực hiện đề tài. Đặc biệt, chúng em xin gửi lời cảm ơn chân thành đến thầy Nguyễn Đình Thúc, chị Trần Hồng Ngọc, chị Trương Thị Mỹ Trang đã động viên, giúp đỡ chúng em trong thời gian thực hiện đề tài.

Mặc dù đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong nhận được sự tận tình chỉ bảo của quý Thầy Cô và các bạn.

Một lần nữa, chúng em xin chân thành cám ơn và mong luôn nhận được những tình cảm chân thành của tất cả mọi người.

Tp. Hồ Chí Minh, tháng 6 năm 2005

Khuất Thị Ngọc Bích – Lê Thị Trúc Lâm

MỤC LỤC

Danh sách các bảng

Danh sách các hình

Danh sách các từ viết tắt

BM	bảo mật
csdl	cơ sở dữ liệu
HQTCSDL	hệ quản trị cơ sở dữ liệu
CNTT	công nghệ thông tin
PMNM	phần mềm nguồn mở
SSL	Secure Sockets Layer

Tóm tắt nội dung của luận văn

- **Chương 1** : Trình bày tổng quan về bảo mật dữ liệu, các chiều hướng bảo mật thông tin hiện tại, các yêu cầu trong bảo mật dữ liệu, tình hình nghiên cứu hiện nay trên thế giới và trong nước về lĩnh vực này, đồng thời nêu lên mục đích, nội dung và ý nghĩa của đề tài.
- **Chương 2** : Trình bày tóm tắt một số phương pháp mã hoá hiện nay, phân loại cũng như đánh giá ưu khuyết điểm của từng phương pháp. Đặc biệt, trong chương này sẽ giới thiệu khá kĩ về hàm BĂM, tạo cơ sở tiền đề để ta nghiên cứu ở các chương sau.
- **Chương 3** : Trình bày khái quát về hệ quản trị cơ sở dữ liệu MySQL. Trong chương này, cơ chế bảo mật của MySQL sẽ được trình bày cụ thể. Cơ chế bảo mật trong môi trường mạng, trong cơ sở dữ liệu cũng như cách lưu trữ password, quá trình kiểm tra password, cơ chế phân quyền sẽ được trình bày trong chương này.
- **Chương 4** : Trình bày về thuật toán bảo mật password trong MySQL : ý tưởng, các bước của thuật toán, đánh giá ưu khuyết điểm. Đồng thời, chúng tôi sẽ đề xuất một số thuật toán tốt hơn có thể hạn chế được khuyết điểm của thuật toán hiện tại.
- **Chương 5 :** Giới thiệu sơ lược mã nguồn mở của MySQL, tìm hiểu các hàm mã hoá password, hàm lưu password trong CSDL. Đồng thời cài đặt thử nghiệm một số thuật toán mới và ứng dụng phát triển cơ chế bảo mật của HQTCSDL MySQL.
- **Chương 6** : Kết luận và hướng phát triển của đề tài.

Chương 1.Giới thiệu

✍ *Nội dung của chương 1 trình bày tổng quan về xu hướng mã nguồn mở và chiều hướng bảo mật thông tin hiện tại, các yêu cầu trong bảo mật dữ liệu, tình hình nghiên cứu hiện nay trên thế giới và trong nước về lĩnh vực này, đồng thời nêu lên mục đích, nội dung và ý nghĩa của đề tài.*

1.1. Tổng quan bảo mật

1.1.1. Nhận định về bảo mật

Bảo mật thông tin ảnh hưởng rất lớn đến sự tồn tại và phát triển của doanh nghiệp hay tổ chức. Do đó việc bảo mật thông tin bảo mật thông tin có ý nghĩa hết sức quan trọng.

Ngày nay với sự phát triển không ngừng của công nghệ thông tin, dữ liệu của các doanh, nghiệp, tổ chức, các nhân không chỉ lưu trữ trên giấy tờ mà được tổ chức thành một cơ sở dữ liệu (csdl). Csdl sẽ được quản lý bằng một hệ quản trị cơ sở dữ liệu (hqtcsdl). Tùy thuộc vào độ lớn của csdl, tình hình tài chính, khả năng của mình mà mỗi công ty sẽ chọn lựa một hệ quản trị phù hợp.

Ngày nay, sự phát triển của internet giúp cho các giao dịch trên mạng ngày càng tăng, lượng thông tin trao đổi trên mạng cũng tăng tương ứng. Vì thế cơ hội cho các đối thủ, các "hacker" thâm nhập đánh cắp dữ liệu càng tăng.

Thông tin cần được bảo vệ khỏi các mối đe dọa như :

- Việc mạo danh truy cập thông tin bất hợp pháp và sử dụng thông tin cho các mục đích riêng của mình.

- Sự tấn công của các hacker vào các dữ liệu nhằm phá hoại dữ liệu để phục vụ cho các mục đích riêng như cạnh tranh không lành mạnh giữa các doanh nghiệp.
- Các thông tin nhạy cảm có thể bị lấy trộm.

Thông tin cần được bảo vệ ngay trong csdl, trên đường mạng để tránh bị đánh cắp, bị thay đổi.

Các nhận định sai lầm về "mất mát thông tin" :

- Thông tin bị mất thường là do các hacker bên ngoài đột nhập vào và đánh cắp. Nhưng thực tế thì 80% dữ liệu bị mất là do những người bên trong hệ thống gây ra.
- Chỉ cần mã hóa dữ liệu là đã đủ khả năng bảo mật dữ liệu. Thực chất mã hóa chỉ là một bước trong một cơ chế bảo mật mà thôi.

Do đó, ta cần phải có cơ chế bảo mật thích hợp cho hệ thống. Một cơ chế bảo mật thường gồm các vấn đề sau :

- Mã hóa dữ liệu.
- Cách thức trao đổi thông tin.
- Cách thức lưu trữ thông tin.
- Các phương pháp chứng thực user.
- Cách thức nhận biết quyền hợp pháp của user đối với thông tin dữ liệu.

1.1.2. Các chiều hướng bảo mật thông tin :

Thông tin thường được lưu tại các server và được tổ chức thành các file vật lý có cấu trúc và được quản trị bằng một hệ quản trị cơ sở dữ liệu thích hợp. Thông tin được truyền trên các đường mạng sẽ được bảo vệ theo các cơ chế riêng.

Dù ở trong hình thức nào thì thông tin đều cần được bảo mật theo các chiều hướng sau:

Hướng bảo mật	Nguyên tắc bảo mật
Vật lý	User chưa được chứng thực thì không được phép truy cập vào máy ở mức vật lý.
Cá nhân	Quản trị viên có trách nhiệm quản trị và bảo mật dữ liệu trong hệ thống. Do đó quản trị viên phải là người đáng tin cậy, có tư cách về đạo đức.
Thủ tục	Các thủ tục dùng trong hệ thống phải dùng đúng các dữ liệu theo đúng chức năng của mình. Ví dụ một người thực hiện công việc back up dữ liệu thì nhiệm vụ duy nhất của người đó là đảm bảo dữ liệu back up và running. Một người chịu trách nhiệm thực hiện tạo các báo cáo về bảng lương và bán hàng thì người đó chỉ có nhiệm vụ kiểm tra và xác nhận tính toàn vẹn của dữ liệu. Vì thế cách quản lý khéo léo nhất là phân chia nhiệm vụ, vai trò cho từng user theo đúng phạm vi chức năng phận sự.
Kỹ thuật	Lưu trữ, truy cập, sử dụng và truyền dữ liệu phải được an toàn bằng những kỹ thuật thi hành theo những chính sách phù hợp.

Bảng 1.1.2 Các chiều hướng bảo mật

Vậy khi đưa ra một giải pháp nào, ta cần phải cân nhắc thật cẩn thận về vấn đề bảo mật. Tuy nhiên, có một số trường hợp mà vấn đề kỹ thuật không thể giải quyết được. Đó là các vấn đề về "bảo mật trong môi trường làm việc". Ví dụ, một nhân viên rời khỏi bàn làm việc của mình trong một lát vì một lý do nào đó và một người khác lợi dụng để xâm nhập đánh cắp hoặc thay đổi dữ liệu.

1.1.3. Bảo mật thông tin

1.1.3.1. Bảo mật thông tin truyền trong môi trường mạng

Khi dữ liệu truyền trên đường mạng thông tin có thể bị lấy mất bất cứ lúc nào. Nếu dữ liệu truyền đi mà không có phương pháp nào bảo mật thì kẻ xấu dễ dàng lấy được thông tin và dùng nó vào những mục đích riêng của mình. Do đó cần phải có những phương pháp bảo mật dữ liệu trên mạng.

1.1.3.2. Bảo mật thông tin CSDL

a) Bảo mật CDSL gồm có các tiêu chuẩn sau :

- Bí mật

  Hệ thống chỉ cho phép mỗi user khi đăng nhập thành công chỉ được thực hiện các thao tác mà user đó có đủ các quyền để thực thi thao tác.

- Toàn vẹn

  Dữ liệu phải được bảo toàn, không bị xóa lỗi.

- Sẵn sàng

  Dữ liệu phải luôn sẵn sàng để phục vụ không được chậm trễ.

Bảo mật csdl là chỉ ra ai là người được truy cập vào dữ liệu, user được thấy những dữ liệu nào của csdl, user có thể thực hiện các thao tác nào trên csdl, user có thể xem các dữ liệu nhạy cảm khi cần thiết hay không ?

b) Bảo mật username và password

Dữ liệu trong csdl trên server luôn cần được bảo mật và chỉ có một số người có chức năng mới được phép truy cập và sử dụng. Để chứng thực một user thì phương pháp thường thấy nhất là dùng một định danh username và password. Tuy nhiên, username và password có thể bị đánh cắp bất cứ lúc nào.

c) Sự truy cập bất hợp pháp vào dữ liệu

Trong csdl thì không phải bất kỳ một user nào cũng có quyền truy cập và thực hiên các thao tác như nhau. Tùy theo mỗi chức vụ, công việc, phạm vi thực hiện của mỗi user mà họ có quyền và có thể thực hiện một số thao tác khác nhau trên csdl. Đó chính là việc phân quyền cho user.

Ủy quyền là công việc trao cho user, program hay process quyền được truy cập thực thể hoặc tập các thực thể. Các quyền này có thể là chỉ là read hay read/write.

Quyền hạn là sự cho phép truy cập mang tính thi hành, ví dụ như quyền được truy vấn trên table. Quyền hạn được cấp cho user theo quyết định của user cấp cao hơn (thường là quản trị viên Administrator). Quyền hạn được cấp cho user hợp lệ để kết nối csdl, thao tác trên csdl. Có 2 mảng quyền chính :

- ✓ System Privileges

  Đây là quyền cấp cao. Thường thì các quyền này chỉ được cấp cho quản trị viên và các người phát triển ứng dụng. Quyền này cho phép user được phép thao tác trên toàn bộ csdl và được phép cấp quyền cho các user khác.

- ✓ Object Privileges

  Đây là các quyền thao tác trên các đối tượng của csdl như database, table, row, column. Các quyền này bao gồm các thao tác INSERT, UPDATE, DELETE, SELECT, CREATE …

1.2. Tình hình an toàn và bảo mật trên thế giới và ở Việt Nam

1.2.1. Trên thế giới : nhu cầu đang gia tăng

Theo kết quả khảo sát do Viện An ninh Máy tính (CSI) phối hợp với Cục điều tra Liên bang Mỹ (FBI) thực hiện về chủ đề tội phạm và an ninh mạng, các vụ đánh cắp thông tin mật gây thiệt hại lớn nhất là 2,7 triệu USD mỗi vụ. Còn theo tờ Computer Economics, trong năm 2003 các loại sâu và virus máy tính đã gây thiệt hại 12,5 tỉ USD trên toàn cầu. Trong an ninh mạng, các doanh nghiệp vừa và nhỏ (SMB) dễ trở thành nạn nhân của các vụ tấn công nhất, bởi đối tượng này thiếu nguồn lực và đội ngũ chuyên gia công nghệ thông tin. Ngay tại nước Mỹ, theo số liệu thống kê, chỉ 35% các doanh nghiệp vừa và nhỏ là có sử dụng hệ thống tường lửa (firewalls).

Cũng giống như môi trường an ninh nói chung, môi trường an ninh trên Internet đang ngày càng trở nên phức tạp. Vấn đề bảo mật hệ thống và song hành với nó là vấn đề lưu trữ thông tin đang đóng vai trò ngày càng quan trọng. Theo nhóm nghiên cứu thị trường Meta Group:

- Hiện tại chỉ có khoảng 3-4% ngân sách CNTT dành cho vấn đề bảo mật và an toàn thông tin, nhưng theo dự báo đến năm 2006 tỷ lệ này sẽ tăng lên 8-10%.
- Thị trường an ninh CNTT Châu á dự tính cũng sẽ đạt mức tăng trưởng 22% từ năm 2003 đến năm 2008, con số gấp gần 2 lần tỷ lệ tăng trưởng của thị trường dịch vụ CNTT nói chung.
- Còn theo số liệu từ hãng nghiên cứu thị trường IDC, thị trường an ninh, bảo mật Châu á-Thái Bình Dương (trừ Nhật Bản) cũng sẽ tăng trưởng

15% từ năm 2002 đến năm 2007 và sẽ đạt tổng giá trị 4,1 tỉ USD vào năm 2007**.**

1.2.2. Ở Việt Nam

Chưa bao giờ vấn đề bảo mật và an toàn dữ liệu lại được coi trọng như hiện nay, trong bối cảnh mạng máy tính phá bỏ mọi ngăn cách, "mọi lúc, mọi nơi" người ta đều có thể lấy được thông tin cần thiết. Thông tin đã trở thành một trong những nguồn tài nguyên quan trọng nhất với tổ chức, doanh nghiệp. Con người tập trung nhiều sức lực, trí tuệ để có thông tin nhanh, chính xác. Ai có thông tin, kẻ đó chiến thắng. Bởi vậy, thông tin đã trở thành mục tiêu săn đuổi của những ai muốn vượt lên, và đồng thời là cái mà ai cũng cố gắng giữ.

Với sự phát triển của CNTT, hầu như mọi thứ đều được "số hóa", đặc biệt là thông tin. Soạn thảo hợp đồng bằng Word, gửi thư qua e-mail, thanh toán với ngân hàng bằng thẻ tín dụng ...; nói chung mọi người làm việc, giao dịch đều qua máy tính và mạng. Ta không thể làm khác đi bởi sẽ bị cô lập, sẽ luôn chậm hơn, mất khả năng cạnh tranh và cuối cùng sẽ thua cuộc.

Dù nằm trong máy tính hay két sắt thì dữ liệu của người dùng vẫn là mục tiêu nhắm tới của các đối thủ cạnh tranh. Trong trường hợp này, "tin tặc" là những tay đáng ngại nhất. Người dùng phải biết cách phòng chống.

Tại VN, vấn đề BM hệ thống thông tin bắt đầu nóng dần lên và đang sẵn sàng cho nhu cầu BM từ quy mô nhỏ cho đến lớn.

1.3. Xu hướng mã nguồn mở

1.3.1. Lợi ích của phần mềm mã nguồn mở (PMNM)

Đặc tính chia sẻ mã nguồn khiến PMNM có vai trò thực sự quan trọng trong lĩnh vực đào tạo và nghiên cứu. Những thư viện mã nguồn mở sẽ giúp sinh viên hiểu rõ và nhanh chóng nắm bắt được công nghệ, rút ngắn được thời gian đào tạo sinh viên CNTT.

Không chỉ trong lĩnh vực giáo dục, đào tạo hay nghiên cứu, điều đáng ngạc nhiên là PMNM cũng hứa hẹn những cơ hội kinh doanh không nhỏ đối với các doanh nghiệp, những người luôn đặt vấn đề lợi ích lên hàng đầu. Cơ hội kinh doanh mà PMNM mang lại không nhỏ hơn những cơ hội kinh doanh dựa trên nền tảng của Microsoft Windows.

1.3.2. Việt Nam

PMNM đã từng được ví như lối thoát hiểm của Việt Nam trước áp lực về bản quyền sở hữu trí tuệ trong quá trình hội nhập quốc tế. Khi nước nhà chuẩn bị gia nhập Tổ chức Thương mại Thế giới WTO, Khu vực Mậu dịch Tự do (AFTA) và thực hiện Hiệp định Thương mại Việt-Mỹ thì PMNM là đường thoát hiểm duy nhất để thoát khỏi tình trạng vi phạm bản quyền phần mềm ở Việt Nam

Hội thảo quốc gia lần thứ nhất về PMNM được tổ chức tháng 12/2000 có thể được xem như một cột mốc đánh dấu sự xuất hiện chính thức của PMNM tại Việt Nam. Hai năm sau đó, Hội thảo Quốc gia về PMNM lần thứ hai, tháng 12/2002, được coi là bước chuẩn bị và nâng cao nhận thức về PMNM. Chính tại Hội thảo này đã cho thấy PMNM đang là một xu hướng phát triển trên thế giới :

- Các tổ chức quốc tế đều khuyến cáo sử dụng PMNM.

- Các nước Châu Á như Trung Quốc, Hàn Quốc, Nhật Bản đang phát triển rất mạnh PMNM. Malaysia gần đây đã đầu tư 30 triệu USD cho PMNM. Năm 2003 Nhật Bản cũng dành 10 triệu USD cho PMNM.
- Không chỉ ở cấp Chính phủ, nhiều công ty đa quốc gia như Oracle, IBM, HP... cũng đang phát triển mạnh theo xu hướng PMNM.
- IBM hiện có một trung tâm với 700 người chuyên nghiên cứu về PMNM.

Quyết định số 235/QĐ-TTg, ngày 2/3/2004, của Thủ tướng Chính phủ phê duyệt Dự án tổng thể "ứng dụng và phát triển phần mềm nguồn mở ở Việt Nam giai đoạn 2004-2008" là điểm mốc đánh dấu việc bắt đầu triển khai PMNM tại Việt Nam.

1.4. Mục tiêu của đề tài

Vấn đề bảo mật hệ thống và song hành với nó là vấn đề lưu trữ thông tin đang đóng vai trò ngày càng quan trọng. Đối với một tổ chức hay cá nhân khi lựa chọn một hệ quản trị CSDL, ngoài tiêu chí chọn hệ quản trị có quy mô phù hợp với độ lớn của CSDL thì vấn đề bảo mật của hệ quản trị đó cũng rất được quan tâm.

Đề tài "Tìm hiểu và phát triển cơ chế bảo mật trên mã nguồn mở của MySQL" được thực hiện nhằm mục tiêu:

- Tìm hiểu các cơ sở lý thuyết về bảo mật, giới thiệu tóm tắt một số phương pháp mã hoá
- Tìm hiểu cơ chế bảo mật của một hệ quản trị mã nguồn mở : MySQL.
- Tìm hiểu, phân tích, đánh giá thuật toán mã hoá password trong MySQL
- Trên cơ sở nghiên cứu một số giải thuật mới, có độ an toàn cao, xây dựng một số cơ chế mã hoá password mới của riêng mình trong MySQL.

Chương 2.Các cơ sở lý thuyết bảo mật

✍ Nội dung của chương 2 sẽ trình bày tóm tắt một số phương pháp mã hoá hiện nay, phân loại cũng như đánh giá ưu khuyết điểm của từng phương pháp. Đặc biệt, trong chương này sẽ giới thiệu khá kĩ về hàm BĂM, tạo cơ sở tiền đề để ta nghiên cứu ở các chương sau.

Thuật toán mã hóa dữ liệu hiện nay phân loại theo số khoá được dùng để mã hoá và giải mã có 3 loại :

1. Secret Key Cryptography (SKC) : sử dụng một khoá chung cho quá trình mã hoá và giải mã.
2. Public Key Cryptography (PKC) : sử dụng một khoá cho phần mã hoá và một khoá khác để giải mã.
3. Hash Functions: sử dụng một phép biến đổi mã hóa thông tin một chiều. Điều này có nghĩa là một khi thông tin đã được mã hóa thì không thể có cách nào để lấy lại được thông tin ban đầu.

2.1. Secret Key Cryptography(Hệ Mã hoá quy ước)

2.1.1. Giới thiệu

Hình 2.1.1 Secret Key Cryptography

Các thuật toán mã hoá quy ước (hay mã khoá bí mật hay hệ mã đối xứng) dùng một khoá bí mật đơn để mã hoá và giải mã dữ liệu. Dữ liệu nguồn x được người gởi A mã hoá bằng thuật toán mã hoá quy ước với khoá bí mật k được thống nhất trước giữa người gởi A và người nhận B. Dữ liệu sau khi mã hoá y sẽ

được truyền cho người nhận B. Sau khi nhận, B sẽ sử dụng khoá bí mật k để giải mã y để có được thông điệp nguồn x ban đầu.

Nếu một người C có được khoá bí mật k thì C sẽ có khả năng giải mã tất cả dữ liệu của A bằng khoá k rồi thay đổi dữ liệu và mã hóa lại bằng khóa k sau đó gởi cho B. Do đó vấn đề bảo mật thông tin được mã hoá phụ thuộc vào việc giữ bí mật nội dung mã khoá k.

Mã hoá khoá bí mật cũng được gọi là mã hoá khoá đối xứng vì chỉ dùng một khoá cho mã hoá lẫn giải mã. Thuật toán mã hoá này có tốc độ cực nhanh và thích hợp đối với việc mã hoá khối lượng dữ liệu lớn.

2.1.2. Phân loại thuật toán

2.1.2.1. Mã hóa theo chuỗi bit

Trong hệ mã hoá theo chuỗi bit, thông điệp là các bit và khoá được phát sinh bởi một bộ phát sinh ngẫu nhiên. Bảng rõ mã hoá theo từng bước để được bản mã.

2.1.2.2. Mã hóa theo chữ

Các hệ mã ban đầu dựa trên cơ sở phép biến đổi một chữ cái trong bảng rõ thành một chữ cái khác trong bảng mã. Kỹ thuật mã hoá này còn được gọi là mã hoá thay thế. Để thực hiện phương pháp này, trước tiên cần định nghĩa 1 bảng mã (như bảng mã ASCII) để số hoá bảng rõ, vì các phép toán sẽ làm việc trên các số thay vì các kí tự.

2.1.2.3. Mã hóa theo khối

Ta thấy, hệ mã hoá theo chữ có độ an toàn không cao vì một chữ cái luôn được mã hoá thành 1 chữ cái khác trong bảng mã. Với khả năng của máy tính

hiện đại, không khó để có thể giải mã 1 bảng mã theo chữ như thế. Để tăng độ an toàn, ta có thể mã hoá theo khối. Trong mã hoá theo khối, bản rõ và bảng mã được chia thành từng khối kí tự trước khi thi hành mã hoá và giải mã.

2.1.2.4. Mã mũ

Do Pohlig và Hellman giới thiệu năm 1976. Có thể được mô tả như sau : Chọn p là 1 số nguyên tố, M là 1 số tương ứng của bản rõ với mỗi kí tự trong bảng rõ được thay thế bằng mã tương ứng như trong bảng.

''	A	B	C	D	E	F	G	H	I	J	K
00	01	02	03	04	05	06	07	08	09	10	11

2.1.3. Một vài thuật toán SKC được sử dụng ngày nay

Những thuật toán **SKC** được sử dụng ngày nay : Data Encryption Standard (DES), Triple-DES (3DES), DESX, RC1,RC2, RC3, RC4, RC5, RC6, Blowfish, Twofish, Camellia, MISTY1, SAFER, KASUMI, SkipJack.

2.1.4. Đánh giá phương pháp mã hóa quy ước

Mặc dù hệ thống mã hoá quy ước cung cấp khá nhiều thuật toán mã hoá có độ bảo mật rất cao nhưng nó có các hạn chế sau :

- Hạn chế về khả năng trao đổi khoá : do cả người nhận và người gởi đều cần phải biết khoá nên phát sinh vấn đề an toàn khi truyền khoá. Nếu khoá bị đánh cắp trong quá trình truyền khoá thì thông tin được mã hoá bằng khoá đó không còn được bảo mật và an toàn. Ngoài ra với mã hoá quy ước không đảm bảo nguồn gốc thông tin được gởi nên không biết kháo có bị mật cắp hay không.

- ❖ Hạn chế khả năng quản lý khoá : đối với từng người cần liên lạc và với từng nội dung thông tin cần phải có một khoá quy ước để mã hoá và giải mã. Do đó nếu trên 1 mạng liên lạc lớn, số lượng khoá cần phải lưu giữ rất nhiều nên nảy sinh vấn đề quản lý khoá quy ước và bảo mật thiết bị khoá quy ước.

2.2. Public Key Crytography (Mã hoá công khai)

2.2.1. Giới thiệu chung

Hình 2.2.1 Public Key Crytography

Người gởi A sử dụng khoá công khai (hệ mã không đối xứng) pk của người nhận B để mã hoá dữ liệu gốc x. Dữ liệu sau khi được mã hoá, y được truyền cho B. Người nhận B sau khi nhận được y sẽ sử dụng khoá riêng sk của mình để giải mã dữ liệu và nhận lại dữ liệu nguồn x ban đầu.

Nếu 1 người C có được dữ liệu đã mã hoá y và khoá công khai pk thì C vẫn không thể giải mã được y. Do khoá riêng sk được giữ bí mật hoàn toàn, chỉ có Người B biết được sk và sk không được giao dịch hay truyền đi nên rủi ro dẫn đến việc khoá sk bị đánh cắp là rất thấp.

Giới thiệu một số thuật toán : EEC, RSA

2.2.2. Đánh giá phương pháp mã hóa công khai

Hệ thống mã hóa khóa công khai ra đời đã giải quyết các hạn chế của mã hóa quy ước. Mã hóa khóa công khai sử dụng một cặp khóa, một khóa (thông thường

là khóa riêng) dùng để mã hóa và một khóa (khóa riêng) dùng để giải mã. Mã hóa khóa công khai giúp tránh bị tấn công khi trao đổi khóa do khóa để giải mã (khóa riêng) không cần phải truyền hoặc chia sẻ với người khác. Ngoài ra, mỗi người chỉ cần sở hữu một cặp khóa công khai – khóa riêng và người gởi thông tin chỉ cần giữ khóa công khai của người nhận do đó số lượng khóa cần phải quản lý giảm khá nhiều. Mỗi người chỉ cần lưu trữ bảo mật một khóa riêng của chính mình.

Tuy nhiên, do nhu cầu mã hóa và giải mã bằng hai khóa khác nhau trong cùng một cặp khóa nên để đảm bảo bảo mật, kích thước khóa công khai – khóa riêng lớn hơn rất nhiều so với khóa công khai. Do đó tốc độ mã hóa khóa công khai chậm hơn tốc độ mã hóa khóa quy ước. Tốc độ mã hóa bằng phần mềm của thuật toán DES nhanh hơn khoảng 100 lần so với mã hóa RSA với cùng mức độ bảo mật.

	Kích thước khóa (tính bằng bit)					
Khóa công khai	56	80	112	128	192	256
RSA/DSA	512	1K	2K	3K	7.5K	15K
ECC	160		224	256	384	512

Bảng 2.2.2 : Kích thước khóa giữa mã hóa quy ước và mã hóa khóa công khai với cùng mức độ bảo mật.

2.3. Hash Function (hàm Băm)

2.3.1. Giới thiệu hàm Băm

Hình 2.3.1 Hash Function

Hàm băm mật mã là hàm toán học chuyển đổi một thông điệp có độ dài bất kỳ thành một dãy bit có độ dài cố định (tùy thuộc vào thuật toán băm). Dãy bit này được gọi là thông điệp rút gọn (message digest) hay giá trị băm (hash value), đại diện cho thông điệp ban đầu.

Dễ dàng nhận thấy rằng hàm băm h không phải là một song ánh. Do đó, với thông điệp x bất kỳ, tồn tại thông điệp x' ≠ x sao cho h(x) = h(x'). Lúc này, ta nói rằng "có sự đụng độ xảy ra".

Một hàm băm h được gọi là an toàn (hay "ít bị đụng độ") khi không thể xác định được (bằng cách tính toán) cặp thông điệp x và x' thỏa mãn x ≠ x' và h(x) = h(x'). Trên thực tế, các thuật toán băm là hàm một chiều, do đó, rất khó để xây dựng lại thông điệp ban đầu từ thông điệp rút gọn.

Hàm băm giúp xác định được tính toàn vẹn dữ liệu của thông tin : mọi thay đổi, dù là rất nhỏ, trên thông điệp cho trước, ví dụ như đổi giá trị 1 bit, đều làm thay đổi thông điệp rút gọn tương ứng. Tính chất này hữu ích trong việc phát sinh, kiểm tra chữ ký điện tử, các đoạn mã chứng nhận thông điệp, phát sinh số ngẫu nhiên, tạo ra khóa cho quá trình mã hóa…

Hàm BĂM được dùng trong chữ ký điện tử. Một đặc tính cơ bản của hàm BĂM là việc tạo mã khóa thông điệp rất dễ nhưng việc phá mã để chuyển ngược mã thông điệp thành bản rõ ban đầu rất khó nếu không muốn nói là không thể.

2.3.2. Tính chất của hàm băm

2.3.2.1. Tính một chiều

Hàm băm được xem là hàm một chiều khi cho trước giá trị băm, không thể tái tạo lại thông điệp ban đầu, hay còn gọi là "tiền ảnh" ("pre-image").

Như vậy, trong trường hợp lý tưởng, cần phải thực hiện hàm băm cho khoảng 2^n thông điệp để tìm ra được "tiền ảnh" tương ứng với một giá trị băm.

Nếu tìm ra được một phương pháp tấn công cho phép xác định được "tiền ảnh" tương ứng với một giá trị băm cho trước thì thuật toán băm sẽ không còn an toàn nữa.

Cách tấn công nhằm tạo ra một thông điệp khác với thông điệp ban đầu nhưng có cùng giá trị băm gọi là tấn công "tiền ảnh thứ hai" (second pre-image attack).

2.3.2.2. Tính an toàn của hàm băm đối với hiện tượng đụng độ

Hàm băm được xem là an toàn đối với hiện tượng đụng độ khi rất khó tìm được hai thông điệp có cùng giá trị băm.

Nhận xét : Trong một tập hợp mà các phần tử mang một trong N giá trị cho trước với xác suất bằng nhau, chúng ta cần khoảng $\sqrt{N}$ phép thử ngẫu nhiên để tìm ra một cặp phần tử có cùng giá trị.

Như vậy, phương pháp hàm băm được xem là an toàn đối với hiện tượng đụng độ nếu chưa có phương pháp tấn công nào có thể tìm ra cặp thông điệp

có cùng giá trị hàm băm với số lượng tính toán ít hơn đáng kể so với ngưỡng $2^{n/2}$, với n là kích thước (tính bằng bit) của giá trị băm.

Phương pháp tấn công dựa vào đụng độ :

- Tìm ra 2 thông điệp có nội dung khác nhau nhưng cùng giá trị băm
- Ký trên một thông điệp, sau đó, người ký sẽ không thừa nhận đây là chữ ký của mình mà nói rằng mình đã ký trên một thông điệp khác.

Như vậy, cần phải chọn 2 thông điệp "đụng độ" với nhau trước khi ký.

2.3.3. Cấu trúc của hàm băm

Hầu hết các hàm băm mật mã đều có cấu trúc giải thuật như sau :

- Cho trước một thông điệp M có độ dài bất kỳ. Tùy theo thuật toán được sử dụng, chúng ta có thể cần bổ sung một số bit vào thông điệp này để nhận được thông điệp có độ dài là bội số của một hằng số cho trước. Chia nhỏ thông điệp thành từng khối có kích thước bằng nhau: M1, M2, …Ms
- Gọi H là trạng thái có kích thước n bit, f là "hàm nén" thực hiện thao tác trộn khối dữ liệu với trạng thái hiện hành
  - ✓ Khởi gán H0 bằng một vector khởi tạo nào đó
  - ✓ Hi = f(Hi-1, Mi) với i = 1, 2, 3, …, s

  Hs chính là thông điệp rút gọn của thông điệp M ban đầu

2.3.4. Giới thiệu một số hàm băm

Hàm băm là nền tảng cho nhiều ứng dụng mã hóa. Có nhiều thuật toán để thực hiện hàm băm, trong số đó, phương pháp SHA-1 và MD5 thường được sử dụng khá phổ biến từ thập niên 1990 đến nay.

Ngày 26/08/2002, Viện Tiêu chuẩn và Công nghệ quốc gia của Hoa Kỳ (National Institute of Standard and Technology - NIST) đã đề xuất hệ thống chuẩn hàm băm an toàn (Secure Hash Standard) gồm 4 thuật toán hàm băm SHA-1, SHA-256, SHA-384, SHA-512. Đến 25/03/2004, NIST đã chấp nhận thêm thuật toán hàm băm SHA-224 vào hệ thống chuẩn hàm băm.

Tiger ra đời năm 1996 bởi Ross Anderson và Eli Biham.

Hàm BĂM Whirlpool do Paul S.L.M Barreto và Vincent Rijment đề xuất năm 2001 và được công nhận tính bảo mật tại hội thảo bảo mật NESSIE tại Lund, Thụy Điển vào ngày 26/02/2003, đạt chuẩn ISO ISO/IEC 10118-3.

Chương 3.Hệ quản trị cơ sở dữ liệu MySQL

✍ *Trình bày khái quát về hệ quản trị cơ sở dữ liệu MySQL. Trong chương này, cơ chế bảo mật của MySQL sẽ được trình bày cụ thể. Cơ chế bảo mật trong môi trường mạng, trong cơ sở dữ liệu cũng như cách lưu trữ password, quá trình kiểm tra password, cơ chế phân quyền sẽ được trình bày trong chương này.*

3.1. Giới thiệu hệ quản trị cơ sở dữ liệu MySQL

3.1.1. Giới thiệu

- MySQL là 1 hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở phổ biến nhất hiện nay. Nó được phát triển, phân chia, hỗ trợ bởi MySQL AB. Đây là một công ty thương mại, được thành lập bởi nhóm phát triển MySQL.
- MySQL server nhanh, đáng ting cậy, dễ sử dụng. Sử dụng môi trường client/server độc lập hoặc nhúng vào các chương trình khác.
- Hiện nay MySQL được dùng trên 4 triệu bản cài đặt trên toàn thế giới.
- Các công ty lớn có thể truy cập mã nguồn MySQL để tạo ra một bản khác dùng nội bộ công ty với mục đích phi thương mại mà không sợ vi phạm bản quyền.
- Tốc độ MySQL khá nhanh, bảng của MySQL lên đến 8 GB đủ dùng cho các công ty không quá lớn.
- Các tính năng mạnh của MySQL đang được bổ sung dần qua từng phiên bản. Phiên bản 4.1 hiện tại không ít hơn tính năng là mấy so với SQL Server.

  Số nhà phát triển dùng MySQL là rất lớn trên thế giới cho nên số các ứng dụng chạy trên MySQL là nhiều và cũng miễn phí.

- Có một sự phân biệt khác rất quan trọng để người ta chọn MySQL, đó là tốc độ. Mặc dù MySQL xử lý transaction không tốt như Oracle, nhưng về tốc độ, nó chạy nhanh hơn Oracle rất nhiều. Cộng thêm mã nguồn mở và miễn phí, nó là giải pháp database tốt nhất cho các website, ngoại trừ các website lớn cần xử lý các tác vụ đặc biệt.

3.1.2. Bản chất

- Chương trình được viết bằng C và C++.
- Được kiểm tra bằng các trình biên dịch khác nhau.
- Làm việc trên nhiều nền khác nhau.
- Sử dụng Automake, Autoconf, và libtool tạo tính năng động.
- Các hàm API cho có giá trị trên C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, và Tcl.
- Những tiến trình nhân (kernel thread) sử dụng đa tiến trình nên có thể sử dụng nhiều CPU nếu người dùng có.
- Hệ thống cấp phát bộ nhớ dựa trên tiến trình rất nhanh.
- Có những bảng BAM bộ nhớ trong (in_memory) được sử dụng như những bảng tạm.
- Những hàm SQL thực thi nhanh do sử dụng các thư viện.
- Server có giá trị như 1 chương trình độc lập trong môi trường mạng client/server, cũng có giá trị như 1 thư viện có thể nhúng vào các chương trình ứng dụng độc lập.

3.1.3. Các kiểu dữ liệu

- Các kiểu dữ liệu :

  SIGNED/UNSIGNED INTEGERS 1, 2, 3, 4, and 8 BYTES LONG

  FLOAT

  DOUBLE

  CHAR

  VARCHAR

  TEXT

  BLOB

  DATE

  TIME

  DATETIME

  TIMESTAMP

  YEAR

  SET

  ENUM

- Fixed-length và variable-length records

3.1.4. Statement và function

- Hỗ trợ toàn bộ các toán tử và chức năng trong câu lệnh truy vấn SELECT và WHERE.

VD :

Mysql> SELECT CONCAT(first_name, ‘’, last_name)

→ FROM citizen

→ WHERE income/dependents > 10000 AND age > 30;

- Hỗ trợ các câu truy vấn SQL GROUP BY và ORDER BY. Ngoài ra còn hỗ trợ các hàm gom nhóm (COUNT(), COUNT(DISTNCT …), AVG(), STD(), SUM(), MAX(), MIN(), GROUP_CONCAT()).
- Hỗ trợ cho LEFT OUTER JOIN và RIGHT OUTER JOIN theo cú pháp chuẩn của SQL và ODBC.
- Hỗ trợ việc ghi các alias cho table, column.
- DELETE, INSERT, REPLACE, UPDATE trả về số dòng ảnh hưởng.
- Câu lệnh đặc biệt MySQL: SHOW có thể được sử dụng nhận thông tin về cơ sở dữ liệu, table, index. Câu lệnh EXPLAIN có thể được dùng chỉ định cách thức giải quyết thõa câu truy vấn.
- Tên các hàm không xung đột với tên các table và column.
- Người dùng có thể sửa chữa các table trong các database khác nhau trong cùng một câu truy vấn (như MySQL 3.22).

3.1.5. Bảo mật

Hệ thống đặc quyền và password rất linh hoạt, an toàn, cho phép xác nhận dựa trên máy chủ (host). Password được bảo mật vì tất cả các dòng di chuyển password được mã hoá khi user kết nối vào server.

3.1.6. Khả năng mở rộng và giới hạn

- Handles của database lớn. MySQL server với các database có thể chứa đến 50 triệu record. Ta có thể biết được các user sử dụng MySQL Server với 60.000 tables và khoảng 5.000.000.000 dòng.
- Có thể tạo 64 indexes trong 1 table (32 indexes đối với các phiên bản trước 4.1.2). Mỗi index có thể gồm từ 1 đến 16 cột hay các phần của cột.

Độ lớn tối thiểu của index là 1000 bytes (500 bytes đối với các phiên bản trước 4.1.2). Một index có thể sử dụng tiền tố (prefix) của cột có các kiểu dữ liệu CHAR, VARCHAR, BLOB, hay TEXT.

- Giới hạn các bảng trong MySQL

  MySQL 3.22 có giới hạn của table là 4GB. Với kỹ thuật lưu trữ MySIAM trong MySQL 3.23, kích thước tối đa của table được tăng 2^{63} byte.

Operating System	File-size Limit
Linux 2.2-Intel 32-bit	2GB (LFS: 4GB)
Linux 2.4	(using ext3 filesystem) 4TB
Solaris 9/10	16TB
NetWare w/NSS filesystem	8TB
win32 w/ FAT/FAT32	2GB/4GB
win32 w/ NTFS	2TB (possibly larger)
MacOS X w/ HFS+	2TB

Bảng 3.1.6 Kích thước giới hạn của file hệ thống trong MySQL

3.1.7. Kết nối

- Clients có thể kết nối với MySQL server bằng cách sử dụng những socket TCP/IP trên bất kì nền nào. Trên nền windows dòng NT, client có thể kết nối sử dụng pipes. Trên nền Unix, clients kết nối sử dụng Unix domain socket files.
- Windows có hỗ trợ kết nối chia sẻ bộ nhớ nếu khởi động với lựa chọn share-memory. Client có thể kết nối qua bộ nhớ chia sẻ bằng cách sử dụng lựa chọn --protocol=memory.

- Giao diện Connector/ODBC (MyODBC) cung cấp MySQL hỗ trợ các chương trình client sử dụng kết nối ODBC (Open Database Connectivity).
- Giao diện Connector/J interface cung cấp MySQL hỗ trợ chương trình client Java sử dụng kết nối JDBC.

3.1.8. Mức hạn định

- Có thể sử dụng cấp thông điệp lỗi cho client ở bất kì ngôn ngữ nào do có rất nhiều bộ kí tự.
- Dữ liệu được lưu trữ theo bộ kí tự đã chọn.

3.2. Cơ chế bảo mật trong MySQL

3.2.1. Tổng quan bảo mật

Bảo mật cơ sở dữ liệu là một yếu tố cần thiết trong bất kì hệ thống cơ sở dữ liệu nào. Bảo mật là bảo vệ dữ liệu chống lại các crackers, đồng thời cũng bảo vệ dữ liệu từ người dùng. Ví dụ, người dùng có thể xoá, sửa, xem hoặc thêm những record mà họ không được phép. Để tránh những tình trạng này, người dùng có thể tạo nên một cơ chế bảo mật trên cơ sở dữ liệu để ngăn chặn các thao tác mà user không được phép thao tác trên CSDL. Bảo mật đóng vai trò chính trong bất kì ứng dụng nào mà người dùng có thể truy cập vào ít nhất một lần. Cơ chế bảo mật của HQTCSDL MySQL rất tốt, là một trong các hệ quản trị cơ sở dữ liệu an toàn nhất hiện nay. Cơ chế bảo mật của HQTCSDL MySQL gồm hai lĩnh vực chính :

- Bảo mật trong môi trường mạng.
- Bảo mật cơ sở dữ liệu.

3.2.2. Bảo mật trong môi trường mạng

Bắt đầu từ phiên bản 4.0.0, MySQL hỗ trợ cho các kết nối an toàn giữa các MySQL client và server dùng giao thức Secure Sockets Layer (SSL).

Cấu hình chuẩn của MySQL có khuynh hướng sao cho chạy nhanh nhất có thể, nhưng việc mã hóa các kết nối không thể dùng mặc định. Làm như thế có thể sẽ làm cho client/server chậm hơn. Mã hoá dữ liệu là thao tác CPU-intensive mà đòi hỏi máy tính cần phải thực hiện thêm một số công việc và có thể làm trì hoãn các nhiệm vụ khác của MySQL. Đối với các ứng dụng mà đòi hỏi bảo mật được cung cấp bởi các kết nối mã hoá, sự tính toán thêm vào là được cho phép.

3.2.3. Các khái niệm cơ bản

Để hiểu MySQL dùng SSL như thế nào, ta cần biết căn bản về các khái niệm SSL và X509. Mặc định, MySQL dùng các kết nối giải mã giữa client và server. Điều này có nghĩa là một người khi truy cập được vào mạng thì có thể biết được mọi hoạt động của người dùng và thấy được tất cả dữ liệu đang được nhận hay gởi đi. Thậm chí họ cũng có thể thay đổi dữ liệu trong khi nó được chuyển đi giữa client và server. Để làm tăng tính bảo mật lên một chút, người dùng có thể nén client/server traffic bằng cách sử dụng lựa chọn --compress khi gọi các chương trình client. Tuy nhiên, nó không đánh lạc hướng được attacker.

Khi người dùng cần chuyển thông tin thông qua mạng theo thiết kế an toàn, kết nối giải mã là không chấp nhận được. Mã hoá là cách làm cho bất kỳ loại dữ liệu nào cũng không đọc được. Thật ra, thực tiễn yêu cầu thêm nhiều yếu tố bảo mật từ các thuật toán mã hoá. Chúng có thể chống lại nhiều cách tấn công như thay đổi trật tự của các message đã mã hoá hoặc lặp lại hai lần dữ liệu.

- SSL protocol

  Secure Socket Layer (SSL) protocol được tạo ra bởi Netscape để đảm bảo bảo mật các giao tác giữa server và client.

  SSL là giao thức mà sử dụng các thuật toán mã hóa khác nhau để bảo đảm rằng dữ liệu nhận được thông qua mạng chung có thể là đúng. Đó là kỹ thuật để dò tìm các dữ liệu bị lặp, bị mất hoặc bị thay đổi. SLL cũng kết hợp chặt chẽ các thuật toán cung cấp sự kiểm tra nhận dạng sử dụng chuẩn X509.

- Chuẩn X509

  X509 làm cho nó có thể nhận dạng một ai đó trên mạng. Nó không được sử dụng thông dụng nhất trong các ứng dụng điện tử. Trong các giới hạn cơ bản, một vài công ty gọi nó "Certificate Authority" (CA : bằng chứng nhận quyền) mà thiết kế các chứng thực điện tử cho những ai cần đến chúng. Các chứng thực đáp lại các thuật toán mã hoá bất đối xứng. Chủ nhân của (bằng) chứng thực có thể chỉ chứng thực cho party khác như là bằng chứng về nhận dạng. Chứng thực gồm khoá chung của chủ nhân của nó. Bất kỳ dữ liệu nào được mã hoá với khoá công cộng này chỉ có thể giải mã bằng khoá bí mật tương ứng, khoá này được giữ bởi chủ nhân của chứng thực.

3.2.3.1. Yêu cầu (Requirements)

Để sử dụng các kết nối SSL giữa MySQL server và các chương trình client, hệ thống của người dùng cần phải hỗ trợ OpenSSL và phiên bản MySQL phải từ 4.0.0 trở lên. Để có các kết nối bảo mật để làm việc với MySQL, người dùng cần phải thực hiện các công việc sau :

1. Cài đặt thư viện OpenSSL
2. Khi người dùng cấu hình MySQL, chạy **configure** script với các lựa chọn --with-vio và -with-openssl.
3. Phải chắc rằng người dùng nâng cấp các bản phân quyền bao gồm các cột SSL-related trong bảng mysql.user. Đây là điều cần thiết nếu các bản phân quyền của người dùng có từ các phiên bản trước 4.0.0. Thủ tục nâng cấp
4. Để kiểm tra mysql server có hỗ trợ OpenSSL hay không, kiểm tra giá trị của biến hệ thống have_openssl:

```
mysql> SHOW VARIABLES LIKE 'have_openssl';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| have_openssl | YES |
+---------------+-------+
```

Nếu giá trị là yes, server hỗ trợ các kết nối OpenSSL

3.2.3.2. Cách thức hoạt động của SSL protocol

SSL hoạt động gồm 7 bước sau :

1. Client gởi yêu cầu hay data tới server
2. Server gởi lại cho client public key với các certificate của nó.
3. Client sẽ kiểm tra certificate được cấp phát bởi party tin cậy (thường lầ root CA tin cậy), đó là các certificate còn giá trị và certificate phải liên quan đến (site contacted).
4. Sau đó client sẽ dùng pubic key để mã hóa secret key được lấy ngẫu nhiên. Và gởi nó tới cho server cùng với thông tin đã được mã hóa bằng secret key.

5. Server sẽ giải mã lấy secret key (symmetric key) bằng chính private key của nó và dùng nó giải mã thông tin mà client đã gởi qua.
6. Server lại mã hóa thông tin mà nó trả lời cho client bằng chính secret key và gởi thông tin đã mã hóa cho client.
7. Client giải mã lại thông tin trả lời vừa nhận được bằng secret key và hiển thị thông tin.

3.2.3.3. Các lựa chọn phân quyền của SSL

MySQL có thể kiểm tra các thuộc tính chứng thực X509 thêm vào trong sự chứng thực thông thường cơ bản là dựa trên username và password. Đối với các lựa chọn SSL-related cho tài khoản MySQL, sử dụng mệnd đề REQUIRE của cú pháp GRANT satement.

Các khả năng khác nhau cho giới hạn các loại kết nối tài Khoản :

- Nếu tài khoản không có các yêu cầu SSL hay X509, các kết nối giải mã được cho phép nếu username và password có giá trị. Tuy nhiên, các kết nối mã hoá cũng có thể sử dụng tại lựa chọn của client, nếu client có chứng thực chính xác và các key file.
- Lựa chọn REQUIRE SSL giới hạn server chỉ cho phép các kết nối mã hóa SSL cho các tài khoản. Chú ý rằng lựa chọn này có thể bị bỏ qua nếu như có bất kỳ dòng dữ liệu ACL mà cho phép các kết nối non-SSL.

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret' REQUIRE SSL;
```

- REQUIRE X509 có nghĩa là client phải có chứng thực có giá trị tuy nhiên chứng thực chính xác, người yêu cầu, và chủ đề là không quan trọng. Chỉ có yêu cầu mà có thể thực hiện để kiểm tra chữ ký của nó với một trong các chứng thực CA.

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret' REQUIRE X509;
```

- REQUIRE ISSUER 'issuer' đặt sự hạn chế đối với việc kết nối mà client phải trình chứng thực X509 có giá trị được cấp bởi CA 'issuer'. Nếu client trình chứng thực có giá trị nhưng khác issuer, server loại bỏ kết nối. sử dụng các chứng thực X509 luôn luôn bao hàm mã hoá, vì thế lựa chọn SSL là không cần thiết.

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret'
-> REQUIRE ISSUER '/C=FI/ST=Some-State/L=Helsinki/
O=MySQL Finland AB/CN=Tonu Samuel/Email=tonu@example.com'
```

Chú ý rằng giá trị ISSUER cần được nhập như là một chuỗi đơn.

- REQUIRE SUBJECT 'subject' đặt sự hạn chế đối với việc kết nối mà client phải trình chứng thực X509 có giá trị với chủ đề 'subject' trên nó. Nếu client trình chứng thực có giá trị nhưng khác chủ đề, server loại bỏ kết nối.

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret'
-> REQUIRE SUBJECT '/C=EE/ST=Some-State/L=Tallinn/
O=MySQL demo client certificate/
CN=Tonu Samuel/Email=tonu@example.com';
```

Chú ý rằng giá trị SUBJECT cần được nhập như là một chuỗi đơn.

- REQUIRE CIPHER 'cipher' cần được bảo đảm là tính toán và chiều dài của key được dùng đủ mạnh. SSL có thể yếu nếu các thuật toán cũ với các khóa mã hoá có chiều dài ngắn được dùng. Sử dụng lựa chọn này,

chúng ta có thể yêu cầu vài phương pháp tính toán chính xác để cho phép kết nối.

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret'
-> REQUIRE CIPHER 'EDH-RSA-DES-CBC3-SHA';
```

Các chọn lựa SUBJECT, ISSUER, và CIPHER có thể kết hợp với nhau trong mệnh đề REQUIRE như sau :

```
mysql> GRANT ALL PRIVILEGES ON test.* TO 'root'@'localhost'
-> IDENTIFIED BY 'goodsecret'
-> REQUIRE SUBJECT '/C=EE/ST=Some-State/L=Tallinn/
O=MySQL demo client certificate/
CN=Tonu Samuel/Email=tonu@example.com'
-> AND ISSUER '/C=FI/ST=Some-State/L=Helsinki/
O=MySQL Finland AB/CN=Tonu Samuel/Email=tonu@example.com'
-> AND CIPHER 'EDH-RSA-DES-CBC3-SHA';
```

Chú ý rằng mỗi giá trị SUBJECT và ISSURE có thể được nhập vào như là một chuỗi đơn. Bắt đầu từ MySQL 4.0.4, từ khoá AND là chọn lựa tuỳ ý giữa các lựa chọn REQUIRE. Các lựa chọn cũ hơn không quan tâm, nhưng không có lựa chọn có thể được định rõ 2 lần.

3.2.3.4. Các chọn lựa SSL Command-line

Chuỗi mô tả các lựa chọn được dùng để chỉ định sử dụng cho SSL, các certificate file, các key file. Các lựa chọn có giá trị bắt đầu từ MySQL 4.0. Chúng có thể được đưa ra bằng các dòng lệnh hoặc trong option file.

- --ssl

  Đối với server, chọn lựa này chỉ định là server cho phép các kết nối SSL. Đối với chương trình client, nó cho phép client kết nối vào server sử dụng SSL. Lựa chọn này không không đủ cho chính nó để thự hiện kết nối SSL được sử dụng. Người dùng cũng phải chỉ rõ các option --ssl-ca, --ssl-cert, và --ssl-key.

  Lựa chọn này thường được sử dụng nhiều hơn trong form trái ngược lại của nó để chỉ ra rằng SSL có thể không được dùng. Để làm như vậy, chỉ định lựa chọn như --skip-ssl hoặc --ssl=0. Chú ý là sử dụng –ssl không đòi hỏi kết nối SSL. Ví dụ, nếu server hoặc client được biên dịch mà không hỗ trợ SSL, kết nối giải mã thông thường được sử dụng.

  Cách bảo mật để chắc rằng kết nối SSL được dùng là để tạo tài khoản trên server bao gồm mệnh đề REQUIRE SSL trong GRANT statement. Sau đó sử dụng tài khoản này để kết nối vào server, với cả server và client có hỗ trợ SSL được enable.

- --ssl-ca=file_name

  Đường dẫn tới file với danh sách của SSL CAs được tin cậy.

- --ssl-capath=directory_name

  Đường dẫn tới thư mục chứa các chứng thực SSL CA được tin cây trong định dạng pem.

- --ssl-cert=file_name

  Tên của file chứng thực SSL dùng cho thiết lập kết nối an toàn.

- --ssl-cipher=cipher_list

  Danh sách các tính toán cho được cho phép để dùng cho mã hoá SSL, Cipher_list có cùng định dạng openssl ciphers command.

- --ssl-key=file_name

  Tên của file khoá SSL dùng để thiết lập kết nối an toàn

3.2.3.5. Kết nối với MySQL Remotely từ Windows với SSH

Đây là ghi chú về cách kết nối để có một kết nối an toàn tới MySQL server ở xa với SSH

1. Cài đặt SSH client trên máy Windowns của người dùng.
2. Start Winđows SSH client của người dùng.

   Đặt Host_Name = yourmysqlserver_URL_or_IP.

   Đặt userid=your_userid để đăng nhập vào server của người dùng. Giá trị userid này có thể không giống như username của tài khoản MySQL của người dùng.
3. Đặt port forwarding. Thực hiện remote forward (đặt local_port: 3306, remote_host: yourmysqlservername_or_ip, remote_port: 3306) hoặc local forward (đặt Set port: 3306, host: localhost, remote port: 3306).
4. Lưu mọi thứ, mặt khác người dùng sẽ phải thực hiện lại trong lần kế tiếp.
5. Đăng nhập vào server của người dùng với SSH session người dùng vừa mới tạo.
6. Trên máy Windows của người dùng, start vài ứng dụng ODBC (chẳng hạn như Access).
7. Tạo mới file trong Windows và link tới MySQL sử dụng đường dẫn ODBC cùng với cách người dùng thường hay làm, trừ loại trong localhost cho MySQL host server, chứ không phải mysql server name của người dùng. Người dùng cần phải có kết nối ODBC tới MySQL, mã hoá dùng SSH.

3.2.4. Bảo mật cơ sở dữ liệu

3.2.4.1. Cơ chế bảo mật cơ sở dữ liệu (CSDL) trong MySQL

Hệ thống bảo mật trong MySQL tương đối phức tạp. MySQL thực hiện bảo mật CSDL dựa vào hệ thống quyền truy cập.

Chức năng chính của hệ thống điều khiển truy cập gồm :

- Chứng thực sự kết nối của user vào MySQL Server.
- Kết hợp user với các quyền riêng trên CSDL chẳng hạn như SELECT, UPDATE, INSERT, DELETE.

Để thực hiện điều khiển hệ thống cần phải truy cập một số thông tin đặc biệt. Các thông tin này được tổ chức thành một CSDL chính là **mysql**.

3.2.4.2. Cơ sở dữ liệu mysql

Ngay sau khi cài đặt hệ quản trị MySQL, sẽ có 2 CSDL được tạo ra :

- **mysql** : lưu trữ các thông tin phục vụ cho việc quản trị.
- **test** : là csdl mà bất kỳ user nào cũng có đầy đủ các quyền trên đó.

Ở đây ta chỉ quan tâm đến CSDL **mysql**. CSDL **mysql** lưu trữ thông tin chung và các đặc quyền của user để kiểm tra kết nối và quyền thực hiện các câu lệnh. Dó đó chỉ có các nhà quản trị CSDL mới có quyền truy cập và thao tác trên CSDL này. CSDL **mysql** cũng giống như bất kì một CSDL nào khác trong MySQL. Tất cả các tập tin dữ liệu của nó lưu trữ dữ liệu trong một thư mục dưới thư mục gốc mysql (mysql parent directory).

Các bảng trong CSDL **mysql** sử dụng cho mục đích bảo mật :

- user
- db
- host

- columns_priv
- tables_priv
- Procs_priv

Người dùng có thể thực thi các câu lệnh trên các bảng này như bất kì bảng nào trong các CSDL bình thường khác.

Các bảng này trong CSDL **mysql** còn được gọi là các bảng phân quyền (Grant tables).

a) **Giới thiệu các bảng trong csdl mysql**

- Bảng user, db, host

Table Name	user	db	host
Scope columns	Host	Host	Host
	User	Db	Db
	Password	User	
Privilege columns	Select_priv	Select_priv	Select_priv
	Insert_priv	Insert_priv	Insert_priv
	Update_priv	Update_priv	Update_priv
	Delete_priv	Delete_priv	Delete_priv
	Index_priv	Index_priv	Index_priv
	Alter_priv	Alter_priv	Alter_priv
	Create_priv	Create_priv	Create_priv
	Drop_priv	Drop_priv	Drop_priv
	Grant_priv	Grant_priv	Grant_priv
	Create_view_priv	Create_view_priv	Create_view_priv
	Show_view_priv	Show_view_priv	Show_view_priv

	Create_routine_priv	Create_routine_priv	
	Alter_routine_priv	Alter_routine_priv	
	References_priv	References_priv	References_priv
	Reload_priv		
	Shutdown_priv		
	Process_priv		
	File_priv		
	Show_db_priv		
	Super_priv		
	Create_tmp_table_priv	Create_tmp_table_priv	Create_tmp_table_priv
	Lock_tables_priv	Lock_tables_priv	Lock_tables_priv
	Execute_priv		
	Repl_slave_priv		
	Repl_client_priv		
Security columns	ssl_type		
	ssl_cipher		
	X509_issuer		
	X509_subject		
Resource control columns	max_questions		
	max_updates		
	max_connections		
	max_user_connections		

Bảng 3.2.4.2.a Danh sách các cột của bảng user, host, db trong MySQL

Các cột ssl_type, ssl_cipher, x509_issuer, and x509_subject columns được thêm vào từ MySQL 4.0.0.

Các cột Create_tmp_table_priv, Execute_priv, Lock_tables_priv, Repl_client_priv, Repl_slave_priv, Show_db_priv, Super_priv, max_questions, max_updates, and max_connections được thêm vào từ MySQL 4.0.2. tuy nhiên Execute_priv không được dùng cho tới MySQL 5.0.3.

Các cột The Create_view_priv and Show_view_priv được thêm vào từ MySQL 5.0.1.

Các cột Create_routine_priv, Alter_routine_priv và max_user_connections được thêm vào từ MySQL 5.0.3.

Server sử dụng thông tin trong cả 3 bảng này trong cả 2 giai đoạn điều khiển truy cập.

- Bảng tables_priv, columns_priv

Trong suốt giai đoạn 2 của điều khiển truy cập, server kiểm tra các yêu cầu để chắc rằng client đưa ra yêu cầu có đủ quyền thực hiện yêu cầu đó. Khi đó server dựa vào các thông tin trong các bảng user, db, host. Ngoài ra server còn tham khảo thêm trong các bảng tables_priv and columns_priv. các bảng này cung cấp các quyền riêng cần thiết đối với mức độ table và column.

Table Name	tables_priv	columns_priv
Scope columns	Host	Host
	Db	Db
	User	User
	Table_name	Table_name
		Column_name
Privilege columns	Table_priv	Column_priv
	Column_priv	
Other columns	Timestamp	Timestamp
	Grantor	

Bảng 3.2.4.2.b Bảng tables_priv, columns_priv trong MySQL

Cột Timestamp và Grantor không sử dụng.

- Bảng procs_priv :

Từ phiên 5.0.3 CSDL mysql có thêm bảng procs_priv. Bảng này được dùng để thẩm định lại các yêu cầu mà được lưu trữ thường xuyên.

Table Name	Procs_priv
Scope columns	Host
	Db
	User
	Routine_name
Privilege columns	Proc_priv
Other columns	Timestamp
	Grantor

Bảng 3.2.4.2.c Danh sách các cột của bảng Procs_priv trong MySQL

b) **Chức năng và cấu trúc của các column trong các table của csdl mysql**

➢ Chức năng

Mỗi table đều có các cột phạm vi (scope columns) và các cột quyền (privilege column).

- ✓ Các cột phạm vi dùng để chỉ định rõ phạm vi cho mỗi mục của table hay chính là ngữ cảnh mà dòng dữ liệu áp dụng.
- ✓ Các cột quyền chỉ định thông tin về các quyền trên một mức của CSDL tùy theo từng table.

Server kết hợp các thông tin trong nhiều bảng phân quyền để thực hiện hoàn thành việc mô tả các đặc quyền của user.

➢ Nội dung của các column

✓ Các cột phạm vi :

Host : Tên máy người dùng. Trong MySQL, người dùng có thể giới hạn số người truy cập dựa vào location mà người đó kết nối.

User : Tên mà người dùng dùng để truy cập vào MySQL

Password : mật khẩu người dung.

Db : Tên CSDL

Table_name : Tên bảng trong CSDL

Column_name : Tên của cột trong một bảng trong CSDL

✓ Các cột quyền : Lưu các thông tin về các quyền của user trên từng mức độ của CSDL tùy theo nó thuộc bảng nào.

➢ Cấu trúc các cột

✓ Các cột phạm vi :

Column Name	Type
Host	CHAR(60)
User	CHAR(16)
Password	CHAR(16)
Db	CHAR(64)
Table_name	CHAR(64)
Column_name	CHAR(64)
Routine_name	CHAR(64)

Bảng 3.2.4.2.d Phạm vi các cột trong các bảng

Đối với các phiên bản MySQL trước 3.23, cột Db là CHAR(32) trong một vài table và là CHAR(60) trong một số table khác.

✓ Các cột quyền

Trong các bảng user, db, host, mỗi quyền được liệt kê trong các cột quyền khác nhau được khai báo như ENUM ('N', 'Y') giá trị mặc định 'N'. Trong các từ, mỗi quyền có thể được disabled hoặc enabled, với mặc định là disabled.

Trong các table tables_priv, columns_priv và procs_priv, các cột quyền được khai báo như là các cột SET columns. Mỗi dòng của table chỉ ra một quyền của user mà thôi. Do đó để biết được đầy đủ các quyền của user trên một table,

column, proc cần phải kết hợp một tập các record. Các giá trị có thể lưu trong các cột :

Table Name	Column Name	Possible Set Elements
Tables_priv	Table_priv	'Select', 'Insert', 'Update', 'Delete', 'Create', 'Drop', 'Grant', 'References', 'Index', 'Alter'
Tables_priv	Column_priv	'Select', 'Insert', 'Update', 'References'
columns_priv	Column_priv	'Select', 'Insert', 'Update', 'References'
procs_priv	Proc_priv	'Execute', 'Alter Routine', 'Grant'

Bảng 3.2.4.2.e Các giá trị trong các cột phân quyền

c) **Sử dụng các table phân quyền**

Server sử dụng các bảng như sau :

- Các cột phạm vi trong bảng user định rõ khi nào thì loại bỏ hay cho phép các kết nối thực hiện. Khi các kết nối được cho phép, một số đặc quyền đã cấp trong các bảng user chỉ cho toàn bộ các quyền của user. Các đặc quyền này được áp dụng cho tất cả các csdl trên server.
- Các cột phạm vi trong bảng db định rõ user nào được phép truy cập vào các csdl nào từ các host nào. Các cột đặc quyền định rõ thao tác nào được phép. Một đặc quyền đã cấp trong một mức csdl áp dụng vào và tất cả các bảng của nó. (các quyền áp dụng đối với csdl)
- Bảng host được dùng trong liên kết với bảng db khi người dùng muốn chỉ ra các dòng trong bảng db để áp dụng vào các host riêng. Bảng host không ảnh hưởng bởi các lệnh GRANT và REVOKE.
- Các bảng tables_priv and columns_priv thì tương tự như bảng db, nhưng có nhiều fine-grained : chúng áp dụng tại các mức độ bảng

và cột hơn là ở mức độ csdl. Các quyền đặc quyền đã cấp cho mức độ bảng thì áp dụng cho bảng và cho tất cả các cột trong bảng. Các đặc quyền được cấp cho tại cột nào thì nó chỉ có tác dụng cho cột đó mà thôi.

- Bảng procs_priv áp dụng cho các thủ tục lưu trữ (stored routines). Đặc quyền đã cấp tại mức thủ tục chỉ áp dụng cho các thủ tục đơn.

3.2.4.3. Các quyền MySQL hỗ trợ

Các đặc quyền :

Privilege	Column	Context
CREATE	Create_priv	databases, tables, or indexes
DROP	Drop_priv	databases or tables
GRANT	Grant_priv	databases, tables, or stored routines
REFERENCES	References_priv	databases or tables
ALTER	Alter_priv	tables
DELETE	Delete_priv	tables
INDEX	Index_priv	tables
INSERT	Insert_priv	tables
SELECT	Select_priv	tables
UPDATE	Update_priv	tables
CREATE VIEW	Create_view_priv	views
SHOW VIEW	Show_view_priv	views
ALTER ROUTINE	Alter_routine_priv	stored routines
CREATE ROUTINE	Create_routine_priv	stored routines
EXECUTE	Execute_priv	stored routines
CREATE	Create_tmp_table_priv	server administration

TEMPORARY TABLES		
FILE	File_priv	file access on server host
LOCK TABLES	Lock_tables_priv	server administration
PROCESS	Process_priv	server administration
RELOAD	Reload_priv	server administration
REPLICATION CLIENT	Repl_client_priv	server administration
REPLICATION SLAVE	Repl_slave_priv	server administration
SHOW DATABASES	Show_db_priv	server administration
SHUTDOWN	Shutdown_priv	server administration
SUPER	Super_priv	server administration

Bảng 3.2.4.3.a Danh sách các đặc quyền

Các quyền CREATE TEMPORARY TABLES, EXECUTE, LOCK TABLES, REPLICATION CLIENT, REPLICATION SLAVE, SHOW DATABASES, và SUPER được thêm trong MySQL 4.0.2. (EXECUTE không được thực thi cho tới MySQL 5.0.3.)

CREATE VIEW và SHOW VIEW được thêm trong MySQL 5.0.1.

CREATE ROUTINE và ALTER ROUTINE được thêm trong MySQL 5.0.3.

Để sử dụng các đặc quyền này khi nâng cấp từ các phiên bản cũ của MySQL mà không có chúng, người dùng cần phải nâng cấp các bảng đặc quyền của người dùng.

Một số đặc quyền quản trị có thể thực hiện bằng cách sử dụng chương trình **mysqladmin** hoặc đưa ra các câu lệnh SQL.

Bảng sau thể hiện các câu lệnh SQL tương ứng với đặc quyền quản trị được thực thi :

Privilege	Commands Permitted to Privilege Holders
RELOAD	flush-hosts, flush-logs, flush-privileges, flush-status, flush-tables, flush-threads, refresh, reload.
SHUTDOWN	shutdown
PROCESS	processlist
SUPER	kill

Bảng 3.2.4.3.b Danh sách các đặc quyền quản trị

Lệnh reload thực hiện việc đọc lại các bảng phân quyền vào trong bộ nhớ.

3.2.4.4. Điều khiển truy cập đặc quyền

Khi người dùng kết nối với CSDL MySQL :

- MySQL tìm xem trong bảng user có hostname, username, and password đó không.
- Nếu có, người dùng được quyền truy cập vào hệ thống. Khi người đó thực thi một câu query, trước tiên MySQL xem trong bảng **user** xem người dùng có những quyền gì. Nếu người dùng này không có đặc quyền trong bảng đó thì MySQL sẽ tìm tiếp trong bảng **db**. Hệ thống sẽ dựa trên the hostname, username, và CSDL xem các đặc quyền mà người này có. Nếu trong bảng này không có quyền để thực thi câu truy vấn thì MySQL sẽ tìm tiếp trong bảng **tables_priv** và sau đó là bảng **columns_priv** xem người đó có quyền thực thi câu truy vấn. Nếu không tìm thấy thì hệ thống sẽ báo lỗi.

Như vậy có 2 trường hợp kiểm tra kết nối :

- Kiểm tra kết nối (connection verification)
- Kiểm tra yêu cầu (request verification).

a) Giai đoạn 1 : kiểm tra kết nối

Khi người dùng cố gắng kết nối vào MySQL server, server chấp nhận hoặc từ chối các kết nối cơ bản dựa trên nhận dạng của bảng và bất cứ khi nào người dùng kiểm tra lại nhận dạng của mình bằng cách cung cấp đúng mật khẩu. Nếu không, server cấm truy cập. Nếu đúng. server chấp nhận kết nối, sau đó chuyển sang giai đoạn 2 và chờ đợi các yêu cầu.

Nhận dạng của user được dựa trên 2 phần thông tin sau:

- client host từ kết nối của người dùng
- username MySQL của người dùng

Bất kì ai kết nối vào MySQL, hệ thống đều đòi hỏi username, password, và hostname.

- username là tên người muốn kết nối
- password là công cụ được thêm vào để xác nhận người dùng
- hostname là tên máy mà người dùng kết nối. MySQL không thể giới hạn số lượng người kết nối mà có thể hạn chế số máy kết nối.
- Các giá trị host trong bảng user có thể được chỉ như sau :
- Giá trị Host có thể là hostname hoặc số IP, hoặc ‘localhost’ để chỉ local host.
- Người dùng có thể sử dụng các ký tự mở rộng như ‘%’ và ‘_’ trong các giá trị của cột Host. điều này có cùng ý nghĩa với thao tác việc dùng ký tự thay thế với toán tử LIKE.

Quá trình kiểm tra kết nối khá đơn giản. MySQL kiểm tra thông tin username, password, và hostname nhập vào có trong bảng user hay không? Nếu có, người dùng được phép kết nối, nếu không sẽ không cho phép truy cập. Việc kiểm tra nhận dạng được thực hiện bằng cách sử dụng 3 cột phạm vi của bảng user (Host, User và Password). Server chấp nhận kết nối chỉ khi các cột Host và User trong vài record trong bảng user tương ứng với hostname và username của client, và client cung cấp password được chỉ rõ trong record đó.

Cột password có thể là không có dữ liệu. Khi đó user kết nối mà không cần password. Các giá trị Password non-blank trong bảng user đại diện cho password đã mã hoá. MySQL không lưu trữ dạng tường tận của password (plaintext) mà nó được mã hoá (bằng hàm PASSWORD()). Password đã được mã hoá sẽ được dùng trong suốt qúa trình kết nối khi kiểm tra password đã đúng. Theo cách nhìn nhận của MySQL thì password đã được mã hoá là password đúng, vì thế người dùng không nên đưa nó cho bất kỳ người nào truy cập vào nó! Đặc biệt, không cho phép các user không phải là quản trị đọc các bảng trong csdl mysql!

Từ phiên bản 4.1, MySQL dùng phương pháp chứng thực để bảo vệ password mạnh hơn trong suốt quá trình kết nối hơn là các phiên bản trước đó. Nó bảo mật ngay cả các gói tin TCP/IP bị sniffed hoặc csdl mysql bị đánh cắp.

Từ phiên bản 4.1.1 MySQL dùng thuật toán SHA-1 để mã hóa password. Password được lưu trong CSDL mysql là giá trị băm của password sau khi dùng SHA-1 băm hai lần liên tiếp.

Quá trình chứng thực password :

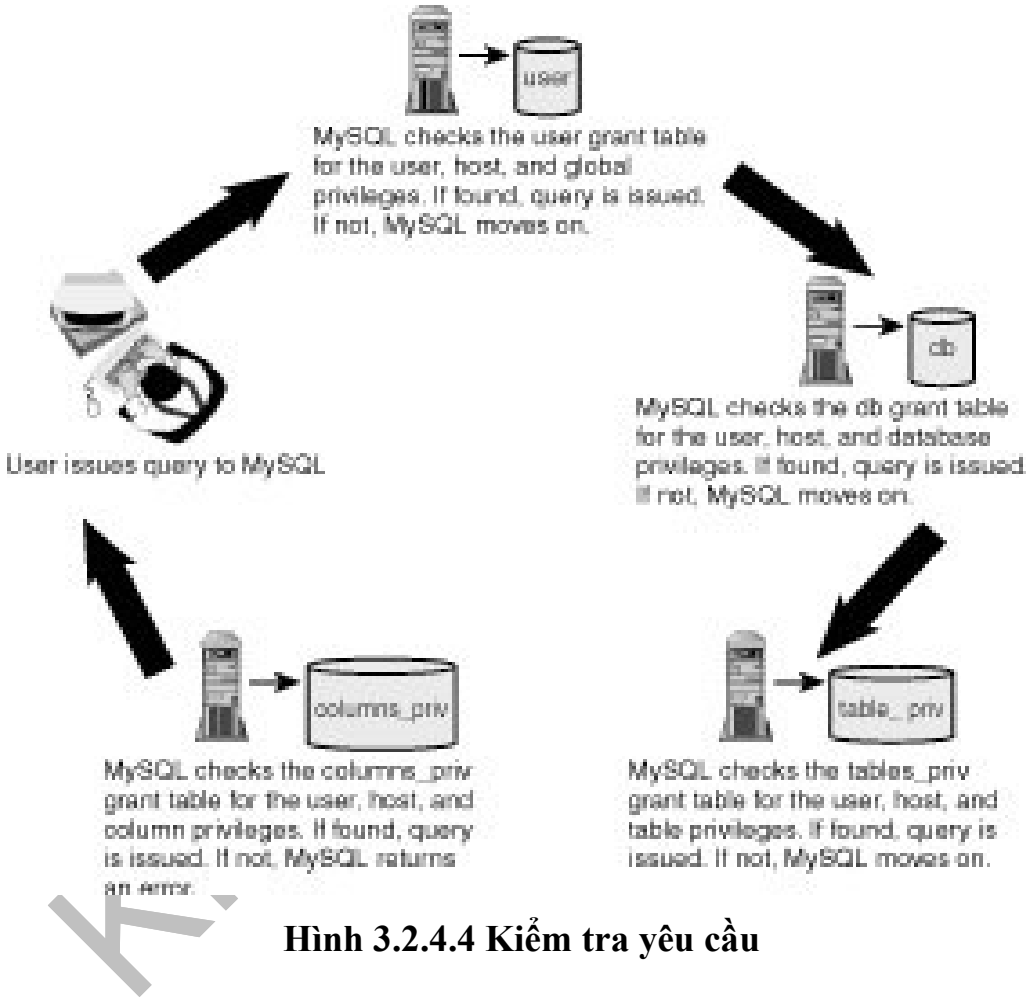
1. Khi nhận được yêu cầu kết nối từ client, Server sẽ tạo một message ngẫu nhiên public_seed và gởi nó cho client.
2. Client nhận public_seed. Sau đó dùng thuật toán băm SHA-1 băm hai lần trên password của mình giá trị nhận được là hash_stage2. Sau đó lại tạo giá trị băm hỗn hợp dựa trên public_seed và hash_stage2 rồi xor nó với hash_stage1 (là giá trị băm SHA-1 của password lần 1). Chuỗi nhận được cuối cùng là reply. Chuỗi này được gởi lại cho server.
3. Server nhận được giá trị reply do client trả về và dùng nó để kiểm tra password đúng hay sai. Trước tiên server sẽ lấy giá trị của password trong CSDL mysql là hash_stage2. Sau đó server thực hiện việc băm hỗn hợp hash_stage2 và public_seed. Chuỗi giá trị nhận được được đem xor với reply. Khi đó kết quả ta nhận được chính là hash_stage1 (đây chính là giá trị nhận được do băm password lần thứ nhất). Lại dùng SHA-1 băm hash_stage1 để nhận được giá trị băm mới candidate_hash2. Nếu candidate_hash2 bằng với hash_stage2 thì password là đúng ngược lại password là sai.

```
SERVER: public_seed=create_random_string()
        send(public_seed)
CLIENT:  recv(public_seed)
        hash_stage1=sha1("password")
        hash_stage2=sha1(hash_stage1)
        reply=xor(hash_stage1, sha1(public_seed,hash_stage2)
        send(reply)
SERVER:  recv(reply)
        hash_stage1=xor(reply, sha1(public_seed,hash_stage2))
        candidate_hash2=sha1(hash_stage1)
        check(candidate_hash2==hash_stage2)
```

b) Giai đoạn 2 : kiểm tra yêu cầu

Kiểm tra yêu cầu xảy ra mỗi khi người dùng thực hiện câu truy vấn vào CSDL Sau khi quá trình kết nối được thiết lập, mỗi dòng lệnh đều đi qua cùng một tiến trình. Điều này đảm bảo người dùng đều có giới hạn làm việc. Quá trình này cũng khá đơn giản. Mỗi khi có yêu cầu đưa ra, trước tiên MySQL kiểm tra xem người dùng này có được phân quyền ở cấp độ user hay không? Nếu có, thì người dùng này sẽ được phép làm bất cứ việc gì trên csdl trong HQTCSDL MySQL. Nếu không tìm thấy thì My SQL sẽ tìm tiếp trong bảng db. Bảng db là cơ chế bảo mật ở cấp độ tiếp theo. Đặc quyền chỉ được cấp trên 1 CSDL cụ thể. Đặc quyền SELECT ở cấp độ này cho phép người dùng xem dữ liệu trên tất cả các bảng của 1 CSDL cụ thể. Còn nếu người dùng muốn phân quyền cụ thể hơn thì hãy sử dụng bảng tables_priv và columns_priv. Bảng columns_priv là nơi cuối cùng MySQl

cấp quyền cho người dùng. Nếu người dùng không được cấp quyền ở cấp độ này thì MySQl sẽ báo lỗi. Quá trình này diễn ra rất nhanh đến nỗi ta không thể biết được. Do đó CSDL của người dùng có độ an toàn cao. Hình dưới đây mô tả quá trình kiểm tra :



Hình 3.2.4.4 Kiểm tra yêu cầu

Chương 4.Thuật toán bảo mật password trong MySQL

✍ *Chương này sẽ trình bày về thuật toán bảo mật password trong MySQL-SHA-1: ý tưởng, các bước của thuật toán, đánh giá ưu khuyết điểm. Đồng thời, chúng tôi đề xuất một số thuật toán tốt hơn có thể hạn chế được khuyết điểm của thuật toán hiện tại.*

Từ phiên bản 4.1 trở về sau MySQL mã hóa password bằng thuật toán SHA-1

4.1. Thuật toán SHA-1

4.1.1. Ý tưởng thuật toán 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. Mục đích của việc mở rộng này 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, $M_0^{(i)}$ 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, $M_0^{(i)}$ chứa 64 bit kế tiếp...

Trước khi thực hiện băm, 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 giá trị băm ban đầu của các thuật toán SHA được trình bày trong phần Phụ lục A.

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 SHA-256. 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.

Các tham số, ký hiệu và các thuật ngữ được sử dụng trong SHA.

Tham số:

a, b,c, ...,h	Các biến là các từ w bit sử dụng trong việc tính toán giá trị băm H(i).
H(i)	Giá trị băm thứ i. H(0) là giá trị băm khởi đầu. H(N) là giá trị băm cuối cùng và được sử dụng để xác định thông điệp rút gọn.
Kt	Hằng số sử dụng cho vòng lặp thứ t trong việc thực hiện băm.
k	Số lượng các số 0 thêm vào thông điệp trong giai đoạn mở rộng thông điệp.
l	Chiều dài thông điệp M (tính bằng đơn vị bit).
m	Số bit trong một khối thông điệp, M(t).
M(i)	Khối thông điệp i, với giá trị m bit.
M(i)j	Từ thứ j của khối thông điệp thứ i, M(t)0 là từ cực trái của khối thông điệp i.
n	Số lượng bit được dịch chuyển khi xử lý một từ.
N	Số lượng khối trong thông điệp mở rộng.
T	w-bit từ tạm sử dụng trong việc thực hiện băm.
w	Số lượng bit trong một từ.
Wt	Từ w-bit thứ t của bảng phân bố thông điệp.

Ký hiệu:

Các ký hiệu sau được sử dụng trong SHA và xử lý trên các từ w-bit.

^	Thao tác AND trên bit.
∨	Thao tác OR trên bit.
⊕	Thao tác XOR trên bit.
¬	Thao tác đảo bit.

+ Thao tác cộng modulo 2w.

<< Thao tác dịch trái, x << n loại bỏ n bit cực trái của từ x và thêm n bit 0 vào bên phải của kết quả.

>> Thao tác dịch phải, x >> n loại bỏ n bit cực phải của từ x và thêm n bit 0 vào bên trái của kết quả.

Thuật ngữ:

Các thuật ngữ liên quan đến chuỗi bit và số nguyên được sử dụng:

a) Một ký số thập lục (hexa) là một phần tử trong tập hợp {0, 1,..., 9, a, ..., f}. Một ký số thập lục biểu diễn một chuỗi 4-bit. Ví dụ, ký số thập lục "7" biểu diễn chuỗi 4-bit "0111", ký số hexa "a" biểu diễn chuỗi 4 bit "1010".

b) Một từ là một chuỗi w-bit có thể được biểu diễn dưới dạng một dãy các ký số hexa. Để chuyển đổi một từ sang ký số hexa, mỗi chuỗi 4-bit được chuyển sang giá trị hexa tương ứng như phần (a). Ví dụ, chuỗi 32-bit

1010 0001 0000 0011 1111 1110 0010 0011

có thể được biểu diễn như sau "a103fe23", và chuỗi 64 bit

1010 0001 0000 0011 1111 1110 0010 0011

0011 0010 1110 1111 0011 0000 0001 1010

có thể được biểu diễn như sau "a103fe2332ef301a"

c) Quy ước "big-endian" được sử dụng trong tài liệu này khi biểu diễn từ có 32 và 64 bit. Do đó với mỗi từ, bit đầu tiên nằm ở vị trí cực trái.

d) Một số nguyên có thể được biểu diễn dưới dạng một từ hoặc một cặp từ. Một từ biểu diễn độ dài thông điệp theo bit, l, được sử dụng trong thao tác mở rộng thông điệp (phần b).

Một số nguyên nằm trong khoảng 0 và 232-1 có thể được biểu diễn bằmg một từ 32-bit. 4 bit cuối cùng của số nguyên được biểu diễn bằng ký

số hexa cực phải của từ. Ví dụ số nguyên 291 = 28 + 25 + 21 + 20 = 258 + 32 + 2 + 1 được biểu diễn bằng từ hexa 0x00000123.

Tương tự, số nguyên trong khoảng 0 và 264-1 có thể biểu diễn bằng từ 64-bit.

Nếu Z là một số nguyên, $0 \le Z < 264$ thì $Z = 232X + Y$, trong đó $0 \le X < 232$ và $0 \le Y < 232$. Do X và Y có thể được biểu diễn bằng từ 32-bit x và y, nên số nguyên Z cũng có thể biểu diễn bằng một cặp từ (x, y). Tính chất này được sử dụng trong SHA-1 và SHA-256.

Nếu Z là một số nguyên, $0 \le Z < 2128$ thì $Z = 264X + Y$, trong đó $0 \le X < 264$ và $0 \le Y < 264$. Do X và Y có thể được biểu diễn bằng từ 64-bit x và y, nên số nguyên Z cũng có thể biểu diễn bằng một cặp từ (x, y). Tính chất này được sử dụng trong SHA-384 và SHA-512.

e) Trong thuật toán băm an toàn, kích thước của khối thông điệp m bit dựa vào thuật toán sau:
   - Đối với SHA-1 và SHA-256, mỗi khối thông điệp có 512 bit biểu diễn dưới dạng một dãy 16 từ 32-bit.
   - Đối với SHA-384 và SHA-512, mỗi khối thông điệp có 1024 bit biểu diễn dưới dạng một dãy 16 từ 64-bit.

Các thao tác xử lý dưới đây được áp dụng cho từ w-bit trong cả 5 thuật toán : SHA-1, SHA-224 và SHA-256 thao tác trên từ 32-bit (w = 32), SHA-384 và SHA-512 thao tác trên từ 64-bit (w = 64).

- Các phép toán luận lý trên bit: $\wedge$, $\vee$, $\oplus$ và $\neg$
- Phép cộng modulo 2w.

  Phép cộng x + y được định nghĩa như sau. Từ x và y biểu diễn số nguyên X và Y trong đó $0 \le X < 2w$ và $0 \le Y < 2w$.

$$Z = (X + Y) \bmod 2w. \quad (4.1)$$

thì $0 \le Z < 2w$. Biến đổi số nguyên Z thành từ z, ta có $z = x + y$.

Phép toán dịch phải SHRn(x) với x là từ w-bit và n là số nguyên $0 \le n < w$ định nghĩa như sau

$$SHRn(x) = x >> n. \quad (4.2)$$

Phép toán này được sử dụng trong SHA-256, SHA-384 và SHA-512.

- Phép toán quay phải ROTRn(x) với x là từ w-bit và n là số nguyên $0 \le n < w$, được định nghĩa như sau:

$$ROTRn(x) = (x >> n) \vee (x << w - n) \quad (4.3)$$

Như vậy, ROTRn(x) tương đương cho một thao tác xoay vòng từ x về phía phải n vị trí.

Phép toán này được sử dụng trong SHA-256, SHA-384 và SHA-512.

- Phép toán quay trái ROTLn(x) với x là từ w-bit và n là số nguyên $0 \le n < w$, được định nghĩa như sau:

$$ROTLn(x) = (x << n) \vee (x >> w - n) \quad (4.4)$$

Như vậy, ROTLn(x) tương đương cho một thao tác xoay vòng từ x về phía trái n vị trí. Phép toán này được sử dụng trong SHA-1.

- Lưu ý rằng các phép toán sau là tương đương với w là không đổi.

$$ROTLn(x) \approx ROTRw\text{-}n(x)$$

$$ROTRn(x) \approx ROTLw\text{-}n(x).$$

4.1.2. Thuật toán SHA-1

4.1.2.1. Giới thiệu

SHA-1 dựa trên các nguyên lý tương tự với những nguyên lý mà giáo sư Ronald L.Rivest của MIT khi thiết kế thuật toán băm MD4, SHA-1 được đề xuất vào tháng 4 năm 1995.

Khi nhập vào một thông điệp có chiều dài bất kỳ nhỏ hơn 2^{64} bit, SHA-1 cho ra kết quả là một thông điệp rút gọn (hay giá trị băm) dài 160 bits.

Trước đâ y, SHA-1 được gọi là an toàn vì không thể tìm ra thông điệp liên quan đến thông điệp rút gọn hay tìm ra hai thông điệp khác nhau nhưng có cùng thông điệp rút gọn. Bất kỳ thay đổi nào của thông điệp, với xác suất cao, kết quả vẫn cho ra các thông điệp rút gọn khác nhau.

4.1.2.2. Thao tác tiền xử lý

a) Các hàm và các hằng số được dùng trong thuật toán SHA-1 xem phần phụ lục A.3

b) Mở rộng thông điệp

Thông điệp M được mở rộng trước khi thực hiện băm. Mục đích của việc mở rộng này là để đảm bảo thông điệp mở rộng có độ dài là bội số của 512bit

Giả sử độ dài của thông điệp M là l bit. Thêm bit 1 vào cuối thông điệp, theo sau là k bit 0 (k là số không âm nhỏ nhất sao cho l+1+k=448(mod512)). Sau đó thêm khối 64 bit là biểu diễn nhị phân của l.

Ví dụ, thông điệp (8-bit ASCII) "abc" có độ dài 8x3=24, do đó thông điệp được mở rộng bằng 1 bit "1", 448-(24+1) = 423 bit "0" và chiều dài thông điệp trở thành thông điệp mở rộng 512 bit.

c) Phân tích thông điệp đã mở rộng

Sau khi thông điệp đã mở rộng, thông điệp cần được phân tích thành N khối m-bit trước khi thực hiện băm. 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, $M_0^{(i)}$ chứa 32 bit kế tiếp...

d) Khởi tạo giá trị băm

Giá trị băm là một chuỗi bit có kích thước bằng kích thước message digest (trừ SHA-384) gồm các words ghép lại. Trong đó $H_j^{(i)}$ là word j trong giá trị băm ở lần lặp i, với $0 \le i \le N$ (số block có được sau khi chia văn bản được đệm) và $0 \le j \le$ số word trong giá trị băm – 1. Trước khi thực hiện băm, 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.

➢ Hằng số và giá trị khởi tạo của SHA-1 xem phần phụ lục A.1

4.1.2.3. Thuật toán của bước tính toán giá trị băm SHA-1:

SHA-1 được sử dụng để băm thông điệp M dài l bit $0 \le l \le 2^{64}$. Thuật toán sử dụng :

- Một bảng phân bố thông điệp gồm 80 từ 32-bit.
- 5 biến 32 bit.
- Một giá trị băm gồm 5 từ 32-bit.

Kết quả của SHA-1 là thông điệp rút gọn 160-bit.

Các từ của bảng phân bố thông điệp được ký hiệu W0, W1, ..., W79. 5 biến ký hiệu a, b, c, d, và e. Các từ của giá trị băm ký hiệu $H_0^{(i)}$, $H_1^{(i)}$, …, $H_4^{(i)}$,

H(0) giữ giá trị băm ban đầu, được thay thế bằng các giá trị băm thành công H(i) sau khi mỗi khối thông điệp được xử lý và kết thúc bằng giá trị băm cuối cùng H(N). Xem chi tiết phần phụ lục A.4.1

4.1.3. Đánh giá ưu khuyết điểm

4.1.3.1. Ưu điểm

- Cùng với MD5, SHA-1 được xem là 2 thuật toán BĂM phổ biến nhất hiện nay.
- SHA-1 được xem là an toàn đối với hiện tượng đụng độ vì rất khó tìm được hai thông điệp có cùng giá trị băm.
- Chuẩn SHA-1 được chứng nhận bởi Viện Công nghệ và Tiêu chuẩn quốc gia (NIST) Mỹ, và là phương pháp mã hoá duy nhất được chấp nhận trong nội bộ chính phủ Mỹ.
- Được coi là chuẩn của việc bảo vệ các kênh liên lạc trực tuyến tồn tại 9 năm qua.

4.1.3.2. Khuyết điểm

- SHA-1 được thiết kế trên các bộ vi xử lý 32-bit, thế hệ sắp tới của các bộ vi xử lý có dùng các từ 64-bit (word), mà SHA-1 không xử ký hiệu quả trên bộ vi xử lý này.
- Tháng 2 năm 2005, SHA-1 đã bị tấn công bởi một nhóm 3 chuyên gia Xiaoyun Wang, Yiqun Lisa Yin, và Hongbo Yu, một nhóm các nhà nghiên cứu của trường Đại học Quảng Đông, Viện Khoa học Trung

Quốc và Trường Đại học Shanghai Jiaotong. Thuật toán này đã bị giải mã thông qua phương pháp tính toán phân bổ[1].

Việc tìm kiếm giá trị đụng độ của SHA thường đòi hỏi một sức mạnh tính toán rất lớn. Các nhà nghiên cứu Trung Quốc khi tấn công SHA-1 đã không có nhiều siêu máy tính trong tay, nên thay vào đó, họ sử dụng một chương trình điện toán phân tán để khai thác sức mạnh nhàn rỗi của hàng nghìn máy tính trên thế giới và hoàn tất công việc. Phá SHA-1, khó hơn gấp 16 lần tấn công MD5, cần 300.000 máy tính nhưng phải mất xấp xỉ 74 năm. Tuy nhiên, với việc tận dụng được sức mạnh liên kết của nhiều máy tính gia đình như các nhà khoa học Trung Quốc đã làm nói trên, thời gian thực hiện điều này đã được rút ngắn rất nhiều.

Xem chi tiết tấn công phần Phu luc C

(1) **Phương pháp tính toán phân bổ**: Là một dạng tính toán trong đó các thành phần và đối tượng khác nhau (tạo nên một ứng dụng) được đặt trên nhiều máy tính khác nhau kết nối vào mạng. Ví dụ: có thể đánh giá ứng dụng xử lý văn bản Word bằng cách phân chia những tính năng chính của chúng cho nhiều máy tính khác nhau: thành phần biên tập, đối tượng kiểm tra chính tả, tính năng bảo mật... Trong một số hệ thống tính toán phân bổ khác, người ta có thể đặt mỗi một đối tượng trong một môi trường hệ điều hành khác nhau.

4.2. Các thuật toán đề xuất

4.2.1. SHA-224[2], SHA-256, SHA-384 và SHA-512

4.2.1.1. Giới thiệu:

Chuẩn SHS đặc tả 5 thuật toán băm an toàn SHA-1, SHA-224, SHA-256, SHA-384 và SHA-512.

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 dữ liệu (word size) được sử dụng thể hiện các tính chất cơ bản của bốn thuật toán băm an toàn.

(2) Đây là thuật toán hàm băm vừa được NIST công nhận thành chuẩn hàm băm an toàn vào 02/2004.

Thuật toán	Kích thước (đơn vị: bit)				Độ an toàn[3] (đơn vị: bit)
	Thông điệp	Khối	Từ	Thông điệp rút gọn	
SHA-1	< 264	512	32	160	80
SHA-224	< 264	512	32	224	112
SHA-256	< 264	512	32	256	128
SHA-384	< 2128	1024	64	384	192
SHA-512	< 2128	1024	64	512	256

Bảng 4.2.1.1 Các tính chất của các thuật toán băm an toàn.

4.2.1.2. Thao tác tiền xử lý

a) Các hàm và các hằng số được dùng trong thuật toán SHA-224[4], SHA-256, SHA-384 và SHA-512 : xem phần phụ lục A.3

b) Mở rộng thông điệp

- SHA-224 và SHA-256 : giống SHA-1
- SHA-384 và SHA-512

Giả sử độ dài của thông điệp M là *l* bit. Thêm bit 1 vào cuối thông điệp, theo sau là k bit 0 (k là số không âm nhỏ nhất sao cho

$$l + 1 + k \equiv 896 \pmod{1024}$$

Sau đó thêm khối 128 bit là biểu diễn nhị phân của l.

Ví dụ, thông điệp (8-bit ASCII) "abc" có độ dài 8x3=24, do đó thông điệp được mở rộng bằng 1 bit "1", 896-(24+1) = 871 bit "0" và chiều dài thông điệp trở thành thông điệp mở rộng 1024 bit.

[3] "Độ an toàn" là việc sử dụng phương pháp tấn công vào thông điệp rút gọn kích thước n, đòi hỏi xử lý xấp xỉ $2^{n/2}$.

[4] Đây là thuật toán hàm băm vừa được NIST công nhận thành chuẩn hàm băm an toàn vào 02/2004.

$$\underbrace{01100001}_{a}\ \underbrace{01100010}_{b}\ \underbrace{01100011}_{c}\ 1\ \overbrace{00..00}^{871}\ \overbrace{00...011000}^{128}\ \underbrace{}_{l=24}$$

Chiều dài của thông điệp mở rộng đã trở thành một bội số của 1024 bit.

c) Phân tích thông điệp đã mở rộng

Đối với SHA-256, giống SHA-1.

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, $M_0^{(i)}$ chứa 64 bit kế tiếp...

d) Khởi tạo giá trị băm

➢ Hằng số của SHA-224 và SHA-256

SHA-224 và SHA-256 sử dụng dãy 64 từ 32 bit là hằng số $K_0^{\{256\}}$, $K_1^{\{256\}}$, …, $K_{63}^{\{256\}}$. Những từ này biểu diễn 32 bit đầu tiên của phần phân số của căn bậc ba của 64 số nguyên tố đầu tiên. Các hằng số được liệt kê phần phụ lục A.1

➢ Hằng số của SHA-384 và SHA-512

SHA-384 và SHA-512 sử dụng cùng dãy 80 từ 64 bit là hằng số $K_0^{\{512\}}$, $K_1^{\{512\}}$, …, $K_{79}^{\{512\}}$. Những từ này biểu diễn 64 bit đầu tiên của phần phân số của căn bậc ba của 80 số nguyên tố đầu tiên. Các hằng số được liệt kê phần phụ luc A .1

➢ Giá trị khởi tạo của các thuật toán SHA : xem phần phụ lục A.2

4.2.1.3. Thuật toán của bước tính toán giá trị băm SHA-224,SHA-256,SHA-384,SHA-512:

a) SHA–224

SHA-224 được sử dụng để băm thông điệp M dài l bit $0 \le l < 2^{64}$. Thuật toán sử dụng :

- Một bảng phân bố thông điệp gồm 64 từ 32-bit.
- 8 biến 32 bit.
- Một giá trị băm gồm 8 từ 32-bit.

Kết quả của SHA-224 là thông điệp rút gọn 224-bit.

Các từ của bảng phân bố thông điệp được ký hiệu W0, W1, ..., W63. 8 biến ký hiệu a, b, c, d, e, f, g và h. Các từ của giá trị băm ký hiệu $H_0^{(i)}$, $H_1^{(i)}$, …, $H_7^{(i)}$, H(0) giữ giá trị băm ban đầu, được thay thế bằng các giá trị băm thành công H(i) sau khi mỗi khối thông điệp được xử lý và kết thúc bằng giá trị băm cuối cùng H(N).Xem chi tiết ở phần phụ lục A.4.2

b) SHA-256

SHA-256 được sử dụng để băm thông điệp M dài l bit $0 \le l < 2^{64}$. Thuật toán sử dụng

- Một bảng phân bố thông điệp gồm 64 từ 32-bit
- 8 biến 32 bit.
- Một giá trị băm gồm 8 từ 32-bit.

Kết quả của SHA-256 là thông điệp rút gọn 256-bit.

Các từ của bảng phân bố thông điệp được ký hiệu W0, W1, ..., W63. 8 biến ký hiệu a, b, c, d, e, f, g và h. Các từ của giá trị băm ký hiệu $H_0^{(i)}$, $H_1^{(i)}$, …, $H_7^{(i)}$, H(0) giữ giá trị băm ban đầu, được thay thế bằng các giá trị băm thành công H(i)

sau khi mỗi khối thông điệp được xử lý và kết thúc bằng giá trị băm cuối cùng H(N)..Xem chi tiết ở phần phu luc A.4.3

c) SHA-384

SHA-384 được sử dụng để băm thông điệp M dài l bit $0 \le l < 2^{128}$.

Thuật toán sử dụng

- Một bảng phân bố thông điệp gồm 80 từ 64-bit
- 8 biến 64 bit.
- Một giá trị băm gồm 8 từ 64-bit.

Kết quả của SHA-384 là thông điệp rút gọn 384-bit.

Các từ của bảng phân bố thông điệp được ký hiệu W0, W1, ..., W79. 8 biến ký hiệu a, b, c, d, e, f, g và h. Các từ của giá trị băm ký hiệu $H_0^{(i)}$, $H_1^{(i)}$, …, $H_7^{(i)}$, H(0) giữ giá trị băm ban đầu, được thay thế bằng các giá trị băm thành công H(i) sau khi mỗi khối thông điệp được xử lý và kết thúc bằng giá trị băm cuối cùng H(N). Xem chi tiết ở phần phu luc A.4.4

d) SHA-512

SHA-512 được sử dụng để băm thông điệp M dài l bit $0 \le l < 2^{128}$.

Thuật toán sử dụng

- Một bảng phân bố thông điệp gồm 80 từ 64-bit
- 8 biến 64 bit.
- Một giá trị băm gồm 8 từ 64-bit.

Kết quả của SHA-512 là thông điệp rút gọn 512-bit.

Các từ của bảng phân bố thông điệp được ký hiệu W0, W1, ..., W79. 8 biến ký hiệu a, b, c, d, e, f, g và h. Các từ của giá trị băm ký hiệu $H_0^{(i)}$, $H_1^{(i)}$, …, $H_7^{(i)}$, H(0) giữ giá trị băm ban đầu, được thay thế bằng các giá trị băm thành công H(i)

sau khi mỗi khối thông điệp được xử lý và kết thúc bằng giá trị băm cuối cùng H(N). Xem chi tiết ở phần phu luc A.4.5

4.2.2. Thuật toán Tiger

4.2.2.1. Giới thiệu

Hiện nay các dòng họ thuật toán băm như gia đình MD, Sneru và SHA-1 đã bị tìm ra các collision hoặc có khả năng bị khám phá ra collision nhờ vào các collision của các thuật toán "hàng xóm".

Thêm vào đó các thuật toán này làm việc kém hiệu quả trên dòng máy tính 64 bits do nó chỉ được thiết kế trên các bộ vi xử lý 32 bits. Mặt khác, thế hệ sắp tới của các bộ vi xử lý có dùng các từ 64-bit (word), và bao gồm luôn cả một loạt các máy DEC Alpha cũng như các bộ vi xử lý thế hệ kế tiếp từ Intel, HP, và IBM. Dường như sẽ hợp lý nếu giả sử việc dùng các vi xử lý trong các ứng dụng nhúng (đa số các hệ thống sẽ dùng các bộ vi xử lý 64-bit trong vòng 5 năm tới).

Do đó, nhằm mục đích cải tiến hiệu suất trên các bộ vi xử lý mới và tạo ra hệ thống băm bảo mật hơn, Ross Anderson và Eli Biham đề xuất thuật toán Tiger, "A Fast New Hash Function" vào năm 1996.

Tiger được đề xuất để cải tiến tốc độ hoặc ít nhất là phải bằng với SHA-1 và gia đình MD. Nó nhanh như SHA-1 trên bộ vi xử lý 32-bit và nhanh hơn khoảng 3 lần trên vi xử lý 64-bit (DEC Alpha), và được cho là nhanh hơn SHA-1 trên các bộ vi xử lý 16-bit vì SHA-1 được thiết kế để chạy trên các máy 32-bit.

Tiger tương thích với các bộ vi xử lý 64 bits và 32 bits, tránh được vấn đề xung đột, vấn đề bội số(chống lại các kiểu tấn công dùng bội số hay các hệ số của các giá trị băm để tìm ra được khoá dùng trong việc băm thông điệp).

Để cải tiến bảo mật, Tiger được thiết kế có chiều dài băm 192 bits. Tuy nhiên để cho tương thích với các ứng dụng hiện tại (sử dụng các thuật toán có chiều dài 128bits, 160 bits), Tiger được cải tiến thành các loại sau :

Tiger/160 có chiều dài là 160 bits. Thuật toán này tương thích với SHA-1.

Tiger/128 có chiều dài là 128 bits. Thuật toán này tương thích với MD4, MD5, RIPE_MD, các hàm băm thuộc họ Snefru và các hàm băm khác.

Các giá trị trên có được bằng cách cắt giảm chiều dài 192 bits của nó.

4.2.2.2. Các bước xử lý trong thuật toán.

Message được chia thành các block 512 bits. Nếu block cuối cùng không đủ chiều dài thì bit 1 sẽ được thêm vào cuối.

Mỗi block được mã hóa riêng biệt với nhau bằng cùng hàm mã hóa.

Hàm mã hóa :

- Giá trị hash key ban đầu được chia thành 3 phần (a, b, c).
- Việc tính toán hash key bao gồm 3 lần thông qua hàm *pass*, và *schedule* theo thứ tự như sau :

pass(a, b, c, 5)
key_schedule
pass(c, a, b, 7)
key_schedule
pass(b, c, a, 9)

Việc tính toán của các hàm được trình bày cụ thể ở phần phụ lục B

- Hàm *pass* thực hiện 8 lần hàm *round.* Hàm *round* sử dụng four S-boxes và các phép hoán vị để bảo đảm rằng các giá trị trả ra sẽ khác nhau trong bất kỳ trường hợp nào khác.
- Trong hàm *schedule*, các giá trị a, b, c sẽ được biến đổi dạng dữ liệu để ngăn cản kẻ tấn công lợi dụng tính thưa của dữ liệu trong 3 vòng lặp qua hàm *pass* trên.

Tất cả các thuật toán được sử dụng có thể chạy song song.

4.2.2.3. Đánh giá thuật toán

Thuật toán này chỉ cần một vùng nhớ rất nhỏ chỉ khoảng 8 Kbyte do S-boxes thật sự không cần lưu trữ.

Nhờ vào việc tính toán hoán đổi liên tục mà hàm băm Tiger rất bảo mật. Nó chống được các phương pháp tấn công "meet-in-the-middle", "birthday".

Tính chất phi tuyến từ các S_box từ 8 bit thành 64 bit (đầu vào 8 bit, đầu ra 64 bit). Điều này tốt hơn nhiều so với việc chỉ kết hợp phép tính cộng với phép XOR (dùng các bit mang theo _ carry bit), và nó ảnh hưởng đến các bit xuất, không chỉ các bit lân cận.

Có một sự lan truyền mạnh, trong mỗi bit thông điệp ảnh hưởng đến tất cả 3 thanh ghi sau 3 vòng _ nhanh hơn những hàm băm khác. Việc lan truyền trong các từ 64 bit (và các S_box 64 bit) thì nhanh hơn nhiều khi các từ ngắn hơn được dùng.

Như ghi chú ở trên, tất cả các biện pháp tấn công đều tấn công trên kết quả của MDx hay Snefru ở một trong các khối trung gian. Tăng giá trị của kết quả trung gian lên 192 bit giúp ngăn chặn các cuộc tấn công này.

Các bảng khoá bảo đảm rằng việc thay đổi một số lượng nhỏ bit trong thông điệp ảnh hưởng đến nhiều bit trong suốt quá trình thông qua hàm pass.

Cùng với việc lan truyền mạnh, nó giúp cho Tiger chống lại các tấn công tương đương với các tấn công khác nhau của Dobbertin trên MD4 (đó việc thay đổi một số bit nào đó trong thông điệp ảnh hưởng đến đa số các bit trong nhiều vòng, và kế đó sự khác biệt nhỏ nào có thể được thực hiện để hủy bỏ trong hàm pass trước đó).

Phép nhân của thanh ghi b trong mỗi vòng cũng góp phần cho việc chống lại các tấn công như vậy, do nó bảo đảm rằng các bit được dùng như đầu vào đến các S_box trong các vòng trước được trộn với những S_box khác, và với cùng S_box với một đầu vào khác. Phép nhân này cũng chống lại các tấn công trên các hàm băm, bởi vì các hằng số là khác nhau ở mỗi vòng.

Hàm feedforward ngăn cản các tấn công "meet-in-the-middle", "birthday" tìm ra các ảnh trước đó của hàm băm (mặc dù sự phức tạp sẽ là 296).

4.2.3. Thuật toán Whirlpool

4.2.3.1. Giới thiệu

Hàm băm Whirlpool được công nhận cùng với phương pháp mã hoá AES là những nền tảng bảo mật mạnh mẽ tại Hội thảo về Bảo Mật NESSIE _ New European Schemes for Signatures, Integrity, and Encryption (các kế hoạch Châu Âu mới cho chữ ký, tính toàn vẹn, và mã hoá) tại Lund, Thụy Điển vào ngày 26/2/2003. Mục đích của dự án là nêu ra những thuật toán bảo mật mới cho thị trường Châu Âu. Vincent Rijmen, trưởng nhóm bảo mật ở văn phòng Cryptomathic.s Belgian ở Leuven, là một trong hai tác giả của thuật toán AES và hàm băm Whirlpool. Nếu AES được chọn vào danh sách mã hoá theo khối 128-bit thì Whirlpool được chọn trong danh sách các hàm băm chống xung đột. Whirlpool được xếp vào chuẩn ISO ISO/IEC 10118-3 cho các hàm băm.

Whirlpool là hàm băm 512-bit được thiết kế dựa trên nguyên lý hoạt động của AES và có thể là một lựa chọn thay cho thuật toán SHA-1, có tính một chiều, chống xung đột thực hiện trên thông điệp có chiều dài ít hơn 2^{256} bit do Paulo S.L.M Barreto và Vincent Rijmen đề xuất năm 2001.

Whirlpool bao gồm việc áp dụng hàm nén nhiều lần, trên nền tảng mã hoá toàn bộ khối 512 bit thông điệp chạy bên dưới, dùng khoá 512 bit. Hàm round và key schedule được thiết kế theo chiến lược Wide Trail. Whirlpool thực thi trên bộ vi xử lý 8-bit và 64-bit thuận lợi đặc biệt do cấu trúc hàm; tuy nhiên, lại không hướng tới bất kỳ nền phần cứng cụ thể nào.

4.2.3.2. Các cơ sở lý thuyết

Các kí hiệu toán học

a) Trường Galois (sự biểu diễn nhị phân)

Ký hiệu trường Galois GF(24) là GF(2)[x] / p4(x) với p4(x) = x4 + x + 1 và trường GF(28) như GF(2)[x] / p8x với p8(x) = x8 + x4 + x3 + x2 + 1.

Đa thức p4(x) và p8(x) là các đa thức chính đầu tiên ở bậc 4 và 8, và được chọn sao cho g(x) = x là phần tử sinh của GF(24) / {0} và GF(28) / {0} tương ứng. Các phần tử thuộc trường Galois được biểu diễn dưới dạng một đa thức $u = \sum_{i=o}^{m-1} ui.xi$ Є GF(2)[x], trong đó ui Є GF(2) với mọi i = 0,…, m-1 sẽ được ghi chú giá trị số $\sum_{i=o}^{m-1} u_i^{2i}$ hay được viết dưới dạng thập lục phân, ví dụ 13x để ký hiệu cho p4(x).

b) Các lớp ma trận

Mmxn [GF(28)] ký hiệu cho tập các ma trận mxn dựa trên trường Galois GF(28). Cir(a_0, a_1, …, a_{m-1}) ký hiệu cho ma trận xoay vòng mxm mà dòng đầu tiên bao gồm a_0, a_1, …, a_{m-1}, tức là :

$$\text{Cir}(a_0, a_1, \ldots, a_{m-1}) \equiv \begin{bmatrix} a_0 & , a_1, \ldots , a_{m-1} \\ a_{m-1}, & a_0, \ldots , a_{m-2} \\ \cdots & \cdots \\ a_1 & , a_2, \ldots , a_0 \end{bmatrix}$$

Đơn giản Cir(a_0, a_1, …, a_{m-1}) = c $\Leftrightarrow$ $c_{ij} = a_{(j-i)} \bmod m$, $0 \le i, j \le m-1$.

c) Mã MDS (MDS code - Maximal Distance Separable code)

Khoảng cách Hamming : giữa hai vectơ u và v từ không gian vectơ n chiều GF(2p)n bằng số tọa độ mà u và v khác nhau.

Trọng lượng Hamming wh(a) : của phần tử a Є GF(2p)n là khoảng cách Hamming giữa a và vectơ không trong trường GF(2p)n tức là số thành phần khác không của a.

Mã tuyến tính [n,k,d] : trên trường GF(2p)n là không gian con k chiều của không gian vectơ (GF(2p))n trong đó khoảng cách Hamming giữa 2 vectơ không gian con phân biệt bất kỳ ít nhất là d (và d là số lớn nhất với đặc tính này).

Ma trận sinh G : cho mã [n,k,d] tuyến tính C là ma trận kxn mà các dòng của ma trận này hình thành nên nền tảng cho C. Một ma trận phát sinh ở dạng chuẩn hay phân cấp bậc, nếu nó có dạng G = [Ikxk Akx(n-k)] trong đó Ikxk là ma trận định vị bậc k. Viết đơn giản là G = [IA] bỏ đi các chỉ mục khi các chiều của ma trận không liên quan đến thảo luận hay đã rõ trong ngữ cảnh đã cho.

Mã tuyến tính [n,k,d] tuân theo ràng buộc một đầu là d ≤ n – k + 1. Mã gặp giới hạn này tức là d = n-k+1 gọi là mã có thể phân rã khoảng cách lớn nhất. Mã [n,k,d] tuyến tính C với ma trận phát sinh G = [Ikxk Akx(n-k)] là MDS khi và chỉ khi mọi ma trận con vuông hình thành từ các dòng và các cột của A thì không kỳ dị.

d) Các thuộc tính mật mã

Tích của m biến Bool phân biệt được gọi là tích bậc m của các biến. Mỗi hàm Bool f : GF(2n) → GF(2) có thể được viết như một tổng trên trường GF(2) của các tích bậc m phân biệt của các đối số của nó, 0≤ m ≤n, và được gọi là dạng chuẩn đại số của f.

Bậc dạng chuẩn đại số của f, kí hiệu là v(f) là bậc lớn nhất của các số hạng xuất hiện trong dạng chuẩn đại số của f. Một hàm Bool tuyến tính là hàm Bool có bậc bằng 1, tức là dạng chuẩn đại số của nó chỉ bao gồm các đối số đơn. Cho trước $\alpha \in GF(2)^n$, ký hiệu hàm Bool tuyến tính là l_α: GF(2)n → GF(2) bao gồm tổng các bit đối số được chọn bởi các bit của α :

$$l_\alpha(x) \equiv \bigoplus_{i=0}^{n-1} \alpha_i \cdot x_i.$$

Một ánh xạ S: GF(2n) → GF(2 n), x S[x] được gọi là S-box. Một S-box cũng có thể được xem như là một ánh xạ S : GF(2n) → GF(2 n), do đó được mô tả dưới dạng các hàm Bool thành phần của nó s_i : GF(2n) → GF(2), $0 \le i \le n-1$ tức là $S[x] = (s_0(x),\ldots,s_{n-1}(x))$.

Bậc của một S-box S, ký hiệu là vS là bậc nhỏ nhất trên tất cả các phép hợp các thành phần của S:

$$\nu_S \equiv \min_{\alpha \in \mathrm{GF}(2)^n} \{\nu(l_\alpha \circ S)\}.$$

Tham số δ của một S-box S được định nghĩa như sau:

$$\delta_S \equiv 2^{-n} \cdot \max_{a \neq 0,\, b} \#\{c \in \mathrm{GF}(2^n) | S[c \oplus a] \oplus S[c] = b\}.$$

Giá trị 2n. δ được gọi là hệ số đồng dạng phân biệt của S.

Hệ số tương quan (correlation) c(f,g) giữa 2 hàm Bool f và g được định nghĩa như sau:

$$c(f,g) \equiv 2^{1-n} \cdot \#\{x | f(x) = g(x)\} - 1.$$

Cực trị (the extreme value) của hệ số tương quan giữa các hàm tuyến tính của các bit đầu vào và các hàm tuyến tính của các bit đầu ra được gọi là độ dốc của S (the bias of S).

Tham số λ của S-box S được định nghĩa là giá trị tuyệt đối cũa độ dốc:

$$\lambda_S \equiv \max_{(i,j) \neq (0,0)} |c(l_i, l_j \circ S)|.$$

Số nhánh B (the branch number B) của ánh xạ tuyến tính □ : GF(2n)k → GF(2 n)m

$$\mathcal{B}(\theta) \equiv \min_{a \neq 0} \{\mathrm{w_h}(a) + \mathrm{w_h}(\theta(a))\}.$$

Cho mã tuyến tính a[k+m,k,d] trên trường GF(2p) với ma trận phát sinh (generator matrix) G = [IkxkMkxm], ánh xạ tuyến tính □ : GF(2n)k → GF(2 n)m được định nghĩa bởi θ(a) = a.M có nhánh B(θ) = d; nếu mã là MDS, một ánh xạ như vậy được gọi là ánh xạ lan truyền tối ưu (optimal diffusion mapping).

e) Ký hiệu khác

Cho một dãy các hàm fm, fm+1,…, fn-1, fn, nếu m x ≤ n, chúng ta dùng ký hiệu

$$\bigcirc_{r=m}^{n} f_r \equiv f_m \circ f_{m+1} \circ \cdots \circ f_{n-1} \circ f_n,$$

và

$$\bigcirc_{m}^{r=n} f_r \equiv f_n \circ f_{n-1} \circ \cdots \circ f_{m+1} \circ f_m$$

nếu m > n cả hai biểu thức thay thế cho ánh xạ đơn vị.

❖ **Các ánh xạ và hằng số thành phần tạo nên Whirlpool**

Whirlpool nguyên thủy chính là hàm băm Merkle dựa trên mã hoá toàn bộ khối (dedicated block cipher), W, hoạt động trên một trạng thái băm (state hash) 512 bit dùng một trạng thái khoá mắc xích (chained key state), cả hai đều xuất phát từ dữ liệu nhập. Sau đây là các ánh xạ và các hằng số thành phần thiết lập nên Whirlpool và đặc tả hàm băm Whirpool.

a) Nhập và xuất

Trạng thái băm (state hash) bên trong được xem như là một ma trận M8x8[GF(28)]. Do đó, các khối dữ liệu 512 bit (được mô tả bên ngoài như các mảng byte bằng cách nhóm các bit lần lượt thành những đoạn 8 bit) phải được ánh xạ đến hay từ định dạng ma trận. Việc này được thực hiện bởi hàm

$$\mu : GF(28)64 \rightarrow M8x8[GF(28)]$$

và nghịch đảo của nó là : $\mu(a) = b \Leftrightarrow b_{ij} = a_{8i+j}$, $0 \le i, j \le 7$

b) Lớp phi tuyến γ

Hàm γ: M8x8[GF(28)] → M8x8[GF(28)] bao gồm việc áp dụng song song của S-box S : GF(28) → GF(28), x S[x] cho tất cả các byte của từng đối số :

$$\gamma(a) = b \Leftrightarrow bij = S[a_{ij}],\ 0 \le i, j \le 7$$

c) Hoán vị theo chu kỳ π

Hàm hoán vị π: M8x8[GF(28)] → M8x8[GF(28)] xoay theo chu kỳ mỗi cột của đối số (argument) của nó một cách độc lập, để các phần tử thuộc cột j được xoay j vị trí trên cột j (xoay theo dòng):

$$\pi(a) = \pi \Leftrightarrow b_{ij} = a_{(i-j) \bmod 8, j},\ 0 \le i, j \le 7$$

Mục đích của π là phát tán các byte của mỗi dòng giữa các dòng với nhau.

d) Lớp lan truyền tuyến tính θ

Lớp lan truyền tuyến tính θ: M8x8[GF(28)] → M8x8[GF(28)] là một ánh xạ tuyến tính dựa trên mã MDS [n,k,d] với n = 16, k = 8, d = 9 với ma trận sinh GC = [IC] trong đó C = Cir(01x, 01x, 04x, 01x, 08x, 05x, 02x, 09x) tức là :

$$C = \begin{bmatrix} 01_x & 01_x & 04_x & 01_x & 08_x & 05_x & 02_x & 09_x \\ 09_x & 01_x & 01_x & 04_x & 01_x & 08_x & 05_x & 02_x \\ 02_x & 09_x & 01_x & 01_x & 04_x & 01_x & 08_x & 05_x \\ 05_x & 02_x & 09_x & 01_x & 01_x & 04_x & 01_x & 08_x \\ 08_x & 05_x & 02_x & 09_x & 01_x & 01_x & 04_x & 01_x \\ 01_x & 08_x & 05_x & 02_x & 09_x & 01_x & 01_x & 04_x \\ 04_x & 01_x & 08_x & 05_x & 02_x & 09_x & 01_x & 01_x \\ 01_x & 04_x & 01_x & 08_x & 05_x & 02_x & 09_x & 01_x \end{bmatrix},$$

để θ(a) = b ⇔ b = a. C. Kết quả của θ là trộn các byte trong mỗi dòng trạng thái.

e) Phép cộng khoá σ[k]

Phép cộng khoá quan hệ σ[k] : M8x8[GF(28)] → M8x8[GF(28)] bao gồm phép XOR của ma trận khoá k Є M8x8[GF(28)] :

$$\sigma[k](a) = b \Leftrightarrow b_{ij} = a_{ij} \oplus k_{ij},\ 0 \le i, j \le 7$$

Ánh xạ này cũng cho biết các hằng số vòng trong bảng xếp lịch khoá.

f) Hằng số vòng cr

Hằng số vòng cho vòng thứ r, r>0, là một ma trận cr Є M8x8[GF(28)], được định nghĩa như sau:

$$\begin{aligned} c^r_{0j} &\equiv S[8(r-1)+j],\ 0 \leqslant j \leqslant 7, \\ c^r_{ij} &\equiv 0, \qquad\qquad 1 \leqslant i \leqslant 7, 0 \leqslant j \leqslant 7. \end{aligned}$$

g) Hàm vòng p[k]

Hàm vòng thứ r là ánh xạ hợp (composite mapping)

$$p[k] : M8x8[GF(28)] \rightarrow M8x8[GF(28)]$$

tham số hoá bởi ma trận khoá k Є M8x8[GF(28)] và được cho bởi:

$$p[k] \equiv \sigma[k] \circ \theta \circ \pi \circ \gamma$$

h) Bảng xếp lịch khoá

Bảng xếp lịch khoá được mở rộng khoá mật mã 512 bit K Є M8x8[GF(28)] thành các khoá vòng K^0,…, K^R :

$$K^0 = K,$$

$$K^r = \rho[c^r](K^{r-1}),\ r > 0.$$

i) Mật mã khối nội W

Mật mã toàn bộ khối 512 bit W[K] : M8x8[GF(28)] → M8x8[GF(28)], tham số hoá bởi khoá mật mã nội 512 bit K, được định nghĩa như sau:

$$W[K] = \left(\bigcirc_{1}^{r=R} \rho[K^r] \right) \circ \sigma[K^0],$$

ở đó các khoá vòng K^0,…, K^R dẫn xuất từ K bởi bảng xếp lịch khóa. Số vòng ngẫu nhiên là R = 10.

4.2.3.3. Các bước xử lý trong thuật toán.

Đầu vào : dữ liệu cần băm M

Đầu ra : giá trị băm (512 bit)

Thuật toán Whirlpool gồm 3 bước:

- Bước 1

Thêm bit : thêm vào bit 0 và 1 để chiều dài của M (tính bằng bit) là bội số của 256, ta được M'

- Bước 2

Chia M' thành t khối : $m_1, m_2,\ldots, m_t$

Băm các khối m_i ($1 \le i \le t$) dùng mật mã toàn bộ khối 512-bit W như sau :

$$\eta_i = \mu(m_i),$$

$$H_0 = \mu(IV),$$

$$H_i = W[H_{i-1}](\eta_i) \oplus H_{i-1} \oplus \eta_i, \quad 1 \le i \le t$$

Trong đó:

IV (Initialisation Vector) là một chuỗi 512-bit '0'

μ: là một ánh xạ GF(28)64 → M8x8[GF(28)] và nghịch đảo của nó là :

$$\mu(a) = b \Leftrightarrow b_{ij} = a_{8i+j},\ 0 \le i, j \le 7$$

- Bước 3: ánh xạ giá trị băm ngược lại chuỗi bit

$$\text{WHIRLPOOL}(M) \equiv \mu^{-1}(H^t).$$

Mật mã toàn bộ khối 512-bit W là ánh xạ W[K]:

W[K]: M8x8[GF(28)] → M8x8[GF(28)]

$$W[K] = \left(\bigcirc_{1}^{r=R} \rho[K^r]\right) \circ \sigma[K^0],$$

4.2.3.4. Đánh giá hàm băm Whirpool

➢ Ưu điểm:

- Có khả năng mở rộng phạm vi sử dụng hơn đa số các hàm băm khác.
- Hiệu quả trên đa số phần cứng (hỗ trợ các bộ xử lý 8 bit, 32 bit, 64 bit).
- Không yêu cầu không gian lưu trữ quá mức cho cả mã lẫn các S-box.
- Được cài đặt hiệu quả trên môi trường có những ràng buộc như các thẻ thông minh, các thiết bị cầm tay, …; và hoạt động với hiệu suất rất cao khi hoạt động trên các bộ nhớ cache lớn hơn của các bộ xử lý hiện đại.
- Chỉ sử dụng những lệnh đơn giản được hỗ trợ sẵn trong bộ xử lý máy tính, không dùng những lệnh không thông thường hay từ các tiện ích .
- Chiều dài giá trị băm dài làm tăng khả năng chống lại các tấn công.

➢ Khuyết điểm

- Thuật toán phức tạp

4.2.4. So sánh SHA-1, Tiger, Whirlpool

SHA-1	Tiger	Whirlpool
SHA-1 được đề xuất vào tháng 4 năm 1995.	Ross Anderson và Eli Biham đề xuất thuật toán Tiger, "A Fast New Hash Function" vào năm 1996.	Paulo S.L.M Barreto và Vincent Rijmen đề xuất năm 2001.
Được chứng nhận bởi Viện Công nghệ và Tiêu chuẩn quốc gia (NIST) Mỹ		Được công nhận là nền tảng bảo mật mạnh mẽ tại Hội thảo về Bảo Mật NESSIE _ New European Schemes for Signatures, Integrity, and Encryption tại Lund, Thụy

		Điển vào ngày 26/2/2003
Chiều dài băm **160** bits	Chiều dài băm **192** bits	Chiều dài băm **512** bits
Có thể bị tấn công bằng phương pháp "meet-in-the-middle", "birthday"	Không bị tấn công bằng phương pháp "meet-in-the-middle", "birthday"	Là hàm băm chống xung đột
Hiện nay đã bị tấn công bởi 3 nhà nghiên cứu Trung Quốc.	Hiện nay chưa bị tấn công	Hiện nay chưa bị tấn công
Kém hiệu quả trên dòng máy 64 bits	Nó nhanh như SHA-1 trên bộ vi xử lý 32-bits và nhanh hơn khoảng 3 lần trên vi xử lý 64-bits (DEC Alpha), và được cho là nhanh hơn SHA-1 trên các bộ vi xử lý 16-bits.	Whirlpool thực thi trên bộ vi xử lý 8-bit và 64-bit thuận lợi đặc biệt.
	Cần vùng nhớ rất nhỏ (8KB)	Không yêu cầu không gian lưu trữ lớn.
	Thuật toán sử dụng four S_boxes	Thuật toán khá phức tạp.
		Whirlpool được xếp vào chuẩn ISO ISO/IEC 10118-3 cho các hàm băm.
		Không hướng tới bất kỳ nền phần cứng cụ thể nào do đó có thể mở rộng phạm vi sử dụng.
		Cài đặt hiệu quả trên môi trường có ràng buộc (thẻ thông minh, thiết bị cầm tay).

Bảnng 5.3.2.2 So sánh SHA-1, Tiger, Whirlpool

Chương 5.Cài đặt thử nghiệm

✍ ***Chương 5 :*** *Chương này sẽ trình bày về yêu cầu phát triển đề tài và chương trình cài đặt thử nghiệm.*

5.1. Yêu cầu chức năng chương trình

Xây dựng cơ chế bảo mật mới cho hệ quản trị cơ sở dữ liệu MySQL trên cả môi trường Windows và Linux bằng cách thay đổi thuật toán mã hoá.

Chỉ can thiệp vào công việc mã hoá password, tất cả các chức năng khác của HQT MySQL không được khác biệt so với ban đầu.

5.2. Chương trình cài đặt

5.2.1. Hướng dẫn cài đặt MySQL từ source code

5.2.1.1. cài đặt trên môi trường Window

a) Yêu cầu cấu hình và phần mềm :

- Trình biên dịch VC++ 6.0 (phiên bản 4 hoặc 5 và pre-processor package). Pre-processor package rất cần thiết cho macro assembler.
- Dĩa trống khoảng 45M.
- 64MB RAM.

b) Cài đặt:

Chú ý rằng các file VC++ workspace từ MySQL 4.1 thì tương thích với Microsoft Visual Studio 6.0 và các phiên bản về sau (7.0/.NET) và được kiểm tra bởi đội ngũ MySQL AB trước mỗi phiên bản.

Các bước biên dịch chương trình:

- Tạo thư mục gốc làm việc (ví dụ C:\workdir).

- Giải nén source trong thư mục vừa tạo ra ở trên. Có thể sử dụng winzip hay một phần mềm khác để giải nén file .zip.
- Khởi động trình biên dịch VC++.
- Trong File menu, chọn Open Workspace.
- Mở mysql.dsw workspace, người dùng có thể tìm thấy trong thư mục làm việc.
- Từ Build menu, chọn Set Active Configuration menu.
- Chọn mysqld – Win32 Debug và chọn OK trong hộp thoại Set Active Project Configuration.
- Nhấn F7 để bắt đầu xây dựng debug server, libraries, và các ứng dụng client.
- Hoàn thành release versions theo cùng cách trên.
- Các Debug versions của chương trình và thư viện được để trong thư mục client_debug và lib_debug directories. Release versions của chương trình được để trong thư mục client_release and lib_release directories. Ta có thể xây dựng cả debug và release versions bằng cách chọn Build All từ Build menu.
- Kiểm tra server. Mặc định server chọn thư mục gốc là c:\mysql và thư mục dữ liệu là C:\mysql\data. Nếu người dùng muốn kiểm tra server bằng cách sử dụng cây thư mục gốc và dữ liệu về thư mục như là thư mục cơ bản và thư mục dữ liệu, người dùng có thể chỉ cho server biết đường dẫn của chúng. Người dùng có thể thực hiện bằng câu lệnh chọn lựa --basedir và --datadir, hoặc là chỉ định các lựa chọn thích hợp trong option file (file my.ini trong thư mục Window hoặc C:\my.cnf). Nếu các file này đã tồn tại mà người dùng muốn sử dụng, người dùng có thể chỉ rõ đường dẫn của nó.

- Khởi động server trong thư mục client_release hoặc client_debug, tùy thuộc server và người dùng sử dụng.

  khi server dang chạy một mình hoặc như là dịch vụ dựa trên cấu hình của người dùng, cố gắng kết nối với nó bằng các câu lệnh tương tác thiết thực mà tồn tại trong thư mục client_release or client_debug.

Khi người dùng thõa mãn với chương trình mà người dùng vừa xây dựng thì hãy ngưng server. Sau đó cài đặt MySQL như sau :

1. Tạo thư mục mà người dùng muốn cài đặt MySQL. Chẳng hạn như C:\mysql. Sau đó tạo các thư mục con (bin, data, share, scripts). Người dùng có thể dùng các câu lệnh sau trong dos :

   C:\> mkdir C:\mysql

   C:\> mkdir C:\mysql\bin

   C:\> mkdir C:\mysql\data

   C:\> mkdir C:\mysql\share

   C:\> mkdir C:\mysql\scripts

   Nếu người dùng muốn biên dich các client khác sau đó kết nối chúng với MySQL, người dùng có thể tạo ra vài thư mục phụ :

   C:\> mkdir C:\mysql\include

   C:\> mkdir C:\mysql\lib

   C:\> mkdir C:\mysql\lib\debug

   C:\> mkdir C:\mysql\lib\opt

   Nếu người dùng muốn chuẩn MySQL, tạo thư mục sau :

   C:\> mkdir C:\mysql\sql-bench

   Benchmarking đòi hỏi hỗ trợ Perl.

2. Từ thư mục biên dịch From the workdir directory, sao chép vào thư mục C:\mysql directory các thư mục sau:

```
C:\> cd \workdir
C:\workdir> copy client_release\*.exe C:\mysql\bin
C:\workdir>copy client_debug\mysqld.exe C:\mysql\bin\mysqld-debug.exe
C:\workdir> xcopy scripts\*.* C:\mysql\scripts /E
C:\workdir> xcopy share\*.* C:\mysql\share /E
```

Nếu người dùng muốn biên dich các client khác sau đó kết nối chúng với MySQL, người dùng có thể sao chép vài file thư viện và header :

```
C:\workdir> copy lib_debug\mysqlclient.lib C:\mysql\lib\debug
C:\workdir> copy lib_debug\libmysql.* C:\mysql\lib\debug
C:\workdir> copy lib_debug\zlib.* C:\mysql\lib\debug
C:\workdir> copy lib_release\mysqlclient.lib C:\mysql\lib\opt
C:\workdir> copy lib_release\libmysql.* C:\mysql\lib\opt
C:\workdir> copy lib_release\zlib.* C:\mysql\lib\opt
C:\workdir> copy include\*.h C:\mysql\include
C:\workdir> copy libmysql\libmysql.def C:\mysql\include
```

Nếu người dùng muốn chuẩn MySQL, tạo thư mục sau :

```
C:\workdir> xcopy sql-bench\*.* C:\mysql\bench /E
```

5.2.1.2. cài đặt trên môi trường Linux

a) Yêu cầu cấu hình và phần mềm :

- Ít nhất là phải dùng Linux 2.0
- Chương trình giải nén GNU

- Trình biên dịch ANSI C++ 6.0 gcc.2.95.2 (phiên bản 4 hoặn 5). Pre-processor package rất cần thiết cho macro assembler
- Dĩa trống khoảng 45M.
- 64MB RAM.
- A working ANSI C++ compiler. **gcc** 2.95.2 hoặc các phiên bản sau, **egcs** 1.0.2 hoặc các phiên bản sau, hay **egcs 2.91.66**, SGI C++, và SunPro C++
- Chương trình **make**. GNU **make** thường được sử dụng. Nếu có vấn đề có

Các bước biên dịch chương trình :

Các dòng lệnh cơ bản để thực hiện biên dich chương trình và cài đặt mysql trong môi trường hệ điều hành Linux. Trước tiên người dùng cần phải mở một terminal và thực hiên các dòng lệnh sau.

```
shell> groupadd mysql
shell> useradd -g mysql mysql
shell> gunzip < mysql-VERSION.tar.gz | tar -xvf –
shell> cd mysql-VERSION
shell> ./configure --prefix=/usr/local/mysql
shell> make
shell> make install
shell> cp support-files/my-medium.cnf /etc/my.cnf
shell> cd /usr/local/mysql
shell> bin/mysql_install_db --user=mysql
shell> chown -R root .
shell> chown -R mysql var
shell> chgrp -R mysql .
shell> bin/mysqld_safe --user=mysql &
```

5.2.2. Hướng dẫn thực thi chương trình

5.2.2.1. Trên môi trường Window

Chạy file mysqld.exe trong thư mục bin

Đăng nhập với quyền quản trị bằng cách chạy file mysql.exe

Để thêm mới một account, thực hiện câu lệnh "insert":

vd :

```
insert into mysql.user(host,user,password) values ("localhost","test",password("test"));
```

Đăng nhập với account đã được cấp bằng câu lệnh :

mysql -u<username> -p (ấn enter)

<password>

Sau đó thực hiện các thao tác trên csdl.

5.2.2.2. Trên môi trường Linux

Chạy file mysqld_safe trong thư mục bin với dòng lệnh sau :

#bin/mysqld_safe --user=mysql &

Chạy file mysql.exe bằng cách

#cd bin

#./mysql

Các bước đăng nhập thực hiện tương tự như bên window.

5.3. Giới thiệu chương trình cài đặt

5.3.1. Chương trình chính

a) Giới thiệu sơ lược mã nguồn mở MySQL

Chương trình gồm 46 project, chứa khoảng 5000 file. Trong đó có một số project chính: client, mysys, vio, sql, myisam.

Việc mã hoá password tập trung ở các project sau:

- Trong môi trường Windows : libmysql, sql, include
- Trong môi trường Linux : libmysql, libmysqld, libmysql_r, sql, include

Cơ sở dữ liệu của hệ thống được tạo ra trong các project t:

- Trong môi trường Windows : dbd,mysqltest, mysql_test_run_newfiles .
- Trong môi trường Linux : mysqltest, scripts.

Việc mã hoá password chủ yếu được thực hiện bằng các hàm trong file password.c.

Các hàm chính trong file Password.cpp

Không có password được chuyển giữa client và server trong kết nối

Không có password được lưu trong mysql ở dạng chuẩn

Quá trình chứng thực và kiểm tra kết nội như sau:

- Một chuỗi ngẫu nhiên được phát sinh và gởi cho client.
- Client phát sinh ra một chuỗi mới từ giá trị phát sinh ngẫu nhiên và các giá trị băm từ password +chuỗi vừa gởi.
- Password được lưu (trong user.password) bằng cách sử dụng hàm PASSWORD() trong mysql.

Đây là file .c bởi vì nó được sử dụng trong libmysqlclient, nó hoàn toàn trong C. (chúng ta cần nó để đem lại sự đa dạng cho hệ thống)

Sự chứng thực thể hiện cụ thể như sau:

SEVER	CLIENT
- public_seed=tạo chuỗi ngẫu nhiên	
- gởi chuỗi public_seed cho client	
	- nhận chuỗi public-seed từ server
	- Băm chuỗi password lần 1

	- Băm chuỗi password lần 2
	- nối chuỗi public_seed và hash_stage2 BĂM chuỗi vừa tạo ra -Rely=XOR giá trị vừa tạo ra và hash_stage1
	-gởi giá trị rely cho server
- nhận giá trị rely từ client	
- Nối chuỗi public_seed và hash_stage2 (được lưu trong CSDL) - BĂM chuỗi vừa tạo ra - hash_stage2=XOR giá trị vừa tạo ra và hash_stage1 - candidate_hash2=BĂM giá trị hash_stage1 - so sánh candidate_hash2 và hash_stage2 (trong CSDL)	

- **Tên hàm:** create_random_string

Khai báo : create_random_string(char *to, uint length, struct rand_struct *rand_st)

Ý nghĩa: phát sinh chuỗi ngẫu nhiên

INPUT:

length: số kí tự được phát sinh ngẫu nhiên trong buffer

OUTPUT:

to:buffer được phát sinh, có chiều dài = length+ 1bytes(them kí tự '\0')

- **Tên hàm:** octet2hex

Khai báo: octet2hex(char *to, const uint8 *str, uint len)

Ý nghĩa: chuyển chuỗi từ hệ bát phân sang thập lục phân

INPUT:

str: chuỗi cần chuyển.

len: chiều dài chuỗi.

OUTPUT:

to: kết quả sau khi chuyển. có chiều dài = 2*len+1 bytes

- **Tên hàm:** hex2octet

Khai báo: hex2octet(uint8 *to, const char *str, uint len)

Ý nghĩa: chuyển chuỗi từ hệ thập lục phân sang hệ bát phân

INPUT:

str: chuỗi cần chuyển.

len: chiều dài chuỗi, length là bội số của 2.

OUTPUT:

to: kết quả sau khi chuyển. có chiều dài = len/2

- **Tên hàm:** my_crypt(char *to, const uchar *s1, const uchar *s2, uint len)

Ý nghĩa: hàm này được dung trong việc chứng thực password

Lưu ý : XOR(s1, XOR(s1, s2)) == s2, XOR(s1, s2) == XOR(s2, s1)

INPUT:

s1,s2: chuỗi nhập vào(có chiều dài bằng nhau)

len: chiều dài chuỗi s1,s2

OUTPUT:

to: kết quả việc thực hiện phép XOR 2 chuỗi s1,s2

- **Tên hàm:** make_scrambled_password

Khai báo: make_scrambled_password(char *to, const char *password)

Ý nghĩa: hàm này dùng để mã hoá password

INPUT:

Password: chuỗi password được nhập vào, có thể là NULL

OUTPUT:

to: kết quả sau khi BĂM ở dạng thập lục phân

Ý tưởng:

Dùng thuật toán SHA1 băm chuỗi password 2 lần

Thêm kí tự "*" vào đầu chuỗi

Chuyển kết quả sang hệ thập lục phân

- **Tên hàm:** scramble

Khai báo: scramble(char *to, const char *message, const char *password)

Ý nghĩa: client sử dụng hàm này để tạo ra giá trị rely trả về cho server.

INPUT:

password: password của người dùng.

message: chuỗi ngẫu nhiên.

OUTPUT:

to: chính là giá trị rely, sẽ được gởi đến server để chứng thực password.

Ý tưởng:

recv(public_seed)

hash_stage1=sha1("password")

hash_stage2=sha1(hash_stage1)

reply=xor(hash_stage1, sha1(public_seed,hash_stage2)

- **Tên hàm:** check_scramble

Khai báo: check_scramble(const char *scramble, const char *message, const uint8*hash_stage2)

Ý nghĩa: dùng để chứng thực password.

INPUT:

Scramble: giá trị reply của client,được tạo ra từ hàm scramble()

Message: chuỗi ngẫu nhiên được phát sinh ban đầu

hash_stage2: chuỗi password đã được mã hoá trong cột password của bảng user trong cơ sở dữ liệu mysql.

OUTPUT:

=0: nếu password đúng

khác 0: nếu password sai

Ý tưởng:

recv(reply)

hash_stage1=xor(reply, sha1(public_seed,hash_stage2))

candidate_hash2=sha1(hash_stage1)

check(candidate_hash2==hash_stage2)

- **Tên hàm:** get_salt_from_password

Khai báo: get_salt_from_password(uint8 *hash_stage2, const char *password)

Ý nghĩa: chuyển chuỗi từ dạng thập lục phân sang dạng nhị phân

INPUT:

password: giá trị trong cột password của bảng user trong CSDL mysql

OUTPUT:

hash_stage2: giá trị sau khi đã chuyển sang dạng nhị phân

- **Tên hàm:** make_password_from_salt

Khai báo: make_password_from_salt(char *to, const uint8 *hash_stage2)

Ý nghĩa: chuyển chuỗi từ dạng nhị phân sang dạng thập lục phân

INPUT:

hash_stage2: chuỗi nhị phân

OUTPUT:

to: chuỗi sau khi đã chuyển sang thập lục phân

Các hàm chính trong file SHA1.cpp

- **Tên hàm:** sha1_reset

Khai báo: sha1_reset(SHA1_CONTEXT *context)

Ý nghĩa: Khởi tạo struct SHA1_CONTEXT để chuẩn bị cho việc BĂM

INPUT:

context: giá trị khởi tạo ban đầu

OUTPUT:

SHA_SUCCESS: thành công

Khác SHA_SUCCESS: bị lỗi

- **Tên hàm:** sha1_input

Khai báo: sha1_input(SHA1_CONTEXT *context, const uint8 *message_array, unsigned length)

Ý nghĩa: nhập dữ liệu vào struct

INPUT:

Context: nội dung của biến context sau khi đã cập nhật

message_array: mảng thông điệp

length: chiều dài mảng thông điệp

OUTPUT:

SHA_SUCCESS: thành công

!= SHA_SUCCESS: bị lỗi

- **Tên hàm:** sha1_result

Khai báo: sha1_result(SHA1_CONTEXT *context, uint8 Message_Digest[SHA1_HASH_SIZE])

Ý nghĩa: chuyển chuỗi từ dạng nhị phân sang dạng thập lục phân

INPUT:

Context: giá trị dung để BĂM

OUTPUT:

Message_Digest: kết quả sau khi BĂM

b) Cài đặt

Để tạo ra cơ chế mã hoá password mới, cần phải can thiệp vào source code. Các công việc cụ thể cần làm như sau :

- Cài đặt thuật toán mới
- Thay đổi các tham số và hằng số liên quan.
- Thay đổi nội dung hàm mã hoá password.
- Thay đổi cấu trúc các bảng lưu trữ password (vì đầu ra của thuật toán SHA-1 là 160 bit, còn thuật toán Tiger là 192 bit và whirlpool là 512 bit).

5.3.2. Chương trình phụ

5.3.2.1. Giới thiệu

Đây là chương trình dùng để so sánh giữa 3 thuật toán SHA-1, Tiger và Whirlpool. Nó sẽ cho thấy kết quả đầu ra, thời gian thực thi các thuật toán khi có cùng giá trị đầu vào

Hình 5.3.2 Chương trình Hash Function

5.3.2.2. Cài đặt

Chương trình gồm có các các hàm băm của các thuật toán.

Mỗi thuật toán có cấu trúc lưu thông tin trong quá trình băm:

Mỗi thuật toán được cài đặt gồm 3 hàm chính :

Hàm	SHA-1	Tiger	Whirlpool	diễn giải
Hàm khởi tạo	sha1_reset	tiger_init	WhirlpoolInit	Khởi tạo các hằng số ban đầu
Hàm thêm thông điệp	sha1_input	tiger_process	WhirlpoolAdd	Lưu giá trị của thông điệp
Hàm trả về kết quả	sha1_result	tiger_done	WhirlpoolFinalize	Thực hiện băm và trả về kết sau khi băm.

Bảnng 5.3.2.2 Các hàm chính trong SHA-1, Tiger,Whirlpool

5.4. Kết quả thực nghiệm

Cơ chế bảo mật được xây dựng theo hướng tiếp cận : thay đổi hàm mã hóa password. Từ thuật toán mã hóa password của MySQL (SHA-1) ta thay đổi bằng một thuật toán khác (Tiger, Whirlpool).

Các kết quả thực nghiệm:

STT	Chiều dài chuỗi (bit)	Thời gian BĂM (ms)		
		SHA-1	Tiger	Whirpool
1	2445168 bits	30	50	110
2	4263216 bits	60	90	150
3	8688320 bits	121	190	290
4	19494960 bits	280	441	660
5	33590768 bits	471	751	1152
6	67013152 bits	962	1582	2394

Bảng B.1. Máy CPU Celeron 950MHz, SDRAM 128 MB, HDD 40GB, Processor 32bit

STT	Chiều dài chuỗi (bit)	Thời gian BĂM (ms)		
		SHA-1	Tiger	Whirpool
1	2445168 bits	31	47	125
2	4263216 bits	47	79	234
3	8688320 bits	78	172	454
4	19494960 bits	188	359	1047
5	33590768 bits	312	656	1757
6	67013152 bits	671	1297	3719

Bảng B.2. Máy CPU PentiumIV 1,5 GHz, DDRAM 384MB, HDD 30 GB, Processor 32bit

STT	Chiều dài chuỗi (bit)	Thời gian BĂM (ms)		
		SHA-1	Tiger	Whirpool
1	2445168 bits	10	30	70
2	4263216 bits	30	50	130
3	8688320 bits	60	100	260
4	19494960 bits	120	221	591
5	33590768 bits	211	390	1022
6	67013152 bits	410	781	2053

Bảng B.3. Máy CPU PentiumIV 2.26 GHz, DDRAM 225MB, HDD 40GB, Processor 32bit

STT	Chiều dài chuỗi (bit)	Thời gian BĂM (ms)		
		SHA-1	Tiger	Whirpool
1	2445168 bits	16	31	47
2	4263216 bits	31	47	78
3	8688320 bits	62	94	172
4	19494960 bits	141	218	368
5	33590768 bits	234	360	640
6	67013152 bits	409	703	1281

Bảng B.4. Máy CPU PentiumIV 2.4 GHz, DRAM 225 MB, HDD 40 GB, Processor 32bit

Chương 6. Kết luận và hướng phát triển

6.1. Kết luận

Theo yêu cầu đặt ra ban đầu là "Tìm hiểu và phát triển cơ chế bảo mật trên mã nguồn mở của MySQL", cho đến thời điểm này, luận văn đã đạt được các nội dung sau :

6.1.1. Cơ chế bảo mật trên HQT CSDL MySQL

6.1.1.1. Cơ chế bảo mật trên HQT CSDL MySQL

- Bảo mật trên môi trường mạng
- Bảo mật cơ sở dữ liệu
  - ✓ Bảo mật password
  - ✓ Cơ chế cấp quyền trong MySQL

6.1.1.2. Phát triển cơ chế bảo mật

Kết hợp các hiểu biết về cơ chế bảo mật của hệ quản trị MySQL và kiến thức tìm hiểu được về các thuật toàn băm, chúng tôi đã thực hiện xây dựng một cơ chế bảo mật mới cho HQTCSDL MySQL.

Cơ chế mới này được xây dựng theo hướng tiếp cận là thay đổi thuật toán mã hóa password. Việc thay đổi này giúp cho hqtcsdl MySQL bảo mật hơn về password và có thể tương thích tốt với dòng máy 64 bits trong tương lai.

Chúng tôi đã sử dụng thuật toán hàm số băm Tiger và Whirlpool thay thế cho SHA-1(đã bị tấn công) mã hóa password.

Việc thay đổi này được thực hiện trên cả hai môi trường hệ điều hành :

- Window
- Linux

6.1.2. Chương trình HashFunction

Dựa trên kiến thức tìm hiểu được về các thuật toán mã hóa đặc biệt là thuật toán băm (SHA-1, Tiger, Whirlpool) chúng tôi đã xây dựng chương trình "HashFunction" để "demo" các thuật toán SHA-1, Tiger, Whirlpool.

Việc thử nghiệm chương trình được thực hiện trên nhiều cấu hình máy khác nhau.

6.2. Hướng phát triển

6.2.1. Cơ chế bảo mật trong HQTCSDL MySQL

Cơ chế bảo mật của HQTCSDL có nhiều khía cạnh. Do đó khi xây dựng một cơ chế bảo mật mới, ta có thể tiếp cận theo nhiều hướng khác nhau và có thể kết hợp các hướng này lại.

Các hướng phát triển:

- Thay đổi cách phân quyền.
- Thay đổi các lưu trữ dữ liệu (mã hóa dữ liệu).
- Thay đổi thuật toán mã hóa password.

Ngoài ra ta còn có thể đề xuất thêm một số cơ chế bảo mật mới.

Trong phạm vi luận văn, do thời gian và điều kiện tài liệu còn hạn chế nên chỉ mới thay đổi thuật toán mã hóa password. Trong thời gian tới nếu có điều kiện có thể phát triển thêm các hướng khác như mã hóa dữ liệu trước khi lưu xuống file.

6.2.2. Chương trình ứng dụng

Ngày nay, xu hướng thương mại điện tử, chính phủ điện tử ngày càng trở nên thông dụng. Khi đó, dữ liệu không chỉ được truy cập trong mạng nội bộ mà có

thể truy cập trên mạng internet. Do đó vấn đề bảo mật càng phải được quan tâm. Như đã nói, các hqtcsdl hiện nay đang được sử dụng là của nước ngoài, cơ chế bảo mật của các hqtcsdl này thực chất vẫn chưa thể nói "bảo mật" hoàn toàn. Do đó việc tạo ra một cơ chế bảo mật riêng mà chỉ có doanh nghiệp sử dụng mới thật sự biết rõ là điều "cần thiết". Mục tiêu của chương trình "tạo ra một cơ chế bảo mật mới cho hqtcsdl" là bước cơ bản tạo ra một hqtcsdl "an toàn, bảo mật". Sử dụng hqtcsdl này, phát triển thành phần mềm hỗ trợ việc thương mại điện tử của doanh nghiệp, chính phủ điện tử …

Đối với các chương trình đã và đang sử dụng thuật toán SHA-1, chúng ta có thể thay thế bằng các thuật toán đã đề xuất để tăng tính bảo mật cho chương trình.

Tài liệu tham khảo

[1] http://www.mysql.com

[2] http://www.dev.mysql.com

[3] MySQL AB, MySQL *Reference Manual*, 10 February 2005 http://www.theage.com.au/articles/2005/02/21/1108834709994.html

[4] Sam *Varghese*, *Chinese team said to have cracked SHA-1*, February 21, 2005,

[5] Bùi Doãn Khanh, Nguyễn Đình Thúc, *Mã hoá thông tin- lý thuyết và ứng dụng,* NXB lao động xã hội, nhà sách Minh Khai, tháng 12/2004

[6] Trương Thị Mỹ Trang, Trần Hồng Ngọc, *Nghiên cứu các phương pháp mã hoá -giấu tin đa tầng và ứng dụng,* Luận văn cử nhân tin học, năm 2004

[7] Trần Minh Triết, *Nghiên cứu một số vấn đề về bảo mật thông tin và ứng dụng*, Luận văn thạc sĩ tin học, năm 2004

[8] Văn Đức Phương Hồng, Nguyễn Minh Huy*, Nghiên cứu và xây dựng ứng dụng bảo mật trên PDA,* Luận văn cử nhân tin học, năm 2004

[9] Blain Hoffman and Evan Lewis, *Tiger Hash Summary Paper*, CS 402 – Network Security,2004

[10]P. S. L. M. Barreto and V. Rijmen, *The Whirlpool Hashing Function*, First open NESSIE Workshop, Leuven, Belgium, May 24, 2004, http://planeta.terra.com.br/informatica/paulobarreto/whirlpool.zip

[11] Mitsuhiro HATTORI, Shoichi HIROSE, and Susumu YOSHIDA, *Complexity of the* Collision *and Near-Collision Attack on SHA-0 with Different Message Schedules,* Graduate School of Informatics, Kyoto University, Kyoto, JAPAN, Feb 2005

[12] R. Anderson and E. Biham, *Tiger : A Fast New Function*, fast Software Encrytion – FSE'96, LNCS 1039, Springer-Verlag(1996), http://www.esat.kukeuven.ac.be/~rijmen/rijndaeldocV2.zip

[13] Ross Anderson, Eli Biham,*"Tiger: A Fast New Hash Function,"* Proceedings of Fast Software Encryption 3, Cambridge, 1996. http://www.cs.technion.ac.il/~biham/Reports/Tiger/tiger/tiger.html

[14] Ross Anderson, Eli Biham,*Generation of the S boxes of Tiger,*Cambridge University,1996

[15] Bart Preneel, K.U.Leuven, *"Hash functions"*, Version 2.b, January 18, 2004

[16] Taizo Shirai, Kyoji Shibutani, *"On the di_usion matrix employed in the Whirlpool hashing function",2004*

[17] FIPS, *Announcing* the *Secure Hash Standard,2004.*

[18] Vincent Rijmen and Elisabeth Oswald, *Update on SHA-1,* IAIK, Graz University of Technology, Febuary 2005

[19] Jesper Kampfeldt, Collision *Search in Hash Functions,* September 20, 2004

[20] John Kelsey and Bruce Schneier, *Second Preimages on n-bit Hash Functions for n* Much *Less than 2 Work,* National Institute of Standards and Technology,2004

[21] Joux, *"Multicollisions in Iterated Hash Functions. Applications to Cascaded* Constructions*,"* Advances in Cryptology Crypto 2004 Proceedings, Springer-Verlag, 2004.

[22] Krystian Matusiewicz and Josef Pieprzyk, *Finding good differential patterns for attacks on SHA-1,* Centre for Advanced Computing - Algorithms and Cryptography, Department of Computing, Macquarie University, Sydney, NSW 2109, AUSTRALIA, 2005

[23] E. Biham and R. Chen, *Near collisions of SHA-0. In M. Franklin, editor, Advances in* Cryptology, CRYPTO'04, volume 3152 of Lecture Notes in Computer Science, Springer-Verlag, 2004.

[24] National Institute of Standards and Technology, *Secure hash standard (SHS),* FIPS 180-2, August 2002.

[25] Jesper Kampfeldt, *Collision Search in Hash Functions,* September 20, 2004

[26] J. Black, P. Rogaway, and T. Shrimpton, *Black-box analysis of the blockcipher-based hash-function constructions from PGV*, Advances in Cryptology –Crypto'2002, Lecture Notes in Computer Science, vol. 2442, Springer-Verlag, 2002,

[27] Lars Knudsen and Havard Raddum, *A first report on Whirlpool, NUSH, SC2000, Noekeon,Two-Track-MAC and RC6,* NES/DOC/UIB/WP3/007/by, 7th March 2001.

[28] Justin Hibbits, Passwords and Authentication, April 26, 2004.

[29] Thế giới vi tính(trang 24 - 47), tháng 4 năm 2005.

Phụ lục

Phụ lục A Thuật toán SHA

A.1. Hằng số sử dụng trong SHA

A.1.1 Hằng số của SHA-1

SHA-1 sử dụng dãy 80 từ 32 bit là hằng số K_0, K_1, ..., K_{79}

$$K_t = \begin{cases} 5a82799 & 0 \le t \le 19 \\ 6ed9eba1 & 20 \le t \le 39 \\ 8f1bbcdc & 40 \le t \le 59 \\ ca62c1d6 & 60 \le t \le 79 \end{cases}$$

A.1.2 Hằng số của SHA-224 và SHA-256

SHA-224 và SHA-256 sử dụng dãy 64 từ 32 bit là hằng số $K_0^{\{256\}}$, $K_1^{\{256\}}$, …, $K_{63}^{\{256\}}$. Những từ này biểu diễn 32 bit đầu tiên của phần phân số của căn bậc ba của 64 số nguyên tố đầu tiên. Các hằng số bao gồm (theo thứ tự từ trái sang phải)

```
428a2f98 71374491 b5c0fbcf e9b5dba5
3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3
72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc
2de92c6f 4a7484aa 5cb0a9dc 76f988da
27b70a85 2e1b2138 4d2c6dfc 53380d13
650a7354 766a0abb 81c2c62e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3
d192e819 d6990624 f4083585 106aa070
```

```
19a4c116 18376c08 2748774c 34b0bcb5
391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208
90befffa a4506ceb bef9a3f7 c67178f2
```

A.1.3 Hằng số của SHA-384 và SHA-512

SHA-384 và SHA-512 sử dụng cùng dãy 80 từ 64 bit là hằng số $K_0^{\{512\}}$, $K_1^{\{512\}}$, …, $K_{79}^{\{512\}}$. Những từ này biểu diễn 64 bit đầu tiên của phần phân số của căn bậc ba của 80 số nguyên tố đầu tiên. Các hằng số bao gồm (theo thứ tự từ trái sang phải)

```
428a2f98d728ae22  7137449123ef65cd
b5c0fbcfec4d3b2f  e9b5dba58189dbbc
3956c25bf348b538  59f111f1b605d019
923f82a4af194f9b  ab1c5ed5da6d8118
d807aa98a3030242  12835b0145706fbe
243185be4ee4b28c  550c7dc3d5ffb4e2
72be5d74f27b896f  80deb1fe3b1696b1
9bdc06a725c71235  c19bf174cf692694
e49b69c19ef14ad2  efbe4786384f25e3
0fc19dc68b8cd5b5  240ca1cc77ac9c65
2de92c6f592b0275  4a7484aa6ea6e483
5cb0a9dcbd41fbd4  76f988da831153b5
983e5152ee66dfab  a831c66d2db43210
b00327c898fb213f  bf597fc7beef0ee4
c6e00bf33da88fc2  d5a79147930aa725
06ca6351e003826f  142929670a0e6e70
27b70a8546d22ffc  2e1b21385c26c926
4d2c6dfc5ac42aed  53380d139d95b3df
650a73548baf63de  766a0abb3c77b2a8
```

```
81c2c92e47edaee6  92722c851482353b
a2bfe8a14cf10364  a81a664bbc423001
c24b8b70d0f89791  c76c51a30654be30
d192e819d6ef5218  d69906245565a910
f40e35855771202a  106aa07032bbd1b8
19a4c116b8d2d0c8  1e376c085141ab53
2748774cdf8eeb99  34b0bcb5e19b48a8
391c0cb3c5c95a63  4ed8aa4ae3418acb
5b9cca4f7763e373  682e6ff3d6b2b8a3
748f82ee5defb2fc  78a5636f43172f60
84c87814a1f0ab72  8cc702081a6439ec
90befffa23631e28  a4506cebde82bde9
bef9a3f7b2c67915  c67178f2e372532b
ca273eceea26619c  d186b8c721c0c207
eada7dd6cde0eb1e  f57d4f7fee6ed178
06f067aa72176fba  0a637dc5a2c898a6
113f9804bef90dae  1b710b35131c471b
28db77f523047d84  32caab7b40c72493
3c9ebe0a15c9bebc  431d67c49c100d4c
4cc5d4becb3e42b6  597f299cfc657e2a
5fcb6fab3ad6faec  6c44198c4a475817
```

A.2 Giá trị khởi tạo trong SHA

SHA-1

$H_0^{(0)} = 67452301$
$H_1^{(0)} = efcdab89$
$H_2^{(0)} = 98badcfe$
$H_3^{(0)} = 10325476$
$H_4^{(0)} = c3d2e1f0$

SHA-224

$H_0^{(0)} = c1059ed8$
$H_1^{(0)} = 367cd507$
$H_2^{(0)} = 3070dd17$
$H_3^{(0)} = f70e5939$
$H_4^{(0)} = ffc00b31$
$H_5^{(0)} = 68581511$
$H_6^{(0)} = 64f98fa7$
$H_7^{(0)} = befa4fa4$

SHA-256

$H_0^{(0)} = 6a09e667$
$H_1^{(0)} = bb67ae85$
$H_2^{(0)} = 3c6ef372$
$H_3^{(0)} = a54ff53a$
$H_4^{(0)} = 510e527f$
$H_5^{(0)} = 9b05688c$
$H_6^{(0)} = 1f83d9ab$
$H_7^{(0)} = 5be0cd19$

SHA-384

$H_0^{(0)} = cbbb9d5dc1\ 059ed8$
$H_1^{(0)} = 629a292a36\ 7cd507$
$H_2^{(0)} = 9159015a30\ 70dd17$
$H_3^{(0)} = 152fecd8f7\ 0e5939$
$H_4^{(0)} = 67332667ff\ c00b31$
$H_5^{(0)} = 8eb44a8768\ 581511$
$H_6^{(0)} = db0c2e0d64\ f98fa7$
$H_7^{(0)} = 47b5481dbe\ fa4fa4$

SHA-512

$H_0^{(0)} = 6a09e667f3\ bcc908$
$H_1^{(0)} = bb67ae8584\ faa73b$
$H_2^{(0)} = 3c6ef372fe\ 94f82b$
$H_3^{(0)} = a54ff53a5f\ 1d36f1$
$H_4^{(0)} = 510e527fad\ e682d1$
$H_5^{(0)} = 9b05688c2b\ 3e6c1f$
$H_6^{(0)} = 1f83d9abfb\ 41bd6b$
$H_7^{(0)} = 5be0cd1913\ 7e2179$

A.3 Các thao tác tiền xử lý trong SHA

- **SHA-1:**

$$f_j(x,y,z) = \begin{cases} \text{ch}(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) & 0 \le j \le 19 \\ \text{parity}(x,y,z) = x \oplus y \oplus z & 20 \le j \le 39 \\ \text{maj}(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) & 40 \le j \le 59 \\ \text{parity}(x,y,z) = x \oplus y \oplus z & 60 \le j \le 79 \end{cases}$$

- **SHA-224 và SHA-256:**

$$\text{ch}(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$\text{maj}(z, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_0^{256}(x) = \text{ROTR}^2(x) \oplus \text{ROTR}^{13}(x) \oplus \text{ROTR}^{22}(x)$$
$$\sum_1^{256}(x) = \text{ROTR}^6(x) \oplus \text{ROTR}^{11}(x) \oplus \text{ROTR}^{25}(x)$$
$$\sigma_0^{256}(z) = \text{ROTR}^7(x) \oplus \text{ROTR}^{18}(x) \oplus \text{SHR}^3(x)$$
$$\sigma_1^{256}(z) = \text{ROTR}^{17}(x) \oplus \text{ROTR}^{19}(x) \oplus \text{SHR}^{10}(x)$$

- **SHA-384 và SHA-521:**

$$\text{ch}(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$\text{maj}(z, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_0^{512}(x) = \text{ROTR}^{28}(x) \oplus \text{ROTR}^{34}(x) \oplus \text{ROTR}^{29}(x)$$
$$\sum_1^{512}(x) = \text{ROTR}^{14}(x) \oplus \text{ROTR}^{18}(x) \oplus \text{ROTR}^{41}(x)$$
$$\sigma_0^{512}(z) = \text{ROTR}^1(x) \oplus \text{ROTR}^8(x) \oplus \text{SHR}^7(x)$$
$$\sigma_1^{512}(z) = \text{ROTR}^{19}(x) \oplus \text{ROTR}^{61}(x) \oplus \text{SHR}^6(x)$$

A.4 Thuật toán tính hàm BĂM trong SHA

A.4.1 SHA-1

SHA-1 cũng sử dụng một từ đơn tạm T.

Với i = 1 đến N:

{

Chuẩn bị bảng phân bố thông điệp {Wt}:

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ ROTL^1(W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16} & 16 \le t \le 79 \end{cases}$$

Khởi tạo 5 biến ban đầu a, b, c, d và e, với giá trị băm thứ i-1:

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

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

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

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

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

Với t = 0 đến 79:

{

T = ROTL5(a) + ft(b,c,d) + e + Kt + Wt

e = d

d = c

c = ROTL30(b)

b = a

a = T

}

Tính giá trị băm H(i) của vòng lặp thứ i

$H_0^{(i)}$ = 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 + $H_4^{(i-1)}$

Sau khi lặp 4 bước trên N lần (sau khi xử lý M(N)), thông điệp rút gọn 160-bit của thông điệp M là :

$$H_0^{(N)} || H_1^{(N)} || H_2^{(N)} || H_3^{(N)} || H_4^{(N)}$$

Ký hiệu || chỉ phép nối chuỗi bit theo thứ tự đã định.

A.4.2 SHA-224

SHA-224 cũng sử dụng hai từ đơn tạm T1, T2.

Với i = 0 đến N

{

Chuẩn bị bảng phân bố thông điệp {Wt}

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases}$$

Khởi tạo 8 biến a, b, c, d, e, f, g, h với giá trị băm thứ i-1

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

Với t = 0 đến 63:

{

T1 = h + $\sum_1^{\{256\}}(e) + Ch(e, f, g) + K_t^{\{256\}} + W_t$

T2 = $\sum_0^{\{256\}}(a) + Maj(a, b, c)$

h = g

g = f

f = e

e = d + T1

d = c

c = b

b = a

a = T1 + T2

}

Tính giá trị băm H(i) của vòng lặp thứ i

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

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

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

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

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

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

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

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

}

Sau khi lặp 4 bước trên N lần (sau khi xử lý M(N)), thông điệp rút gọn 224-bit của thông điệp M là :

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)}$$

A.4.3 SHA-256

SHA-256 cũng sử dụng hai từ đơn tạm T1, T2.

Với i = 0 đến N

{

Chuẩn bị bảng phân bố thông điệp {Wt}

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 63 \end{cases}$$

Khởi tạo 8 biến a, b, c, d, e, f, g, h với giá trị băm thứ i-1

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

Với t = 0 đến 63:

{

T1 = h + $\sum_1^{\{256\}}$(e) + $Ch(e, f, g)$ + $K_t^{\{256\}}$ + W_t

T2 = $\sum_0^{\{256\}}(a)$ + $Maj(a, b, c)$

h = g

g = f

f = e

e = d + T1

d = c

c = b

b = a

a = T1 + T2

}

Tính giá trị băm H(i) của vòng lặp thứ i

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

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

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

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

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

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

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

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

}

Sau khi lặp 4 bước trên N lần (sau khi xử lý M(N)), thông điệp rút gọn 256-bit của thông điệp M là :

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)}$$

A.4.4 SHA-384

SHA-384 cũng sử dụng hai từ đơn tạm T1, T2.

Với i = 1 đến N

{

Chuẩn bị bảng phân bố thông điệp {Wt}

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{512\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{512\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 79 \end{cases}$$

Khởi tạo 8 biến a, b, c, d, e, f, g, h với giá trị băm thứ i-1

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

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

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

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

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

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

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

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

Với t = 0 đến 63:

{

T1 = h + $\sum_1^{\{512\}}$(e) + $Ch(e, f, g) + K_t^{\{512\}} + W_t$

T2 = $\sum_0^{\{512\}}(a) + Maj(a, b, c)$

h = g

g = f

f = e

e = d + T1

d = c

c = b

b = a

a = T1 + T2

}

Tính giá trị băm H(i) của vòng lặp thứ i

$H_0^{(i)}$ = 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 + $H_7^{(i-1)}$

}

Sau khi lặp 4 bước trên N lần (sau khi xử lý M(N)), thông điệp rút gọn 384-bit của thông điệp M là :

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)}$$

A.4.5 SHA-512

SHA-512 cũng sử dụng hai từ đơn tạm T1, T2.

Với i = 1 đến N

{

Chuẩn bị bảng phân bố thông điệp {Wt}

$$W_t = \begin{cases} M_t^{(i)} & 0 \le t \le 15 \\ \sigma_1^{\{512\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{512\}}(W_{t-15}) + W_{t-16} & 16 \le t \le 79 \end{cases}$$

Khởi tạo 8 biến a, b, c, d, e, f, g, h với giá trị băm thứ i-1

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

Với t = 0 đến 63:

{

$T1 = h + \sum_1^{\{512\}}(e) + Ch(e, f, g) + K_t^{\{512\}} + W_t$

$T2 = \sum_0^{\{512\}}(a) + Maj(a, b, c)$

h = g

g = f

f = e

e = d + T1

d = c

c = b

b = a

a = T1 + T2

}

Tính giá trị băm H(i) của vòng lặp thứ i

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

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

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

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

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

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

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

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

}

Sau khi lặp 4 bước trên N lần (sau khi xử lý M(N)), thông điệp rút gọn 512-bit của thông điệp M là

$$H_0^{(N)} \,||\, H_1^{(N)} \,||\, H_2^{(N)} \,||\, H_3^{(N)} \,||\, H_4^{(N)} \,||\, H_5^{(N)} \,||\, H_6^{(N)} \,||\, H_7^{(N)}$$

Phụ lục B Thuật toán Tiger

Việc tính toán này bao gồm 3 lần thông qua hàm pass, và cứ giữa hai hàm pass là hàm key schedule, hàm này có tác dụng biến đổi dạng dữ liệu đầu vào ngăn cản kẻ tấn công lợi dụng tính thưa của dữ liệu trong 3 vòng lặp qua hàm pass trên. Cuối cùng hàm feedforward được dùng, ở đó giá trị của a, b, c được kết hợp với các giá trị khởi đầu của nó để cho giá trị băm h_{i+1}:

save_abc

pass(a, b, c, 5)

key_schedule

pass(c, a, b, 7)

key_schedule

pass(b, c, a, 9)

feedforward

Trong đó:

- **save_abc lưu lại giá trị hi**

  aa = a

  bb = b

  cc = c

  dd = d

- **pass(a, b, c, mul) là:**

  round(a, b, c, x0, mul);

  round(b, c, a, x1, mul);

  round(c, a, b, x2, mul);

round(a, b, c, x3, mul);

round(b, c, a, x4, mul);

round(c, a, b, x5, mul);

round(a, b, c, x6, mul);

round(b, c, a, x7, mul);

trong đó:

c ^= x;

a -= t1[c0] ^ t2[c2] ^ t3[c4] ^ t4[c6] ;

b += t4[c1] ^ t3[c3] ^ t2[c5] ^ t1[c7] ;

b *= mul;

Ở đây c_i là byte thứ i của c ($0 \leq i \leq 7$) và t1 đến t4 là 4 bảng S_box được trình bày trong tập tin "sboxes.h" đính kèm.

- **key_schedule**

x0 -= x7 ^ 0xA5A5A5A5A5A5A5A5;

x1 ^= x0;

x2 += x1;

x3 -= x2 ^ ((~x1)<<19);

x4 ^= x3;

x5 += x4;

x6 -= x5 ^ ((~x4)>>23);

x7 ^= x6;

x0 += x7;

x1 -= x0 ^ ((~x7)<<19);

x2 ^= x1;

x3 += x2;

x4 -= x3 ^ ((~x2)>>23);

x5 ^= x4;

x6 += x5;

x7 -=x6 ^ 0x0123456789ABCDEF;

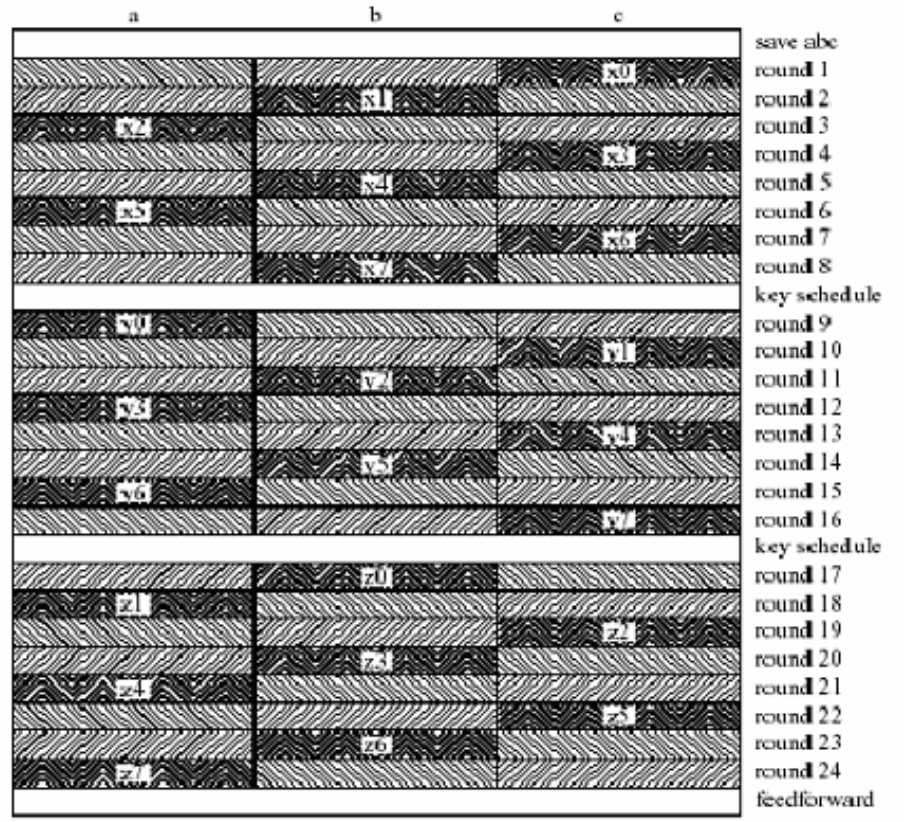
- **feedforward**

a ^= aa;

b – = bb;

c += cc;

Các thanh ghi a, b, c chứa các giá trị băm (trung gian) hi + 1 dài 192 bit



Hình phác thảo chức năng nén của Tiger

Vùng màu đen là các thanh ghi bị ảnh hưởng mà các đường sọc trong các thanh ghi này chỉ đến các byte trong vùng màu trắng. Các biến y0,…,y7 và

các biến z0,…z7 là ký hiệu của x0,…,x7 tương ứng trong hàm pass thứ 2 và thứ 3. Cuối cùng, giá trị trung gian hn được xem như đầu ra Tiger/192.

Phụ lục C Tấn công SHA-1

Sau dây là tài liệu giới thiệu chung về cách tấn công SHA-1 do nhóm nghiên cứu công bố[5].

1. Giới thiệu

Trong phần này,chúng tôi tóm tắt lại kết quả của nghiên cứu cách thức tấn công xung đột mới vào SHA-1. Chi tiết của công nghệ sẽ được cung cấp trong thời gian tới.

Chúng tôi phát triển tập các công nghệ có hiệu quả đối với việc nghiên cứu các xung đột trong SHA-1. Sự phân tích của chúng tôi chỉ ra rằng các xung đột của SHA-1 có thể được thấy với độ phức tạp ít 2^{69} các thao tác băm khác. Đây là lần tấn công đầu tiên gồm đầy đủ 80 bước SHA-1 với độ phức tạp ít hơn 2^{80} lần lý thuyết. Dựa trên sự ước lượng, chúng tôi mong các xung đột thật sự giảm xuống 70 bước có thể được sử dụng trong ngày gần đây.

Cách đây vài năm, có một vài thuận lợi quan trọng trong việc phân tích các hàm băm. Công nghệ phát triển trong thời gian gần đây cung cấp nền móng quan trọng cho các lần tấn công mới vào SHA-1. Thực chất, các phân tích của chúng tôi được xây dựng dựa vào nguồn gốc tấn công khác nhau vào SHA-0, tấn công xung đột gần đây vào SHA-0, các công nghệ xung đột đa block (multiblock collision techniques), tương tự như kỹ thuật biến đổi thông điệp sử dụng trong

[5] shanote "Collision Search Attacks on SHA-1", February 13, 2005

nghiên cứu xung đột tấn công vào MD5. Phá vỡ SHA-1 không thể không có các công nghệ phân tích mạnh.

Phương thức tấn công áp dụng vào SHA-0 và biến đổi giảm toàn bộ cho SHA-1. Đối với SHA-0, việc tấn công hoàn toàn hiệu quả đến nỗi chúng tôi đã có thể tìm thấy những xung đột thật sự của toàn bộ SHA-0 với ít hơn 2^{39} thao tác băm. Chúng tôi cũng có thể bổ sung vào việc tấn công vào SHA-1 để giảm xuống 58 bước về tìm xung đột thật sự với ít hơn 2^{33} thao tác băm.

Sau đây là 2 ví dụ xung đột được đưa ra trong tài liệu.

2. Ví dụ collision của các bước SHA-0

$h_1 = compress(h_0,M_0)$

$h_2 = compress(h_1,M_1) = compress(h_1,M_1')$

h_0:	67452301	efcdab89	98badcfe	10325476	c3d2e1f0
M_0:	65c24f5c	0c0f89f6	d478de77	ef255245	
	83ae3a1f	2a96e508	2c52666a	0d6fad5a	
	9d9f90d9	eb82281e	218239eb	34e1fbc7	
	5c84d024	f7ad1c2f	d41d1a14	3b75dc18	
h_1:	39f3bd80	c38bf492	fed57468	ed70c750	c521033b
M_1:	474204bb	3b30a3ff	f17e9b08	3ffa0874	
	6b26377a	18abdc01	d320eb93	b341ebe9	
	13480f5c	ca5d3aa6	b9f3bd88	21921a2d	
	4085fca1	eb65e659	51ac570c	54e8aae5	

M_1' :	c74204f9	3b30a3ff	717e9b4a	3ffa0834	
	6b26373a	18abdc43	5320eb91	3341ebeb	
	13480f1c	4a5d3aa6	39f3bdc8	a1921a2f	
	4085fca3	6b65e619	d1ac570c	d4e8aaa5	
h_2:	2af8aee6	ed1e8411	62c2f3f7	3761d197	0437669d

Bảng 1: Collision của toàn bộ 80-bước SHA-0. Hai thông điệp đụng độ là: (M_0, M_1) và (M_0, M_1'). Lưu ý rằng quy luật để tìm ra đụng độ không được công bố

3. Ví dụ collision của 58 bước SHA-1

$$h_1 = compress(h_0,M_0) = compress(h_0,M_0')$$

h_0:	67452301	efcdab89	98badcfe	0325476	c3d2e1f0
M_0:	132b5ab6	a115775f	5bfddd6b	4dc470eb	
	0637938a	6cceb733	0c86a386	68080139	
	534047a4	a42fc29a	06085121	a3131f73	
	ad5da5cf	13375402	40bdc7c2	d5a839e2	
M_0' :	332b5ab6	c115776d	3bfddd28	6dc470ab	
	e63793c8	0cceb731	8c86a387	68080119	
	534047a7	e42fc2c8	46085161	43131f21	
	0d5da5cf	93375442	60bdc7c3	f5a83982	
h1:	9768e739	b662af82	a0137d3e	918747cf	c8ceb7d4

Bảng 2: Collision của SHA-1 được giảm xuống 58 bước. Hai thông điệp đụng độ là: M_0 và M_0'. Lưu ý rằng quy luật để tìm ra đụng độ không được công bố.