CHƯƠNG IV
CÁC ỨNG DỤNG AN NINH TRONG HỆ THỐNG THÔNG TIN

Các cơ chế mật mã và xác thực thông tin là cơ sở cho việc xây dựng các ứng dụng an ninh trong hệ thống thông tin, đặc biệt là trong môi trường mạng. Chương này sẽ trình bày một số ứng dụng của các kỹ thuật mật mã và xác thực thông tin vào việc xây dựng các *giao thức* (Protocol) và *dịch vụ* (Service) bảo mật trên mạng nhằm đảm bảo an toàn hệ thống thông tin. Các nội dung chính được trình bày trong chương này bao gồm:

- Giao thức xác thực (Authentication Protocol)
- Bộ giao thức IPSec (Internet Protocol Security)
- Bộ giao thức SSL (Secure Sockets Layer)
- Giao thức SET (Secure Electronic Transaction)

1. GIAO THỨC XÁC THỰC (Authentication Protocol)

Xác thực và cho phép

Xác thực (Authentication) trong bảo mật an toàn hệ thống là một quá trình/ cơ chế nhận dạng (Identifying) một cá nhân, thường dựa trên tên người dùng (Username) và mật khẩu (Password).

Cho phép (Authorization), trong các hệ thống bảo mật an toàn, là quá trình/ cơ chế cho phép hay không các cá nhân được truy cập vào các đối tượng của hệ thống dựa trên nhân dạng (Identity) đã được xác thực của chúng.

Việc xác thực thường đề cập đến việc xác nhận nhân dạng của một thực thể nhằm đảm bảo an toàn cho việc giao tiếp trao đổi thông tin với thực thể đó.

Thông thường, một thực thể bên ngoài hệ thống muốn truy cập vào hệ thống thì phải cung cấp các thông tin về nhân dạng của mình cho một hệ thống xác thực **AS** (Authentication System) để hệ thống này xác thực nhân dạng của thực thể đó.

Hệ điều hành Windows Server 2003 cung cấp một số giao thức xác thực:

- Kerberos authentication protocol
- NT LAN Manager (NTLM) authentication protocol
- Secure Sockets Layer/Transport Security Layer (SSL/TLS)
- Digest authentication
- Smart cards
- Virtual Private Networking (VPN) and Remote Access Services (RAS)

Cơ chế xác thực thông dụng và đơn giản nhất

Trong số các cơ chế xác thực thì cơ chế xác thực dựa trên thông tin mà thực thể biết là cơ chế xác thực đơn giản nhất và được sử dụng nhiều nhất.

Thông tin đó thường là tên người dùng (UserName) và mật khẩu (Password).

1.1. Mật khẩu (Password)

Mật khẩu

Theo nghĩa thông thường thì mật khẩu là một từ hay một chuỗi ký tự bí mật mà chỉ một nhóm người biết được, thường được dùng để giữ sự bí mật đối với những người khác.

Trong lĩnh vực tính toán bằng máy tính, mật khẩu là một từ (Word) hoặc một chuỗi ký tự (String of characters) mà chỉ một hay một nhóm người dùng biết được, nó được dùng cùng với tên người dùng để đăng nhập vào máy tính hay mạng máy tính.

Trong các cơ chế xác thực bằng mật khẩu hiện nay, mật khẩu thường được hiểu như là bí mật chia sẻ (Shared Secret)

Không gian mật khẩu

Thông thường mỗi mật khẩu thuộc về một *không gian mật khẩu* (Password Space), là tập hợp tất cả các chuỗi ký tự có thể được dùng làm mật khẩu. Mỗi hệ thống xác thực có một không gian mật khẩu khác nhau. Không gian mật khẩu càng lớn thì khả năng hệ thống bị tấn công mật khẩu bằng *phương thức vét cạn* (Brute Force) càng thấp.

Mật khẩu phức tạp

Một mật khẩu được gọi là phức tạp nếu nó khó bị phát hiện bằng *phương pháp tấn công tự điển* (Dictionary Attack).

Dạng mật khẩu phổ biến

Theo khảo sát thì những dạng mật khẩu được dùng phổ biến nhất hiện nay là:

- Tên người dùng (User Name) hoặc thêm một vài chữ số (ví dụ ngày sinh, số điện thoại, …)
- Tên đăng nhập (Logon Name)
- Tên máy tính (Computer Name)
- Số điện thoại, ngày sinh …. .
- Từ khóa đặc biệt như *computer, hacker*, …
- Từ có nghĩa trong tự điển
- Tên một người có quan hệ mật thiết với người dùng

.....

Chính sách về mật khẩu đối với người dùng

Những mật khẩu như trên đều có độ phức tạp rất thấp và do đó dễ dàng bị phát hiện. Các hệ thống xác thực thường đưa ra các chính sách về mật khẩu (Password Policy) đối với người dùng. Các chính sách về mật khẩu thường có những quy định sau đây đối với mật khẩu:

- Về độ phức tạp của mật khẩu (Password Complexity): quy định chiều dài tối thiểu, độ khó, mật khẩu không được chứa User Name hoặc Logon Name.
- Về tuổi của mật khẩu (Password Age): quy định thời gian sử dụng tối đa của mật khẩu.
- Về lịch sử của mật khẩu (Password History): quy định không được phép

dùng lại mật khẩu cũ.

Nguyên tắc chung để đặt mật khẩu

Về phía người dùng, những nguyên tắc chung cần được thực hiện để tăng độ an toàn cho việc xác thực bằng mật khẩu bao gồm:

- Sử dụng nhiều loại ký tự khác nhau để làm mật khẩu.
- Không sử dụng các mật khẩu quá ngắn.
- Không sử dụng những từ khóa hoặc từ có nghĩa trong mật khẩu.
- Thường xuyên thay đổi mật khẩu.
- Không ghi chép mật khẩu lên bất kỳ vị trí nào.
- Không tiết lộ mật khẩu cho người khác, ngay cả trong những tình huống an toàn nhất.

Ghi chú về xác thực bằng mật khẩu (ghi nhớ)

1- Mặt khác, để đảm bảo an toàn hơn nữa, trên các *máy chủ xác thực* AS (Authentication Servers), *không được phép lưu trữ mật khẩu của người dùng trực tiếp dưới dạng thông tin gốc* (Cleartext) mà phải được mã hoá dưới một dạng nào đó.

2- Ngoài ra trong viễn thông, để mật khẩu không bị đánh cắp trong khi truyền đi trên mạng, đã có nhiều thủ tục xác thực tinh vi được thiết kế để đảm bảo *không truyền đi trên mạng mật khẩu ở dạng thông tin gốc*.

1.2. Xác thực bằng mật khẩu trong mô hình điểm-điểm (Point to Point Model)

Mô hình điểm - điểm

Mô hình điểm- điểm là mô hình dùng để thiết lập một kết nối trực tiếp giữa 2 node trên mạng thay vì phải qua một trục trung tâm (Central Hub).

Trong các hệ thống cổ điển, các kết nối từ xa thường được thực hiện thông qua mô hình điểm- điểm, theo đó các *giao thức kết nối* như: SLIP (Serial Line Internet Protocol) hoặc PPP (Point to Point Protocol) thường được sử dụng. Trong trường hợp này, các thủ tục xác thực đều là một chiều, tức là chỉ có máy chủ xác thực AS xác thực người dùng chứ không có chiều ngược lại.

Hai giao thức xác thực trong mô hình điểm - điểm

Hai giao thức xác thực bằng mật khẩu thường được dùng trong các hệ thống kết nối dạng này là:

1. Giao thức xác thực PAP (Password Authentication Protocol)

Là giao thức xác thực dựa trên mật khẩu đơn giản nhất và do đó kém an toàn nhất. Quá trình xác thực bằng PAP giữa bên khởi xướng (Initiator) và bên xác thực (Authenticator) gồm 2 bước sau đây:

1- Sau pha thiết lập kết nối (LCP[1] Link Establishment), bên khởi xướng gởi tên đăng nhập và mật khẩu gốc của mình cho bên xác thực dưới dạng một *gói tin yêu*

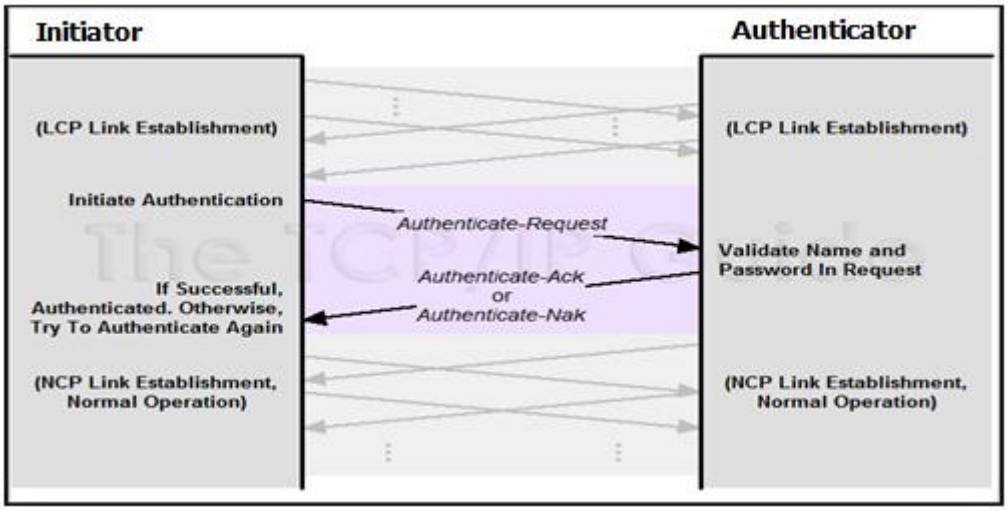
[1] LCP: Link Control Protocol

cầu xác thực (Authenticate Request Packet).

2- Bên xác thực sẽ xác thực thông tin chứa trong gói dữ liệu này:

□ Nếu thông tin yêu cầu xác thực đúng với thông tin xác thực như đã lưu trữ trước đó thì bên xác thực sẽ trả lời bằng một *gói tin báo xác thực đầy đủ* (Authenticate ACK Packet) và quá trình xác thực xem như thành công.

□ Ngược lại, nếu thông tin yêu cầu xác thực không đúng với thông tin xác thực đã được lưu trữ, bên xác thực sẽ trả lời bằng *gói tin báo xác thực không đầy đủ* (Authenticate NAK Packet), xem như quá trình xác thực thất bại, khi đó bên khởi xướng cố gắng xác thực lại.



2. Giao thức xác thực CHAP (Challenge-Handshake Authentication Protocol)

CHAP cũng là giao thức xác thực bằng mật khẩu nhưng phức tạp hơn PAP, được sử dụng bởi các server để xác thực tính hợp lệ về nhân dạng của client ở xa.

CHAP kiểm tra định kỳ nhân dạng của client bằng cách sử dụng giao thức *bắt tay ba chiều* (Three-way Handshake). Điều này xảy ra vào lúc thành lập kết nối lần đầu tiên, và có thể xảy ra thêm nữa vào bất kỳ lúc nào sau này. Việc xác thực dựa trên bí mật chia sẻ (Shared Secret), ví dụ như mật khẩu của client.

CHAP có ưu điểm hơn PAP về phương diện bảo mật *vì có dùng các hàm băm một chiều (One way hash function) nên không gửi thông tin xác thực gốc trực tiếp trên mạng*.

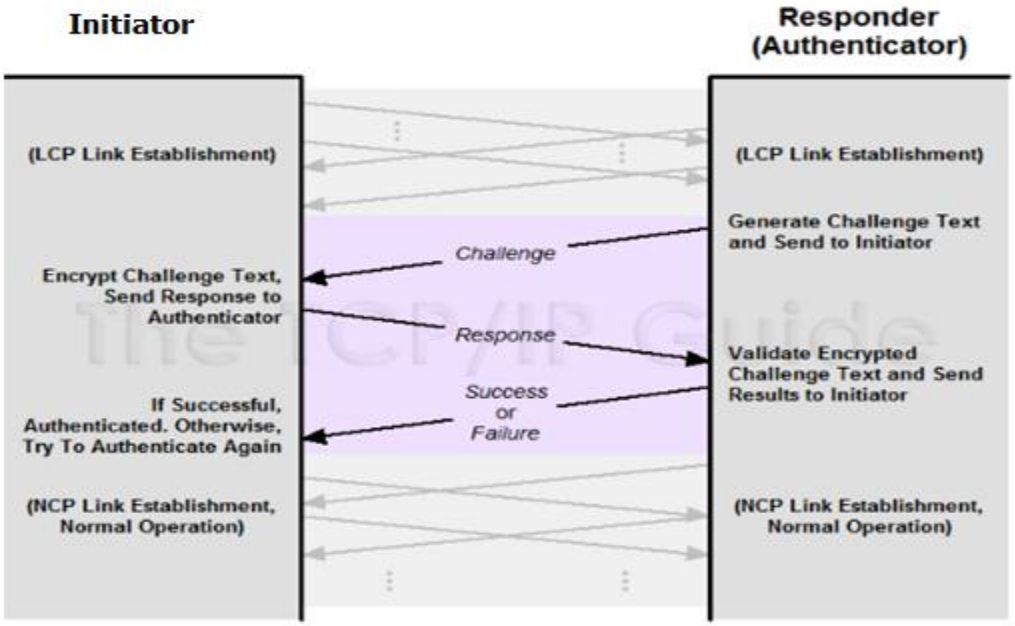
Quá trình xác thực bằng giao thức CHAP giữa người khởi xướng (Initiator/Client) và máy chủ xác thực (Authenticator/Server) gồm các bước sau:

1- Sau khi hoàn thành pha thiết lập kết nối, máy chủ xác thực gửi một thông điệp "Challenge" cho phía đối tác người khởi xướng.

2- Sau khi nhận được Challenge, phía người khởi xướng trả lời bằng một "Response". Response được tính bằng cách sử dụng hàm băm một chiều trên Challenge và một bí mật được kết hợp vào.

3- Khi nhận được Response, phía máy chủ xác thực kiểm tra lại Response dựa vào sự tính toán của mình về giá trị băm đang mong đợi. Nếu nhận được giá trị phù hợp thì máy chủ xác thực hồi đáp xác thực là "Success" (thành công) ; ngược lại là "Failure" (thất bại), khi đó người dùng cố gắng xác thực lại.

Chú ý rằng, theo từng khoảng thời gian máy chủ xác thực sẽ gửi một Challenge mới đến phía đối tác người dùng để xác thực định kỳ.

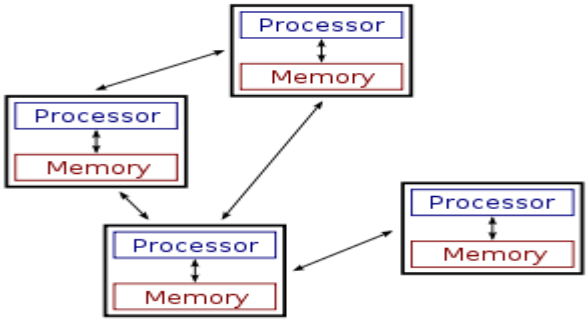


1.3. Xác thực bằng mật khẩu trong hệ thống phân tán (Distributed System)

Hệ thống phân tán

Một hệ thống phân tán được hiểu là một mạng bao gồm các máy tính tự trị (Autonomous Computers). Chúng cho phép chia sẻ các nguồn lực và khả năng của chính mình để cung cấp cho người dùng một mạng đơn nhất và tích hợp chặt chẽ.

Các thành phần nằm trên hệ thống phân tán truyền đạt và phối hợp hành động của chúng bằng cách truyền các thông điệp cho nhau. Các thành phần này tương tác với nhau để đạt được một mục đích chung.



Yêu cầu đối với thủ tục xác thực trong hệ thống phân tán

Trong các hệ thống phân tán, các máy chủ dịch vụ SS (Service Server) được quản lý bởi một máy chủ xác thực AS duy nhất. Các thủ tục xác thực trong các hệ thống phân tán phải đảm bảo được 2 yêu cầu cơ bản là:

- Đảm bảo an toàn đối với thông tin xác thực, tức là tên đăng nhập và mật khẩu gốc không được truyền đi trực tiếp trên mạng.

- Người dùng chỉ cần đăng nhập một lần cho mỗi phiên giao dịch nhưng có

khả năng sử dụng tất cả các dịch vụ có trong hệ thống phân tán trong phiên giao dịch đó.

Thủ tục xác thực cơ bản trong hệ thống phân tán

Khi một người dùng đăng nhập từ một máy con C trong hệ thống và có yêu cầu truy xuất đến máy chủ dịch vụ SS thì một *thủ tục xác thực cơ bản* điển hình trong hệ thống phân tán thực hiện lần lượt các bước như sau:

1- Từ máy con C, người dùng cung cấp cho máy chủ xác thực AS: tên đăng nhập (ID_C), mật khẩu (P_C), địa chỉ mạng (AD_C) của máy con C và nhân dạng ID_{SS} của máy chủ dịch vụ SS mà mình cần.

$$C \xrightarrow{ID_C + P_C + AD_C + ID_{SS}} AS$$

2- Sau khi nhận được thông tin từ C:

- Máy chủ xác thực AS kiểm tra xem tên đăng nhập và mật khẩu có hợp lệ hay không, đồng thời kiểm tra xem người dùng này có được phép truy xuất các dịch vụ trên máy chủ dịch vụ SS hay không.

- Nếu cả hai việc kiểm tra trên đều thành công thì người dùng được phép truy xuất dịch vụ trên máy chủ dịch vụ SS.

Khi đó, máy chủ xác thực AS tạo ra một *thẻ dịch vụ* (Ticket) chứa các thông tin bao gồm: nhân dạng của người dùng (ID_C), địa chỉ mạng của máy con C (AD_C) và nhân dạng của máy chủ dịch vụ SS (ID_{SS}). Thông tin này được mã hóa bằng khóa bí mật $K_{AS,SS}$ dùng chung giữa máy chủ xác thực AS và máy chủ dịch vụ SS.

Máy chủ xác thực AS gửi thẻ dịch vụ Ticket cho máy con C.

$$C \xleftarrow{Ticket = E(ID_C + AD_C + ID_{SS}, K_{AS,SS})} AS$$

3- Sau đó, từ máy con C, người dùng có thể yêu cầu các dịch vụ của máy chủ dịch vụ SS bằng cách gởi nhân dạng của mình (ID_C) có gắn kèm thẻ dịch vụ Ticket vừa nhận được cho máy chủ dịch vụ SS.

$$C \xrightarrow{ID_C + Ticket} SS$$

Máy chủ dịch vụ SS sẽ giải mã thẻ dịch vụ Ticket để kiểm tra ID_C, và qua đó có chấp nhận cho máy con C truy xuất các dịch vụ của mình hay không.

Hai vấn đề cần được giải quyết

Thủ tục xác thực cơ bản trên đã giải quyết được vấn đề bảo mật bằng cách đưa ra khái niệm *thẻ dịch vụ* (Ticket), do máy chủ xác thực AS tạo ra. Trong thẻ dịch vụ đó các thông tin bí mật được mã hóa khi được chuyển đi trên mạng. Tuy nhiên, vẫn còn hai vấn đề chưa được giải quyết là:

1- Nếu người dùng có nhu cầu sử dụng dịch vụ nhiều lần, hoặc sử dụng nhiều

dịch vụ trên các máy chủ dịch vụ khác nhau thì liệu người dùng có cần phải thực hiện thủ tục xác thực nhiều lần không?, tức là có cần phải nhập lại bằng mật khẩu của mình nhiều lần hay không?

2- Thủ tục xác thực cơ bản vẫn còn một bước không ổn, là bước đầu tiên, vì khi đó thông tin dùng để xác thực, mà chủ yếu là mật khẩu, được gởi đi trực tiếp trên mạng mà chưa mã hóa. Điều này là cần tránh.

Thủ tục xác thực cải tiến giải quyết được hai vấn đề trên

Để khắc phục 2 vấn đề đã nêu, một thành phần mới được thêm vào hệ thống xác thực, được gọi là *máy chủ cấp thẻ dịch vụ TGS* (Ticket Granting Server).

Với giải pháp này, khi người dùng xác thực thành công với máy chủ xác thực AS thì thay vì cấp *thẻ dịch vụ* ($Ticket_{SS}$) trực tiếp cho người dùng thì máy chủ xác thực AS chỉ cấp cho người dùng *thẻ xác thực* ($Ticket_{TGS}$). Thẻ xác thực ($Ticket_{TGS}$) này có tác dụng xác nhận đây là một người dùng hợp hệ. Kể từ đó về sau, mỗi khi người dùng cần truy xuất dịch vụ nào trên máy chủ dịch vụ SS thì chỉ cần gởi thẻ xác thực $Ticket_{TGS}$ và yêu cầu của mình đến máy chủ cấp thẻ dịch vụ TGS để được cấp *thẻ dịch vụ* $Ticket_{SS}$.

Như vậy, máy chủ xác thực AS chỉ cần cấp thẻ xác thực $Ticket_{TGS}$ cho người dùng một lần. Nói cách khác thẻ xác thực $Ticket_{TGS}$ đó có thể được dùng lại, cả trong trường hợp người dùng sử dụng một dịch vụ nhiều lần hoặc sử dụng nhiều dịch vụ khác nhau mà không cần phải đăng nhập lại.

Thủ tục xác thực cải tiến bao gồm các bước như sau:

1- Người dùng từ máy con C yêu cầu một *thẻ xác thực* người dùng hợp lệ bằng cách gởi cho máy chủ xác thực AS: nhân dạng của người dùng (ID_C), địa chỉ mạng của máy con C (AD_C) và nhân dạng của máy chủ cấp thẻ dịch vụ TGS (ID_{TGS}).

$$C \xrightarrow{ID_C + AD_C + ID_{TGS}} AS$$

2- Sau khi nhận được yêu cầu và xác thực xong, máy chủ xác thực AS tạo *thẻ xác thực* $Ticket_{TGS}$. Do thẻ xác thực này có khả năng được dùng lại, nên để quản lý việc tồn tại của nó thì, ngoài những thông tin nhận được, trong thẻ này còn được gắn thêm một *nhãn thời gian* TS_1 chỉ rõ thời điểm tạo thẻ và *thời gian tồn tại hợp lệ* $LifeTime_1$ của thẻ.

Để tránh trường hợp làm thay đổi và giả mạo thẻ xác thực, nội dung của thẻ này được mã hóa bằng khóa bí mật $K_{AS,TGS}$ dùng chung của máy chủ xác thực AS và máy chủ cấp thẻ dịch vụ TGS.

Máy chủ xác thực AS gửi cho máy con C thẻ xác thực $Ticket_{TGS}$ sau khi đã mã hóa một lần nữa bằng khóa bí mật $K_{AS,C}$ dùng chung với người dùng (máy con C).

$$C \xleftarrow{\begin{matrix} Tickket_{TGS} = E(ID_C + AD_C + ID_{TGS} + T_1 + LT_1, K_{AS.TGS}) \\ E(Tickket_{TGS}, K_{AS.C}) \end{matrix}} AS$$

3- Sau khi nhận được thông tin gửi đến từ máy chủ xác thực AS, người dùng thực hiện giải mã thông tin đó để có thẻ xác thực $Ticket_{TGS}$. Rõ ràng, khi người dùng có đúng mật khẩu (chính là $K_{AS,C}$) thì mới giải mã thành công, ngược lại, việc xác thực xem như kết thúc không thành công. Như vậy, mật khẩu gốc của người dùng đã không bị gởi đi trực tiếp trên mạng. Giả sử việc giải mã thành công.

Sau khi đã có thẻ xác thực $Ticket_{TGS}$, từ máy con C người dùng có thể yêu cầu máy chủ cấp thẻ dịch vụ TGS cấp thẻ dịch vụ $Ticket_{SS}$ đối với máy chủ dịch vụ SS mà mình muốn. Khi đó, thông tin gởi đến cho máy chủ cấp thẻ dịch vụ TGS bao gồm: nhân dạng của máy chủ dịch vụ SS (ID_{SS}), nhân dạng của máy con C (ID_C), địa chỉ mạng của máy con C (AD_C) và thẻ xác thực $Ticket_{TGS}$.

$$C \xrightarrow{ID_{SS} + ID_C + AD_C + Ticket_{TGS}} TGS$$

4- Sau khi nhận được thông tin từ máy con C, máy chủ cấp thẻ dịch vụ TGS thực hiện việc giải mã thẻ xác thực $Ticket_{TGS}$. Nếu giải mã thành công và xác thực đúng thì máy chủ cấp thẻ dịch vụ TGS tạo ra thẻ dịch vụ $Ticket_{SS}$, bao gồm nhân dạng của máy con C (ID_C), địa chỉ mạng của máy con C (AD_C), nhân dạng của máy chủ dịch vụ SS (ID_{SS}), thời điểm tạo thẻ TS_2 và thời gian tồn tại của thẻ $Lifetime_2$. Dữ liệu này được mã hóa bằng khóa bí mật $K_{TGS,SS}$ dùng chung với máy chủ dịch vụ SS.

Máy chủ cấp thẻ dịch vụ TGS gửi cho máy con C thẻ dịch vụ $Ticket_{SS}$.

$$C \xleftarrow{Ticket_{SS} = E(ID_C + AD_C + ID_{SS} + TS_2 + LT_2, K_{TGS,SS})} TGS$$

5. Sau khi có thẻ dịch vụ $Ticket_{SS}$, người dùng có thể yêu cầu dịch vụ trên máy chủ dịch vụ SS bằng cách gửi cho máy chủ dịch vụ SS: nhân dạng của máy con C (ID_C), địa chỉ mạng của máy con C (AD_C) và thẻ dịch vụ $Ticket_{SS}$.

Máy chủ dịch vụ SS thực hiện giải mã thẻ dịch vụ $Ticket_{SS}$ và kiểm tra thông tin. Nếu thành công thì người dùng được sử dụng các dịch vụ trên máy chủ dịch vụ SS.

$$C \xrightarrow{ID_C + AD_C + Ticket_{SS}} SS$$

Hai vấn đề khác nảy sinh trong thủ tục xác thực cải tiến

Như vậy, thủ tục xác thực cải tiến trên đã giải quyết được 2 vấn đề đã nêu, đó là: dùng lại thẻ xác thực $Ticket_{TGS}$ và không gởi mật khẩu gốc (khóa bí mật dùng chung) trực tiếp trên mạng. Tuy nhiên, lại có thêm 2 vấn đề khác nảy sinh là:

- Thứ nhất, nếu thời gian tồn tại của các thẻ xác thực $Ticket_{TGS}$ / thẻ dịch vụ $Ticket_{SS}$ quá ngắn thì người dùng có thể phải đăng nhập lại để tạo thẻ mới. Trái lại, nếu thời gian này quá dài thì nguy cơ bị lấy cắp thẻ tăng lên. Do đó, khi nhận được các thẻ này, máy chủ cấp thẻ dịch vụ TGS hoặc máy chủ dịch vụ SS cần phải biết chắc rằng mình đang làm việc với đúng người dùng có tên đăng nhập chứa trong thẻ. Đây là vấn đề xác thực máy con C của máy chủ cấp thẻ đích vụ TGS và máy chủ dịch vụ SS.

- Thứ hai, song song với việc xác thực trên thì cũng cần phải có thao tác xác thực ngược lại từ máy con C đến máy chủ xác thực AS để loại trừ trường hợp chính máy chủ xác thực AS bị giả mạo.

Hai vấn đề về việc xác thực 2 chiều này được giải quyết bởi giao thức xác thực Kerberos.

1.4. Giao thức xác thực Kerberos

Đặc điểm của giao thức xác thực Kerberos

Kerberos là một giao thức xác thực mạng máy tính hoạt động bằng Ticket *(thẻ)*, cho phép người dùng tại các node truyền thông trên mạng không an toàn chứng minh nhận dạng của họ với nhau theo cách an toàn. Tức là các đối tác có thể xác thực lẫn nhau.

Giao thức này được đặt tên theo nhân vật Kerberos (hay Cerberus) trong truyện thần thoại Hy Lạp, là con chó bảo vệ ba đầu của thần Hades.

Người ta thiết kế giao thức này nhằm mục đích chủ yếu là cho các hoạt động trong mô hình Client - Server, và cung cấp phương thức xác thực lẫn nhau giữa client và server.

Các thông điệp trong giao thức Kerberos được bảo vệ để chống lại các kiểu tấn công *nghe lén* (Eavesdrop) và *phát lại* (Replay).

Giao thức Kerberos được xây dựng trên hệ mật mã khóa đối xứng, yêu cầu có một bên thứ ba đáng tin cậy, và có thể tùy chọn sử dụng mật mã khóa bất đối xứng trong một số giai đoạn xác thực.

Giao thức Kerberos được thiết kế dựa trên giao thức Needham-Schroeder.

Giao thức Kerberos sử dụng một bên thứ ba tham gia vào quá trình xác thực gọi là *trung tâm phân phối khóa* KDC (Key Distribution Center). Trung tâm này bao gồm hai thành phần là m*áy chủ xác thực AS* và *máy chủ cung cấp thẻ dịch vụ TGS*. *Thẻ* trong hệ thống xác thực Kerberos chính là các giấy chứng nhận cho tính hợp lệ của người dùng.

Mỗi thực thể sử dụng (cả Client và Server) trong hệ thống luôn chia sẻ một *khóa bí mật* dùng chung với máy chủ Kerberos. Việc sở hữu thông tin về khóa bí mật này được xem như là bằng chứng để chứng minh tính hợp lệ của thực thể.

Với mỗi *phiên giao dịch* (Session) giữa hai thực thể trong hệ thống, máy chủ Kerberos sẽ tạo ra một *khóa phiên* (Sesion Key) để dùng cho phiên giao dịch đó.

Phiên giao dịch (Session)

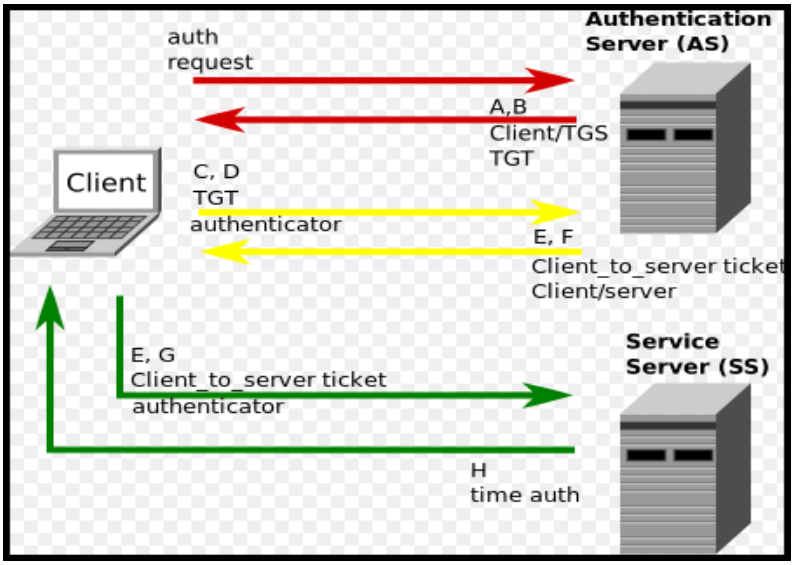
Một phiên giao dịch được mô tả một cách vắn tắt như sau:

- Trước tiên, người dùng xác thực mình với máy chủ xác thực AS để có thẻ xác thực.

- Tiếp theo chứng minh với máy chủ cấp thẻ dịch vụ TGS rằng mình đã được xác thực để được nhận thẻ dịch vụ.

- Sau cùng chứng minh với máy chủ dịch vụ SS rằng mình đã được chấp thuận để sử dụng dịch vụ.

Chi tiết một phiên giao dịch sử dụng giao thức xác thực Kerberos



*1. **C** yêu cầu thẻ xác thực*

Tại Client (máy khách C), người dùng nhập vào định danh máy khách ID_C và mật khẩu P_C của mình.

Client thực hiện hàm băm một chiều lên mật khẩu P_C để được mã băm $H(P_C)$. Mã băm này là khóa riêng PR_C của người dùng và sẽ được sử dụng cùng với khóa công khai PU_C tương ứng được tính.

Client gửi đến máy chủ xác thực AS gói tin AuthRequest để yêu cầu được cấp thẻ xác thực. Gói tin này gồm có các nội dung: định danh máy khách ID_C, địa chỉ máy khách AD_C, định danh máy chủ cấp thẻ dịch vụ ID_{TGS} và thời điểm yêu cầu T_1.

Nhận xét rằng mật khẩu P_C không bị gửi đi trực tiếp trên mạng.

$$C \xrightarrow{ID_C + AD_C + ID_{TGS} + T_1} AS$$

*2. **AS** cấp thẻ xác thực và khóa phiên giữa TGS và C*

Sau khi nhận được gói tin từ Client, máy chủ xác thực AS kiểm tra xem ID_C có nằm trong cơ sở dữ liệu người dùng của mình hay không. Nếu không thì việc xác thực là thất bại, phiên giao dịch kết thúc, nếu có thì tiếp tục.

Máy chủ xác thực AS tạo ra khóa phiên $K_{C.TGS}$ giữa Client và máy chủ cấp thẻ dịch vụ TGS. Khóa phiên này được mã hóa bằng khóa công khai PU_C của người dùng, đặt vào gói tin A và gửi nó cho Client

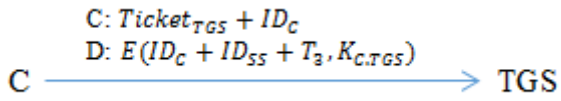
Máy chủ xác thực AS tiếp tục tạo ra thẻ xác thực $Ticket_{TGS}$, đặt vào gói tin B và gửi nó cho Client. Thẻ xác thực $Ticket_{TGS}$ bao gồm các nội dung: định danh máy khách ID_C, địa chỉ máy khách AD_C, thời điểm tạo thẻ T_2, thời gian tồn tại của thẻ LT_2, định danh máy chủ cấp thẻ dịch vụ ID_{TGS}, khóa phiên $K_{C.TGS}$, tất cả được mã hóa bằng khóa bí mật $K_{AS.TGS}$

$$\begin{aligned} &A: E(K_{C.TGS}, PU_C) \\ &B: Ticket_{TGS} = E(ID_C + AD_C + T_2 + LT_2 + ID_{TGS} + K_{C.TGS}, PU_C) \end{aligned}$$
$$C \xleftarrow{\qquad} AS$$

*3. **C** yêu cầu cấp thẻ dịch vụ*

Sau khi nhận được hai gói tin A và B từ máy chủ xác thực AS, trước tiên Client giải mã gói tin A bằng khóa riêng PR_C để được khóa phiên $K_{C.TGS}$.

Tiếp theo, để yêu cầu thẻ dịch vụ, Client gửi đến máy chủ cấp thẻ dịch vụ TGS gói tin C bao gồm: thẻ xác thực $Ticket_{TGS}$ và định danh máy khách ID_C, và sau đó là gói tin D có nội dung: định danh máy khách ID_C, định danh máy chủ dịch vụ ID_{SS}, thời điểm yêu cầu thẻ T_3, tất cả được mã hóa bằng khóa phiên $K_{C.TGS}$



*4. **TGS** cấp thẻ dịch vụ và khóa phiên giữa C và SS*

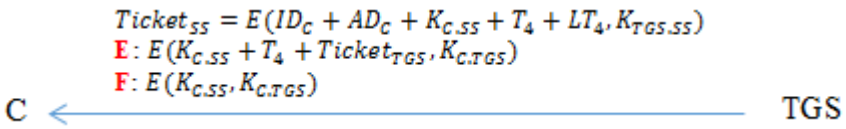
Sau khi nhận được hai gói tin C và D từ Client, trước tiên máy chủ cấp thẻ dịch vụ TGS giải mã thẻ xác thực $Ticket_{TGS}$ trong gói tin C để được khóa phiên $K_{C.TGS}$

Sau đó lại dùng khóa phiên này để giải mã gói tin D. Khi đó máy chủ cấp thẻ dịch vụ TGS nhận được ID_C và ID_{SS}.

Máy chủ cấp thẻ dịch vụ TGS so sánh ID_C nhận được với ID_C trong gói tin C để xác thực Client.

Trong trường hợp xác thực thành công, máy chủ cấp thẻ dịch vụ TGS tạo ra khóa phiên $K_{C.SS}$ giữa Client và máy chủ dịch vụ SS, và thẻ dịch vụ $Ticket_{SS}$ có nội dung: định danh máy khách ID_C, địa chỉ máy khách AD_C, khóa phiên $K_{C.SS}$, thời điểm tạo thẻ T_4, thời gian tồn tại của thẻ LT_4, tất cả được mã hóa bằng khóa bí mật $K_{TGS.SS}$

Máy chủ cấp thẻ dịch vụ TGS gửi cho Client hai gói tin E và F có nội dung như sau:

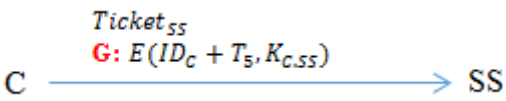


*5. **C** yêu cầu dịch vụ*

Sau khi nhận được hai gói tin E và F, trước tiên Client giải mã gói tin E bằng khóa phiên $K_{C.TGS}$ để được khóa phiên $K_{C.SS}$ và thẻ dịch vụ $Ticket_{SS}$

Client tiếp tục giải mã gói tin F bằng khóa phiên $K_{C.TGS}$ để một lần nữa được khóa phiên $K_{C.SS}$, và so sánh khóa phiên này với khóa phiên $K_{C.SS}$ nhận được từ gói tin E để xác thực

Nếu xác thực thành công, Client gửi cho máy chủ dịch vụ SS thẻ dịch vụ $Ticket_{SS}$ và gói tin G có nội dung: định danh máy khách ID_C, thời điểm yêu cầu dịch vụ T_5, tất cả được mã hóa bằng khóa phiên $K_{C.SS}$

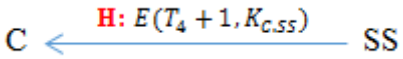


*6. **SS** cung cấp dịch vụ*

Sau khi nhận được thẻ dịch vụ $Ticket_{SS}$ và gói tin G, trước tiên máy chủ dịch vụ SS giải mã thẻ dịch vụ $Ticket_{SS}$ bằng khóa bí mật $K_{TGS.SS}$ để được khóa phiên $K_{C.SS}$, ID_C và T_4

Tiếp theo, máy chủ dịch vụ SS giải mã gói tin G bằng khóa phiên $K_{C.SS}$ vừa nhận được để được ID_C. So sánh ID_C này với ID_C trong thẻ dịch vụ $Ticket_{SS}$ để xác thực.

Nếu xác thực thành công thì máy chủ dịch vụ SS gửi cho Client gói tin H có nội dung:



*7. **C** sử dụng dịch vụ*

Khi nhận được gói tin H, Client giải mã gói tin H bằng khóa phiên $K_{C.SS}$ để được T_4+1. Client kiểm tra thời điểm yêu cầu thẻ dịch vụ $Ticket_{SS}$, nếu đúng thì bắt đầu sử dụng dịch vụ.

Kết hợp giữa nhiều hệ thống Kerberos (Lãnh địa Kerberos)

Một môi trường xác thực Kerberos đầy đủ là một hệ thống bao gồm: *máy chủ Kerberos* (Kerberos Server), *các máy chủ ứng dụng* (Application Servers) và *các máy con sử dụng dịch vụ* (Clients). Trong đó các điều kiện sau đây phải được thỏa:

- Máy chủ Kerberos phải có danh sách tất cả các *tên đăng nhập* và *mật khẩu* đã được mã hóa của các người dùng.

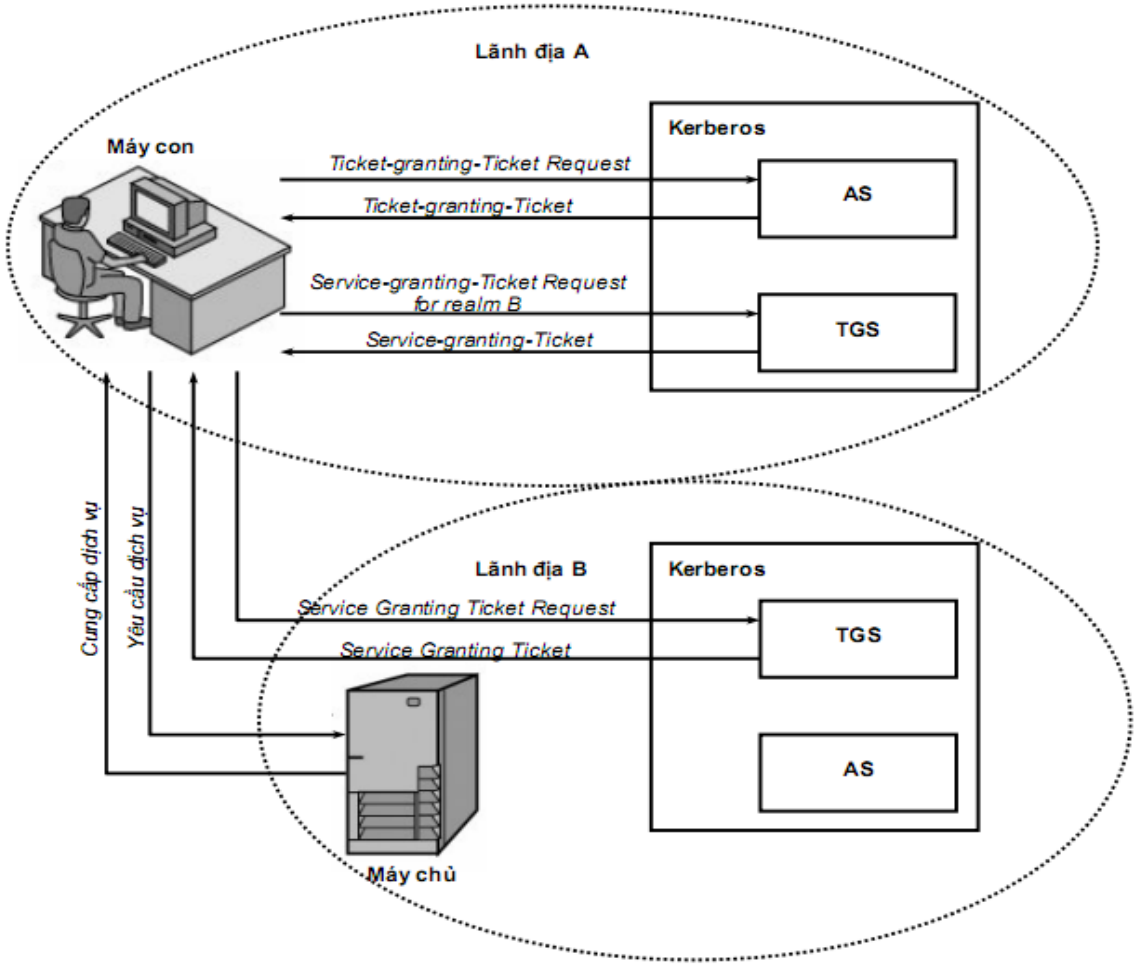
- Tất cả các máy con sử dụng dịch vụ đều phải đăng ký với máy chủ Kerberos.

- Máy chủ Kerberos sử dụng một *khóa bí mật* dùng chung với mỗi máy chủ ứng dụng còn lại.

- Tất cả các máy chủ ứng dụng đều phải đăng ký với máy chủ Kerberos.

Một môi trường thỏa mãn các điều kiện như vậy được gọi là *một lãnh địa Kerberos* (Kerberos Realm). Như vậy, máy chủ Kerberos, các máy chủ ứng dụng và máy con sử dụng dịch vụ thuộc các đơn vị quản lý khác nhau sẽ thuộc về các lãnh địa Kerberos khác nhau.

Giao thức xác thực Kerberos cũng bao gồm *các thủ tục cho phép kết hợp* các lãnh địa Kerberos lại để cung cấp dịch vụ một cách đồng nhất. Hình sau đây mô tả hoạt động của thủ tục kết hợp này: trong đó một client trong realm A yêu cầu sử dụng dịch vụ trong realm B.



(1) : Yêu cầu AS trong lãnh địa A cấp thẻ xác thực

(2) : AS trong lãnh địa A cấp thẻ xác thực

(3) : Yêu cầu TGS trong lãnh địa A cấp thẻ dịch vụ trong lãnh địa B

(4) : TGS trong lãnh địa A cấp thẻ dịch vụ

(5) : Yêu cầu TGS trong lãnh địa B cấp thẻ dịch vụ

(6) : TGS trong lãnh địa B cấp thẻ dịch vụ

(7) : Yêu cầu dịch vụ trong lãnh địa B

(8) : Máy chủ ứng dụng trong lãnh địa B cung cấp dịch vụ

2. BỘ GIAO THỨC IPSec (Internet Protocol Security Suite)

IPSec (IP Security) là một tập các giao thức, thuộc Networl Layer, được thiết kế bởi IETF (Internet Engineering Task Force) để cung cấp sự an toàn cho một packet IP. IPSec giúp tạo ra các packet xác thực và bí mật.

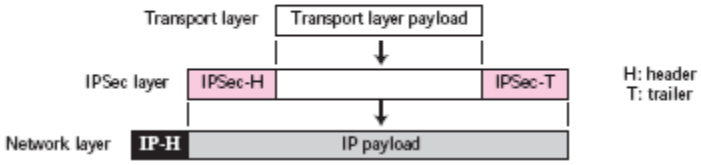
2.1. Hai chế độ (Two Modes)

IPSec hoạt động ở một trong hai chế độ khác nhau: chế độ vận chuyển (Transport Mode) và chế độ đường hầm (Tunnel Mode).

Chế độ vận chuyển

Trong **chế độ vận chuyển**, IPSec bảo vệ những gì được phân phối từ Transport layer đến Netwwork layer. Nói cách khác, chế độ vận chuyển bảo vệ tải trọng (Payload) được bao bọc trong Network layer, như được cho thấy trong Figure 30.1.

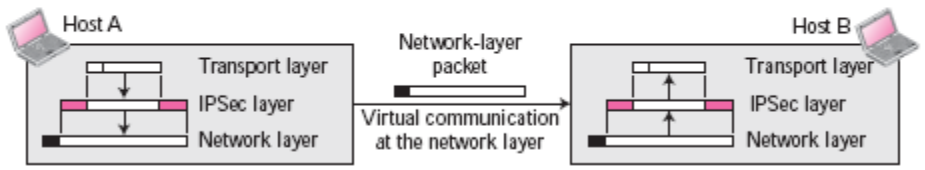
Figure 30.1 *IPSec in transport mode*



Chú ý rằng chế độ vận chuyển không bảo vệ toàn bộ IP packet, cụ thể là không bảo vệ IP header. Nói cách khác, chế độ vận chuyển không bảo vệ toàn bộ IP packet; nó chỉ bảo vệ packet đến từ Transport layer (IP layer payload). Trong chế độ này, IPSec header và trailer được thêm vào thông tin đến từ transport layer. IP header được thêm sau.

Chế độ vận chuyển thường được sử dụng khi chúng ta cần phải bảo vệ dữ liệu Host-to-Host (End-to-End). Host gửi sử dụng IPSec để xác thực và/hoặc mã hóa payload được đưa đến từ Transport layer. Host nhận sử dụng IPSec để kiểm tra sự xác thực và/hoặc giải mã IP packet nhận được và đưa nó đến Transport layer. Figure 30.2 cho thấy khái niệm này.

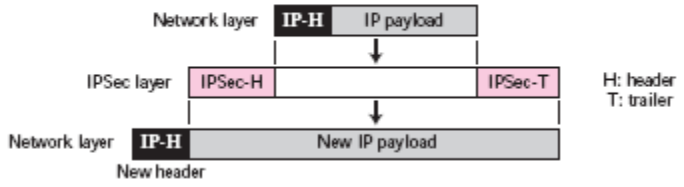
Figure 30.2 *Transport mode in action*



Chế độ đường hầm

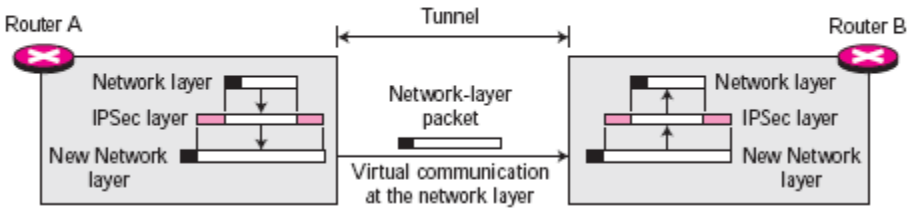
Trong **chế độ đường hầm**, IPSec bảo vệ toàn bộ IP packet. Nó lấy toàn bộ một IP packet, gồm cả header và payload, áp dụng các phương pháp mật mã/ xác thực vào, và sau đó thêm vào kết quả một IP header mới, như được cho thấy trong Figure 30.3.

Figure 30.3 *IPSec in tunnel mode*



IP header mới, như chúng ta vừa thấy, có những thông tin khác với IP header gốc. Chế độ đường hầm thường được sử dụng giữa hai router, giữa một host và một router, hoặc giữa router và một host, như được cho thấy trong Figure 30.4. Toàn bộ IP packet gốc được bảo vệ khỏi sự xâm nhập vào giữa sender và receiver, như thể toàn bộ packet đi qua một tunnel tưởng tượng.

Figure 30.4 *Tunnel mode in action*



So sánh (Comparison) chế độ vận chuyển và chế độ đường hầm

Trong chế độ vận chuyển, IPSec ở giữa giữa Transport layer và Network layer. Trong chế độ đường hầm, luồng xuất phát từ Network layer đến IPSec layer và rồi trở lại Network layer một lần nữa. Figure 30.5 so sánh hai chế độ này.

Figure 30.5 *Transport mode versus tunnel mode*

2.2. Hai giao thức an ninh (Two Security Protocols)

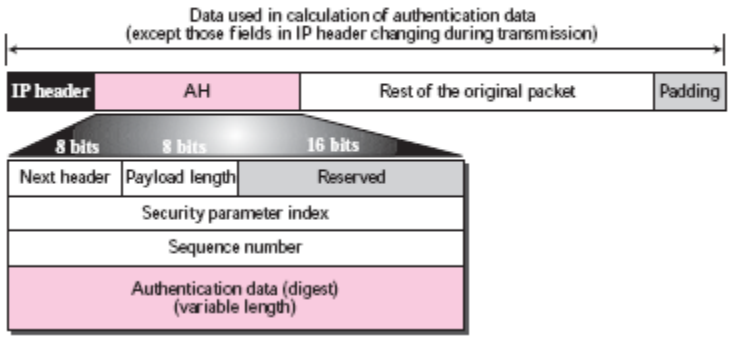
IPSec định rõ hai giao thức: giao thức *tiêu đề xác thực* AH (Authentication Header) và giao thức *bao gói tải trọng an ninh* ESP (Encapsulating Security Payload), để cung cấp việc xác thực và/hoặc mã hóa cho các IP packet ở mức IP.

Giao thức AH

Giao thức **AH** được thiết kế để xác thực Host nguồn và để đảm bảo cho tính toàn vẹn của payload được mang trong IP packet. Giao thức này sử dụng một hàm băm và một khóa bí mật đối xứng để tạo ra một *tóm tắt thông điệp* (Message Digest); tóm tắt này được xen vào trong tiêu đề xác thực. Sau đó tiêu đề xác thực này được đặt vào một vị trí thích hợp trong IP

packet, dựa trên chế độ vận chuyển hoặc chế độ đường hầm. Figure 30.6 cho thấy các field của tiêu đề xác thực và vị trí của nó trong chế độ vận chuyển.

Figure 30.6 *Authentication Header (AH) protocol*



Khi một IP packet có mang một tiêu đề xác thực, giá trị gốc của field về protocol của IP header được thay thế bởi giá trị 51. Một field bên trong tiêu đề xác thực này, field về next header, nắm giữ giá trị gốc của field về protocol (kiểu của payload đang được mang bởi datagram packet). Việc thêm một tiêu đề xác thực AH trong IP packet được thực hiện theo các bước:

1. Một tiêu đề xác thực AH được thêm vào trong payload và field về authentication data của nó được đặt là 0.

2. Field về padding có thể được thêm vào trong paload để làm cho chiều dài tổng cộng của IP packet đúng với một thuật toán băm cụ thể.

3. Việc băm dựa trên toàn bộ packet. Tuy nhiên, chỉ có các field nào của IP header không thay đổi trong khoảng thời gian truyền tải là được đưa vào trong việc tính toán tóm tắt thông điệp (field về authentication data: dữ liệu xác thực).

4. Dữ liệu xác thực này được xen vào tiêu đề xác thực.

5. IP header được thêm vào sau khi thay đổi giá trị của field về protocol thành 51.

Một mô tả ngắn gọn mỗi field trong tiêu đề xác thực AH:

□ **Next header.** 8-bit next header field định rõ kiểu của payload được mang bởi IP datagram (ví dụ TCP, UDP, ICMP hoặc OSPF).

□ **Payload length**. Tên của 8-bit field này là sai lạc. Nó không định rõ chiều dài của payload; nó định rõ chiều dài của tiêu đề xác thực ở dạng bội của 4-byte, nhưng nó không bao gồm 8 byte đầu.

□ **Security parameter index (SPI).** 32-bit SPI field này giữ vai trò của một bộ nhận dạng mạch ảo và giống nhau cho tất cả packet được gửi đi trong khoảng thời gian một kết nối được gọi là một *Liên kết an ninh* SA (Security Association) (được thảo luận sau này)

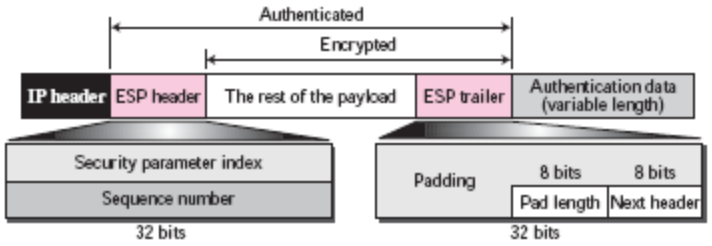
□ **Sequence number**. Một 32-bit sequence number field cung cấp thông tin sắp thứ tự cho một bộ các datagram. Các sequence number ngăn chặn sự phát lại. Chú ý rằng sequence number này không bị lặp lại ngay cả nếu một packet được truyền lại. Một sequence number không vòng quanh lại sau khi đạt đến 2^{32}; một kết nối mới phải được thiết lập.

□ **Authentication data**. Sau cùng, authentication data field này là kết quả của việc áp dụng hàm băm lên toàn bộ IP datagram ngoại trừ đối với các field bị thay đổi trong khoảng thời gian quá cảnh (Transit).

Giao thức EPS

Giao thức AH protocol không cung cấp tính bí mật, chỉ có cung cấp sự xác thực nguồn và tính toàn vẹn dữ liệu (chỉ sử dụng băm mà không sử dụng mã hóa). Sau này IPSec định rõ một giao thức có thể chọn để thay thế, đó là giao thức **ESP**, là cái cung cấp sự xác thực nguồn, tính toàn vẹn và tính bí mật. Giao thức ESP thêm vào IP packet một header và trailer. Chú ý rằng authentication data field trong ESP được thêm vào ở cuối IP packet, điều này làm cho việc tính toán của ra thành ra dễ hơn. Figure 30.7 cho thấy vị trí của ESP header và trailer.

Figure 30.7 *Encapsulating Security Payload (ESP)*



Khi một IP datagram mang theo một ESP header và trailer, giá trị của field về protocol trong IP header là 50. Một field bên trong ESP trailer, field về next header, nắm giữ giá trị gốc của field về protocol (kiểu của payload được mang theo bởi IP datagram, ví dụ TCP hoặc UDP). Thủ tục của ESP đi theo các bước:

1. Một ESP trailer được thêm vào payload.
2. Payload và trailer được mã hóa.
3. Một ESP header được thêm vào payload.
4. ESP header, payload và ESP trailer được dùng để tạo ra dữ liệu xác thực (Authentication data).
5. Authentication data được thêm vào sau ESP trailer.
6. IP header được thêm vào sau khi thay đổi giá trị của field về protocol là 50.

Các field của ESP header và ESP trailer là như sau:

□ **Security parameter index (SPI).** 32-bit SPI field tương tự như cái được định rõ cho giao thức AH.

□ **Sequence number**. 32-bit sequence number field tương tự như cái được định rõ cho giao thức AH

□ **Padding**. Variable-length field này (0 đến 255 byte) toàn 0 phục vụ một padding.

□ **Pad length**. 8-bit pad length field định rõ số padding byte. Giá trị này ở giữa 0 và 255; giá trị tối đa là hiếm.

□ **Next header**. 8-bit next header field tương tự như cái đã được định rõ trong giao thức AH. Nó phục vụ cùng mục đích như field về protocol trong IP header trước khi bọc lại.

□ **Authentication data**. Sau cùng, authentication data field là kết quả của việc áp dụng một lược đồ xác thực vào các phần của datagram. Hãy chú ý đến sự

khác nhau giữa dữ liệu xác thực trong AH và ESP. Trong AH, IP header được bao gồm trong việc tính toán dữ liệu xác thực; trong ESP thì không.

AH so với ESP

Giao thức ESP được thiết kế sau khi giao thức AH được sử dụng rồi. ESP làm bất cứ thứ gì mà AH làm và có thêm chức năng được bổ sung (Tính bí mật). Vấn đề là vì sao chúng ta cần AH? Câu trả lời là chúng ta không cần. Tuy nhiên, việc thực thi AH đã được bao hàm trong một vài sản phẩm thương mại, có nghĩa là AH vẫn là một phần của Internet cho đến khi các sản phẩm này rút lui dần.

2.3. Các dịch vụ được cung cấp bởi (Services Provided by IPSec)

Hai giao thức AH và ESP cung cấp vài dịch vụ an ninh cho các packet tại Network layer. Table 30.1 cho thấy danh sách các dịch vụ được cung cấp bởi giao thức AH và ESP.

Table 30.1 *IPSec services*

Services	*AH*	*ESP*
Access control	Yes	Yes
Message authentication (message integrity)	Yes	Yes
Entity authentication (data source authentication)	Yes	Yes
Confidentiality	No	Yes
Replay attack protection	Yes	Yes

Access Control – Kiểm soát truy cập

IPSec cung cấp việc kiểm soát truy cập một cách gián tiếp bằng cách sử dụng cơ sở dữ liệu Liên kết an ninh SAD (Security Association Database), như chúng ta sẽ thấy trong phần tiếp theo. Khi một packet đến một đích nếu không có Liên kết an ninh đã được thiết lập cho packet này thì packet bị hủy.

Message Integrity – Tính toàn vẹn thông điệp

Tính bí mật thông điệp được lưu giữ trong cả AH và ESP. Một tóm tắt dữ liệu được tạo ra bởi sender để được kiểm tra bởi receiver.

Entity Authentication – Xác thực thực thể

Liên kết an ninh và tóm tắt dữ liệu được gửi đi bởi sender bằng băm có khóa sẽ xác thực sender của dữ liệu trong cả AH và ESP.

Confidentiality – Tính bí mật

Việc mã hóa thông điệp trong ESP cung cấp tính bí mật. Tuy nhiên, AH không cung cấp tính bí mật. Nếu tính bí mật được cần đến, người ta sẽ sử dụng ESP thay vì AH.

Replay Attack Protection – Chống tấn công phát lại

Trong cả hai giao thức AH và ESP, tấn công phát lại bị ngăn chặn bởi việc sử dụng các Sequence number và một Sliding receiver window (cửa sổ trượt nơi người nhận). Mỗi IPSec header chứa một sequence number duy nhất khi Liên kết an ninh được thiết lập.

Number này bắt đầu từ 0 và tăng lên cho đến khi giá trị đạt đến $2^{32} - 1$. Khi sequence number đạt đến giá trị tối đa, nó được đặt trở lại là 0 và, cùng lúc Liên kết an ninh cũ bị xóa và một cái mới được thiết lập. Để ngăn chặn việc tạo ra các packet nhân bản (Duplicate packet), IPSec ủy nhiệm cho một fixed sized window tại receiver. Size của window này được xác định bởi receiver với một giá trị mặc nhiên là 64.

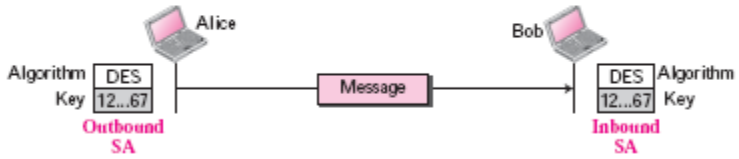
2.4. Liên kết an ninh SA (Security Association)

Liên kết an ninh là một nét riêng rất quan trọng của IPSec. IPSec yêu cầu một quan hệ logic, được gọi là **Liên kết an ninh**, giữa hai host. Đầu tiên phần này thảo luận về ý tưởng và sau đó cho thấy cách sử dụng Liên kết an ninh đó trong IPSec.

Ý tưởng về Liên kết an ninh (Idea of Secutity Association)

Một Liên kết an ninh là một hợp đồng giữa hai bên đối tác; nó tạo ra một kênh truyền an toàn giữa họ. Chúng ta hãy giả định rằng Alice cần trao đổi thông tin với Bob theo một phương hướng duy nhất. Nếu Alice và Bob chỉ quan tâm đến mặt bí mật về sự an ninh, họ có thể tạo ra một khóa bí mật (Secret Key) chia sẻ với nhau. Chúng ta có thể nói là có hai Liên kết an ninh giữa Alice và Bob; một cái là *xuất ngoại* (outbound) và một cái là *hồi hương* (inbound). Mỗi cái trong số chúng lưu trữ giá trị của khóa trong một biến và tên của thuật toán mã hóa/ giải mã trong một biến khác. Alice sử dụng thuật toán và khóa để mã hóa một thông điệp gửi cho Bob; Bob sử dụng thuật toán và khóa này khi cần giải mã thông điệp nhận được. Figure 30.8 cho thấy một Liên kết an ninh đơn giản.

Figure 30.8 *Simple SA*

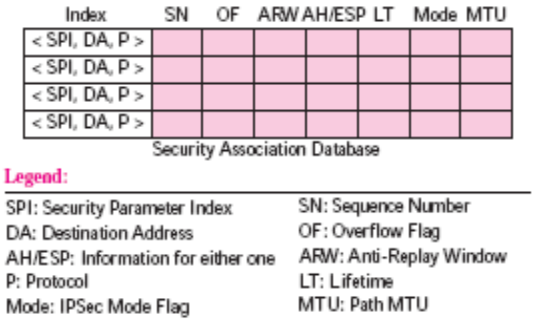


Liên kết an ninh này có thể là rắc rối phức tạp nếu hai bên cần đến tính toàn vẹn của thông điệp và việc xác thực thông điệp. Khi đó, mỗi Liên kết an ninh cần đến những dữ liệu khác nữa ví dụ như thuật toán cho tính toàn vẹn của thông điệp, khóa, và các tham số khác nữa. Và có thể phức tạp nhiều hơn nữa nếu hai bên cần sử dụng các thuật toán đặc biệt và các tham số đặc biệt cho các giao thức khác nhau, ví dụ như IPSec AH hoặc IPSec ESP.

Cơ sở dữ liệu Liên kết an ninh SAD (Security Association Database)

Một Liên kết an ninh có thể rất phức tạp. Điều này đặc biệt là đúng nếu Alice muốn gửi thông điệp cho nhiều người, và Bob cần nhận các thông điệp từ nhiều người. Ngoài ra, mỗi site cần phải có cả Liên kết an ninh xuất ngoại và Liên kết an ninh hồi hương để cho phép việc trao đổi thông tin hai chiều. Nói cách khác, chúng ta cần một bộ các Liên kết an ninh có thể được tập hợp vào trong một cơ sở dữ liệu. Cơ sở dữ liệu này này được gọi **Cơ sở dữ liệu liên kết an ninh**. Cơ sở dữ liệu liên kết an ninh này có thể được xem như là một bảng hai chiều mà mỗi dòng định rõ một Liên kết an ninh đơn. Thông thường, có hai Liên kết an ninh, một là hồi hương và một là xuất ngoại. Figure 30.9 cho thấy khái niệm về Liên kết an ninh hồi hương và Liên kết an ninh xuất ngoại cho một thực thể.

Figure 30.9 *SAD*



Khi một host cần gửi đi một packet phải mang theo một IPSec header, host này phải tìm entry tương ứng trong SDA xuất ngoại để tìm thông tin áp dụng vào việc đảm bảo sự an toàn cho packet. Tương tự, khi một host nhận được một packet mang theo một IPSec header, host này cần tìm entry tương ứng trong SAD hồi hương để tìm thấy thông tin cho việc kiểm tra sự an toàn của packet. Việc tìm kiếm này phải cụ thể theo nghĩa host nhận cần phải chắc chắn rằng thông tin đúng đắn được sử dụng cho việc xử lý packet đó. Mỗi entry trong một SAD hồi hương được chọn lọc bằng các sử dụng một bộ ba dấu hiệu: Chỉ số tham số an ninh SPI (Security Parameter Index) là một con số dài 32 bit định rõ Liên kết an ninh tại đích, địa chỉ đích (Destination Address) và giao thức là (Protocol) AH hoặc ESP.

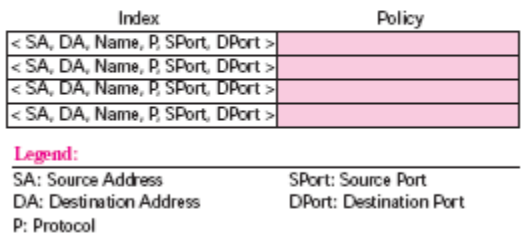
Chính sách an ninh SP (Security Policy)

Một nét riêng quan trọng khác của IPSec là **chính sách an ninh**, là cái định rõ loại an ninh được áp dụng vào một packet khi nó được gửi đi hoặc khi nó đến. Trước khi sử dụng SAD, đã được thảo luận trong phần trước, một host phải định rõ chính sách đã vạch sẵn cho packet đó.

Cơ sở dữ liệu chính sách an ninh SPD (Security Policy Database)

Mỗi host đang sử dụng giao thức của IPSec cần nắm giữ một **cơ sở dữ liệu chính sách an ninh**. Một lần nữa, tồn tại sự cần thiết đối với một SPD hồi hương và một SPD xuất ngoại. Mỗi entry trong SPD có thể được truy cập bằng cách sử dụng một bộ 6 dấu hiệu: địa chỉ nguồn (Source Address), địa chỉ đích (Destination Address), tên (Name), giao thức (Protocol), cổng nguồn (Source Port), và cổng đích (Destination Port), như được cho thấy trong Figure 30.10.

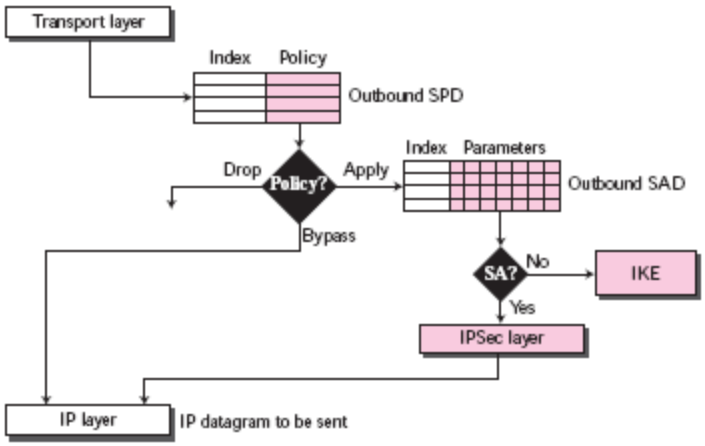
Figure 30.10 *SPD*



Địa chỉ nguồn và địa chỉ đích có thể là unicast, multicast, hoặc wildcard address. Tên thường định rõ một DNS của thực thể. Giao thức thì hoặc là AH hoặc là ESP. Cổng nguồn và cổng đích là các địa chỉ cổng cho tiến trình xử lý đang chạy tại host nguồn và host đích.

SPD xuất ngoại. Khi một packet được gửi đi ra, SPD xuất ngoại được tham khảo. Figure 30.11 cho thấy việc xử lý một packet bởi một sender.

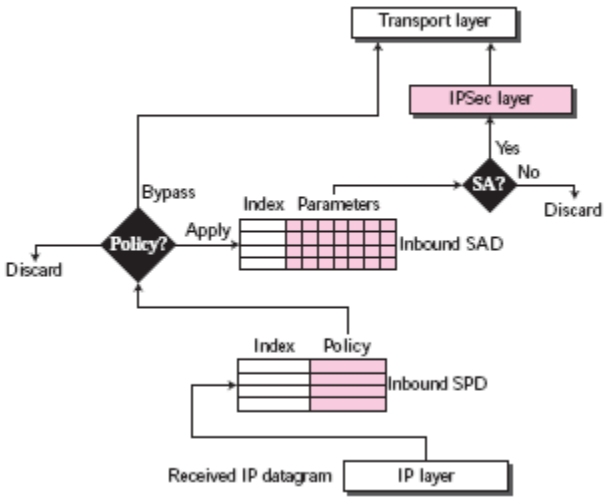
Figure 30.11 *Outbound processing*



Input cho SPD xuất ngoại là một bộ sáu dấu hiệu; output là một trong ba trường hợp sau: drop (packet không thể được gửi đi), bypass (tiêu đề an ninh tránh né), và apply (áp dụng sự an toàn theo SAD; nếu không có SAD thì tạo ra).

SPD hồi hương. Khi một packet đến, SPD hồi hương được tham khảo. Mỗi entry trong SPD hồi hương cũng được truy cập bằng cách sử dụng một bộ 6 dấu hiệu như vậy. Figure 30.12 cho thấy việc xử lý một packet bởi một receiver.

Figure 30.12 *Inbound processing*



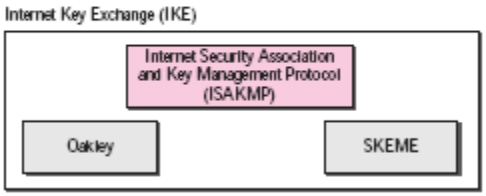
Input cho SPD hồi hương là một bộ 6 dấu hiệu; output là một trong ba trường hợp sau: discard (bỏ rơi packet), bypass (tránh sự an ninh và phân phát packet cho Transport layer), và apply (áp dụng chính sách bằng cách sử dụng SAD).

2.5. Trao đổi khóa IKE (Internet Key Exchange)

Trao đổi khóa **IKE** là một giao thức được thiết kế để tạo ra cả Liên kết an ninh hồi hương và Liên kết an ninh xuất ngoại. Như chúng ta đã thảo luận trong phần trước, khi một người ngang hàng cần gửi đi một IP packet, thì cần tham khảo SPD để biết có hay không một Liên kết an ninh cho kiểu lưu thông đó. Nếu không có, IKE được gọi đến để thiết lập một SPD.

IKE là một giao thức phức tạp dựa trên ba giao thức khác: Oakley, SKEME và ISAKMP, như được cho thấy trong Figure 30.13.

Figure 30.13 *IKE components*



Giao thức **Oakley** được phát triển bởi Hilarie Orman. Nó là giao thức tạo khóa.

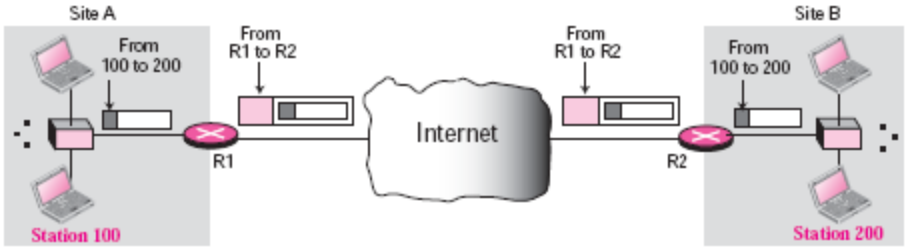
Giao thức **SKEME**, được thiết kế bởi Hugo Krawcyzk, là một giao thức khác cho việc trao đổi khóa. Nó sử dụng mã hóa khóa bất đối xứng cho việc xác thực thực thể trong một giao thức trao đổi khóa.

Giao thức **ISAKMP (Internet Security Association and Key Management Protocol)** là một giao thức được thiết kế bởi NSA (National Security Agency), là cái trên thực tế thực hiện việc trao đổi được xác định trong IKE. Nó định rõ vài packet, giao hức, và các tham số cho phép sự trao đổi của IKE diễn ra trong các thông điệp đã chuẩn hóa và được định dạng, để tạo ra các Liên kết an ninh.

2.6. Mạng riêng ảo VPN (Virtual Private Network)

Một trong các ứng dụng của IPSec là **mạng riêng ảo VPN**. Một VPN là một công nghệ đang thu được sự ưa chuộng trong các tổ chức lớn sử dụng Internet toàn cầu cho cả việc truyền thông ở tổ chức bên trong (Intra-Organization) và ở tổ chức qua lại (Inter-Organization), nhưng đòi hỏi một sự riêng tư trong việc truyền thông trong tổ chức bên trong của họ. VPN là một mạng riêng tư nhưng ảo. Nó riêng tư vì nó đảm bảo tính riêng tư bên trong tổ chức. Nó ảo vì nó không sử dụng WAN riêng thực; mạng này là công cộng về mặt vật lý nhưng là riêng tư ảo. Figure 30.14 cho thấy ý tưởng về một VPN.

Figure 30.14 *Virtual private network*

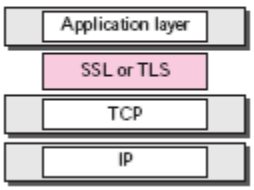


Router R1 và R2 sử dụng công nghệ VPN để đảm bảo sự riêng tư cho tổ chức. VPN sử dụng giao thức ESP của IPSec trong chế độ đường hầm. Một datagram riêng tư, gồm cả header, được bao gói trong một ESP packet. Router tại biên của địa điểm gửi sử dụng địa chỉ của chính nó và địa chỉ của router tại địa điểm đích trong datagram mới. Mạng công cộng (Internet) chịu trách nhiệm mang đi packet từ R1 đến R2. Những người ngoài cuộc không thể giải mã các nội của packet hoặc địa chỉ nguồn và đích. Việc giải mã diễn ra tại R2, nó tìm thấy địa chỉ đích của packet và phân phối packet.

3. HAI GIAO THỨC SSL (Secure Sockets Layer) VÀ TLS (Transport Layer Security)

Hiện nay, hai giao thức nổi trội trong việc cung cấp sự an ninh tại Transport layer là: giao thức **SSL** và giao thức **TLS**. Cái sau trên thực tế là một phiên bản theo IETF của cái trước. Chúng ta thảo luận về SSL trong phần này; TLS thì rất giống vậy. Figure 30.15 cho thấy vị trí của SSL và TLS trong mô hình Internet.

Figure 30.15 *Location of SSL and TLS in the Internet model*



Một trong các mục tiêu của các giao thức này là cung cấp việc xác thực server và client, tính bí mật của dữ liệu, và tính toàn vẹn của dữ liệu. Các chương trình client/server ở Application layer, ví dụ như HTTP sử dụng các service của TCP có thể bao lại dữ liệu của chúng trong các SSL packet. Nếu server và client có khả năng chạy các chương trình SSL/TLS thì client có thể sử dụng URL *https://...* thay vì *http://...* để cho phép các thông điệp HTTP được gói lại trong các SSL/TLS packet. Ví dụ, các số thẻ tín dụng (Credit Card Number) có thể được chuyển tải một cách an toàn thông qua Internet đối với những người mua hàng trực tuyến.

3.1. Kiến trúc SSL (SSL Architecture)

SSL được thiết kế để cung cấp các dịch vụ về sự an ninh và nén cho các dữ liệu được sinh ra từ Application layer. Điển hình là, SSL có thể nhận dữ liệu xuất phát từ bất kỳ giao thức nào ở Application layer, nhưng thường là giao thức HTTP. Dữ liệu đã nhận được từ ứng dụng sẽ được nén (tùy chọn), được đánh dấu và được mã hóa. Sau đó dữ liệu được chuyển sang một giao thức đáng tin cậy ở Transport layer, ví dụ như giao thức TCP. Netscape đã phát triển SSL trong năm 1994. Version 2 và 3 được phát hành năm 1995. Trong phần này chúng ta thảo luận về SSLv3.

Các dịch vụ (Services)

Giao thức SSL cung cấp vài dịch vụ trên dữ liệu nhận được từ Application layer.

□ Phân mãnh (Fragmentation)

Đầu tiên, SSL chia dữ liệu thành các block 2^{14} byte hoặc ít hơn.

□ Nén (Compression)

Mỗi mãnh dữ liệu được nén lại bằng cách sử dụng một trong các phương pháp nén không mất dữ liệu được đàm phán giữa client và server. Dịch vụ này là tùy chọn.

□ Tính toàn vẹn thông điệp (Message Integrity)

Để bảo vệ tính toàn vẹn của dữ liệu, SSL sử dụng một hàm băm có khóa (Keyed Hash Function) để tạo ra một mã xác thực thông điệp MAC (Message Authentication Code).

□ Tính bí mật (Confidentiality)

Để cung cấp tính bí mật, dữ liệu gốc và MAC được mã hóa bằng cách sử dụng mật mã khóa bất đối xứng

□ Tạo khung (Framing)

Một header được thêm vào payload đã được mã hóa. Sau đó, payload này được chuyển sang một giao thức dáng tin cậy ở Transport layer.

Thuật toán trao đổi khóa (Key Exchange Algorithms)

Để trao đổi một thông điệp đã được xác thực và bí mật, client và server mỗi bên cần một tập các bí mật mật mã (Cryptographic Secret). Tuy nhiên, để tạo ra các bí mật này, một bí mật chính có trước (Pre-master Secret) phải được thiết lập giữa hai bên. SSL xác định vài phương pháp trao đổi khóa để tạo ra bí mật chính có trước này.

Thuật toán mã hóa/ giải mã (Encryption/ Decryption Algorithms)

Client và server cũng cần đồng ý với nhau về các thuật toán để mã hóa/ giải mã.

Thuật toán băm (Hash Algorithms)

SSL sử dụng thuật toán băm để cung cấp tính toàn vẹn của thông điệp (xác thực thông điệp). Vài thuật toán băm cũng được định rõ cho mục đích này.

Bộ mật mã (Cipher Suite)

Sự phối hợp giữa thuật toán trao đổi khóa, thuật toán băm và thuật toán mã hóa xác định một bộ mật mã cho mỗi phiên SSL.

Thuật toán nén (Compression Algorithm)

Việc nén là tùy chọn trong SSL. Không có các thuật toán nén cụ thể được định rõ. Do đó một hệ thống có thể sử dụng bất cứ thuật toán nén nào mà nó muốn.

Sinh tham số mật mã (Cryptographic Parameter Generation)

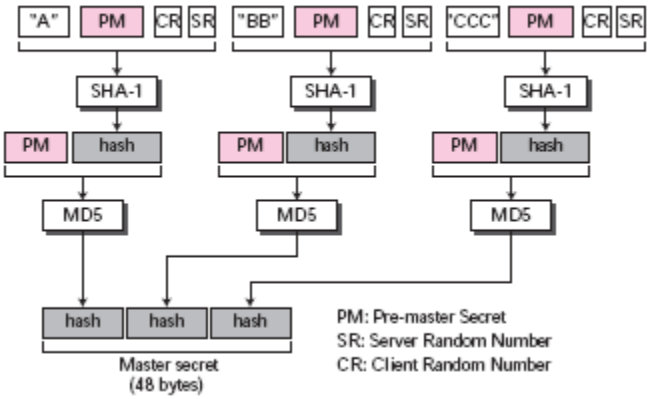
Để đạt được tính toàn vẹn và tính bí mật của thông điệp, SSL cần sáu bí mật mật mã (Cryptographic Secrets), bốn khóa (Keys) và hai vector ban đầu IV (Initialization Vector). Client cần một khóa cho việc xác thực thông điệp, một khóa cho việc mã hóa, và một IV như một block gốc trong tính toán. Server cần giống vậy. SSL yêu cầu là các khóa cho một hướng thì khác với các khóa cho các hướng khác. Nếu có một tấn công trong một hướng, các hướng khác không bị tấn công. Các tham số được sinh ra bằng cách sử dụng thủ tục có các bước như sau:

1. Client và server trao đổi hai con số ngẫu nhiên; một được tạo ra bởi client và một được tạo ra bởi server.

2. Client và server trao đổi một **bí mật chính có trước** bằng cách sử dụng một trong các thuật toán trao đổi khóa đã được xác định rõ.

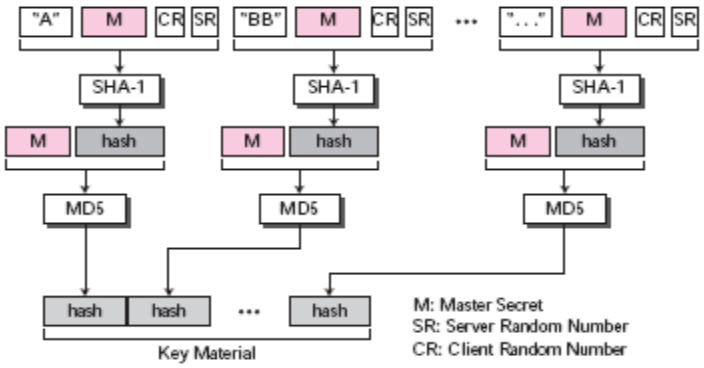
3. Một **bí mật chính** dài 48 bit được khởi tạo bằng cách áp dụng hai hàm băm (SHA-1 và MD5), như được cho thấy trong Figure 30.16.

Figure 30.16 *Calculation of master secret from pre-master secret*



4. Bí mật chính này được sử dụng để tạo ra nguyên liệu khóa có chiều dài thay đổi được (Variable-length Key Material) bằng cách áp dụng cùng một tập hợp các hàm băm và bằng cách thêm vào đầu bằng các hằng khác nhau, như được cho thấy trong Figure 30.17. Module này được lặp lại cho đến khi nguyên liệu khóa có kích thước thích hợp được tạo ra.

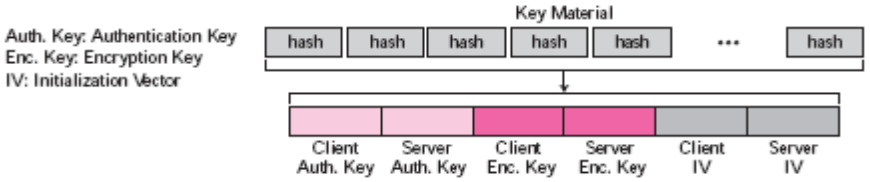
Figure 30.17 *Calculation of key material from master secret*



Chú ý rằng chiều dài của block nguyên liệu khóa phụ thuộc vào bộ mật mã đã chọn và kích thước của các khóa cần cho bộ này.

5. Sáu bí mật khác nhau được rút trích từ nguyên liệu khóa, như được cho thấy trong Figure 30.18.

Figure 30.18 *Extractions of cryptographic secrets from key material*



Các kết nối và phiên (Sessions and Connections)

SSL phân biệt một **kết nối** với một **phiên.** Phiên là một sự liên kết (Association) giữa client và server. Sau khi một phiên được thiết lập, hai bên có thông tin chung ví dụ như: bộ nhận dạng phiên (Session Identifier), chứng nhận xác thực (Certificate Authenticating) của mỗi người trong số họ, phương pháp nén (Compression Method) nếu cần, bộ mật mã (Cipher Suite), và một bí mật chính (Master Secret) được sử dụng để tạo ra các khóa cho việc mã hóa xác thực thông điệp.

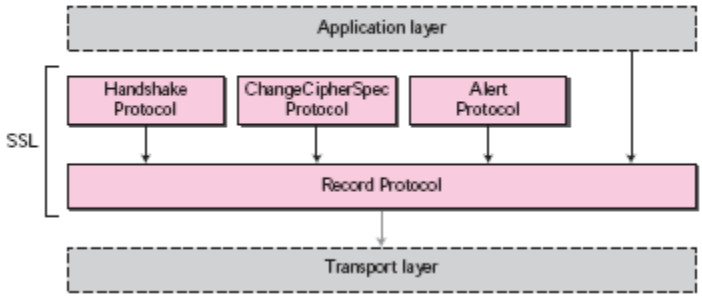
Để hai thực thể trao đổi dữ liệu, việc thiết lập một phiên là cần, nhưng không đủ; họ cần tạo ra một kết nối giữa chính họ. Hai thực thể trao đổi hai con số ngẫu nhiên và tạo ra, bằng cách sử dụng bí mật chính, các khóa và các tham số cần thiết cho việc trao đổi thông điệp liên quan đến việc xác thực và sự riêng tư.

Một phiên có thể gồm nhiều kết nối. Một kết nối giữa hai thực thể có thể bị kết thúc và được tái thiết lập trong cùng một phiên. Khi một kết nối bị kết thúc, hai thực thể cũng có thể kết thúc phiên, nhưng là không bắt buộc. Một phiên có thể bị tạm thời ngưng và được hoạt động lại.

3.2. Bốn giao thức (Four Protocols)

Chúng ta đã thảo luận về ý tưởng của SSL mà không cho thấy cách SSL hoàn thành nhiệm vụ của nó. SSL định nghĩa 4 giao thức trong hai layer, như được cho thấy trong Figure 30.19.

Figure 30.19 *Four SSL protocols*



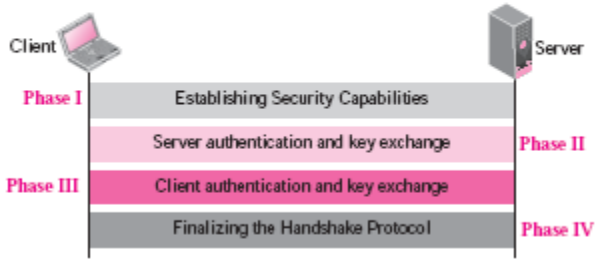
Record Protocol là carrier (vật mang). Nó mang các thông điệp đến từ ba giao thức khác cũng như dữ liệu đến từ Application layer. Các thông điệp xuất phát từ Record Protocol là các payload đến Transport layer, bình thường là TCP. **Handshake Protocol** cung cấp các tham số an ninh (Security Parameter) cho Record Protocol. Nó thiết lập một bộ mật mã và cung cấp các khóa và các tham số an ninh. Nó cũng xác thực server đối với client và client đối với server nếu cần. **ChangeCipherSpec Protocol** được sử dụng cho việc báo hiệu sự sẵn

sàng của các bí mật mật mã. **Alert Protocol** được sử dụng để thông báo về các tình trạng bất thường. Chúng ta sẽ thảo luận ngắn gọn các giao thức này trong phần này.

Handshake Protocol

Handshake protocol sử dụng các thông điệp để đàm phán về bộ mật mã, để xác thực client với server và server với client nếu cần, và để trao đổi thông tin cho việc xây dựng các bí mật mật mã. Việc bắt tay được thực hiện trong 4 phase, như được cho thấy trong Figure 30.20.

Figure 30.20 *Handshake Protocol*



Phase I: Thiết lập công suất an ninh (Establishing Security Capacity)

Trong phase I, client và server thông báo các khả năng an ninh của họ và chọn những cái thuận tiện cho cả hai. Trong phase này, một ID phiên được thiết lập và một bộ mật mã được chọn. Các bên đồng ý trên một phương pháp nén cụ thể. Sau cùng, hai con số ngẫu nhiên được chọn, một bởi client và một bởi server, để được sử dụng cho việc tạo ra một bí mật chính như chúng ta đã thấy trước đây.

Sau pha I, client và server biết được phiên bản của SSL, các thuật toán mật mã, các phương pháp mã hóa, và hai con số ngẫu nhiên để sinh khóa.

Phase II: Trao đổi khóa và xác thực server (Server Key Exchange and Authentication)

Trong phase II, server xác thực bản thân nó nếu cần. Sender có thể gửi đi chứng nhận, khóa công khai của nó, và cũng có thể yêu cầu về các chứng nhận xuất phát từ client.

Sau pha II, server được xác thực đối với client, và client biết được khóa công khai của server nếu có yêu cầu.

Phase III: Trao đổi khóa và xác thực client (Client Key Exchange and Authentication)

Phase III được thiết kế để xác thực client.

Sau pha III, client được xác thực đối với server, và cả client và server biết được điều bí mật có trước.

Phase IV: Hoàn tất và kết thúc (Finalizing and Finishing)

Trong phase IV, client và server gửi đi các thông điệp để thay đổi sự đặc tả mật mã (Cipher Specification) và để kết thúc handshake protocol.

ChangeCipherSpec Protocol

Chúng ta đã là thấy việc đàm phán về bộ mật mã và việc sinh ra các bí mật mật mã được hình thành dần dần trong khoảng thời gian của Handshake Protocol. Vấn đề bây giờ là:

Khi nào hai bên có thể sử dụng các bí mật tham số này này? SSL bắt buộc các bên không thể sử dụng các tham số hoặc bí mật này cho đến khi họ gửi đi hoặc nhận được một thông điệp đặc biệt, thông điệp ChangeCipherSpec, là cái được trao đổi trong khoảng thời gian của Handshake Protocol và được định rõ trong giao thức ChangeCipherSpec. Lý do là vấn đề không chỉ là việc gửi đi hoặc việc nhận một thông điệp. Sender và receiver cần hai trạng thái, không phải là một. Một trạng thái, trạng thái còn để treo đó (Pending State), giữ lại vết của các tham số và các bí mật. Một trạng thái khác, trạng thái hoạt động (Active State), nắm giữ các tham số và bí mật được sử dụng bởi Record Protocol để đánh dấu/ xác minh hoặc mã hóa/ giải mã các thông điệp. Ngoài ra, mỗi trạng thái nắm giữ hai tập hợp các giá trị: *read* (hồi hương) và *write* (xuất ngoại).

Alert Protocol

SSL sử dụng Alert Protocol cho việc thông báo lỗi và các tình trạng bất thường. Nó chỉ sử dụng một thông điệp mô tả vấn đề và mức của nó (warning hoặc fatal).

Thông thường giao thức này có hai mức báo động là:

- *Mức cảnh báo (Warning Level):* là mức chỉ đơn giản là thông báo cho đầu kia các tình trạng bất thường đang diễn ra.

- *Mức chết người (Fatal Level):* là mức yêu cầu kết thúc kết nối TLS/SSL hiện hành. Các kết nối khác trong cùng phiên giao dịch có thể vẫn được duy trì nhưng phiên giao dịch không được thiết lập thêm kết nối mới.

Alert protocol truyền báo động bằng thông điệp báo động được nén và mã hóa. Một thông điệp báo động được hình thành bởi 2 trường:

- *Mức khốc liệt (Severity Level)*: trường này có chiều dài 1 byte có giá trị 1 hoặc 2, tùy thuộc vào mức độ khốc liệt. Thông điệp với giá trị 1 là *thông điệp cảnh báo,* với giá trị 2 là *thông điệp chết người*

- *Mô tả báo động (Alert Description)*: trường này dài 1 byte mô tả đặc trưng của thông điệp báo động. Giá trị của trường này là một trong 12 số cụ thể và có thể có 1 trong các nghĩa như sau:

• Với thông điệp chết người, nội dung của thông điệp có thể là như sau:

1- *Unexpected Message:* Nhận được một thông điệp không phù hợp.

2- *Bad Record MAC:* Thông điệp MAC không hợp lệ đã được nhận.

3- *Decompression Failure:* Chức năng giải nén đã nhận được một input không chính xác (ví dụ quá dài) nên thực hiện không thành công.

4- *Handshake Failure:* Phía gởi thông điệp cho biết không thể thương lượng về tập các tham số bảo mật theo các tùy chọn có sẵn.

5- *Illegal Parameters:* Một trường nào đó trong thông điệp bắt tay nằm ngoài phạm vi hoặc không phù hợp với các trường khác

• Với thông điệp cảnh báo, nội dung của thông điệp có thể là như sau:

6- *Close Notify:* Thông báo cho người nhận biết rằng người gửi sẽ không gửi thêm bất kỳ thông điệp nào nữa. Kết thúc kết nối.

7- *No Certificate:* Khi nhận được yêu cầu cung cấp chứng nhận, nếu không có chứng nhận nhận được nào thích hợp thì gởi thông điệp này.

8- *Bad Certificate:* Chứng nhận đã bị hỏng (ví dụ có chữ ký không thể thẩm tra)

9- *Unsupported Certificate:* Chứng nhận đã nhận không được hỗ trợ.

10- *Certificate Revoked:* Chứng nhận đã bị người ký hủy bỏ, thu hồi.

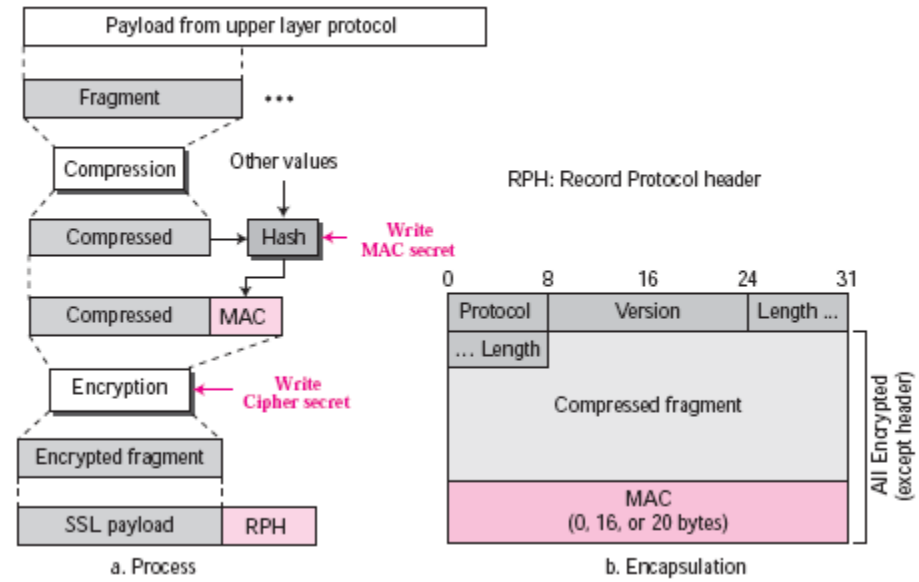
11- *Certificate Expired:* Chứng nhận đã hết hạn sử dụng, hết hiệu lực.

12- *Certificate Unknown:* Khi có một số vấn đề không thể xác định khác xảy ra trong quá trình xử lý chứng nhận, làm cho nó không thể chấp nhận được thì gửi thông điệp này

Record protocol

Record Protocol mang các thông điệp xuất phát từ layer trên hơn (Handshake Protocol, ChangeCipherSpec Protocol, Alert Protocol, hoặc application layer). Thông điệp này được phân mảnh và không bắt buộc bị nén; một mã xác thực thông điệp MAC được thêm vào thông điệp bị nén này bằng cách sử dụng hàm băm được đàm phán. Mảnh được nén và MAC được mã hóa bằng cách sử dụng encryption algorithm được đàm phán. Cuối cùng, header được thêm vào thông điệp đã được mã hóa. Figure 30.21 cho thấy tiến trình xử lý này tại sender. Tiến trình xử lý tại receiver là ngược lại.

Figure 30.21 *Processing done by the Record Protocol*



So sánh TLS/SSL với IPSec

TLS/SSL và IPSec là hai bộ giao thức bảo mật tương đồng với nhau về chức năng. Cả hai đều được thiết kế để bảo mật an toàn dữ liệu được truyền đi trên các kết nối bằng các cơ chế xác thực và mã hóa. Tuy nhiên, chúng vẫn có *những điểm khác biệt* như sau:

• TLS/SSL hoạt động ở Application Layer/ Presentation Layer, do đó nó được gắn kết với *không gian người dùng (User Space)* trong các hệ thống đầu cuối. IPSec hoạt động ở Network Layer, nên được tích hợp vào trong *chức năng của hệ điều hành*. Đây chính là sự khác nhau cơ bản nhất giữa TLS/SSL và IPSec.

• Cả TLS/SSL và IPSec đều cung cấp chức năng mã hóa (Encryption) để đảm bảo tính bí mật dữ liệu (Confidentiality), và xác thực (Authentication) để đảm bảo tính toàn vẹn dữ liệu (Integrity). Tuy nhiên, TLS/SSL đơn giản hóa các kỹ thuật này để áp dụng trong mô hình của nó, trong khi IPSec bao gồm đầy đủ các chi tiết hơn, và do đó, khi tổ hợp lại sẽ xuất hiện nhiều lỗi tương thích trong nội bộ IPSec.

• IPSec là thành phần của hệ điều hành, do đó, để triển khai IPSec thì phải thay đổi cấu hình hệ điều hành nhưng không cần thay đổi cấu hình chương trình ứng dụng. Ngược lại, TLS/SSL hoạt động ở mức người dùng nên phải cài đặt với từng ứng dụng cụ thể (ví dụ mail, web, …) mà không cần khai báo với hệ điều hành,

Vì những khác biệt trên đây, TLS/SSL thường được sử dụng để bảo vệ kết nối cho từng ứng dụng cụ thể, đặc biệt là với các ứng dụng Web, E-mail. Trong khi đó, IPSec thường được dùng để xây dựng các mạng riêng ảo (VPN) rồi trên cơ sở đó mới triển khai các dịch vụ ứng dụng.

4. SET (Secure Electronic Transaction)

4.1. Tổng quan về SET

Đặc điểm

SET (Giao dịch điện tử an toàn) là một chuẩn về giao thức truyền thông (Communication Protocol Standard) được sử dụng để bảo đảm an toàn cho các giao dịch bằng thẻ tín dụng (Credit Card) trên một mạng không an toàn, cụ thể là trên liên mạng Internet. Giao thức SET không phải là một hệ thống thanh toán, mà là một tập hợp các giao thức bảo mật và các format (định dạng) cho phép người dùng sử dụng cơ sở hạ tầng thanh toán bằng thẻ tín dụng hiện tại trên mạng mở theo cách an toàn.

Một điều cần chú ý là SET hoạt động bằng cách truy xuất trực tiếp đến tầng Transport Layer/ Internet Layer mà không dùng các giao thức khác ở tầng Application Layer. Tuy vậy hoạt động của SET không ảnh hưởng đến các giao thức bảo mật khác như IPSec hoặc TLS/SSL.

Giao thức SET phiên bản 1 được đề xuất năm 1996 (*MasterCard* và *Visa* chủ trì), sau đó được nhiều nhà sản xuất khác tham gia phát triển (như Microsoft, IBM, Netscape, RSA, Terisa và Verisign).

Mục đích

Giao thức SET thiết lập các giao dịch thanh toán bằng thẻ tín dụng với với mục đích:

- Hỗ trợ sự tin cậy về thông tin.

- Đảm bảo tính toàn vẹn cho các yêu cầu thanh toán và các dịch vụ liên quan đến dữ liệu.

- Hỗ trợ cơ chế xác thực giữa người bán hàng (Merchant) và người mua hàng (Cardholder) với nhau.

Chức năng

Giao thưc SET có các chức năng sau:

• *Mã hóa đảm bảo bí mật thông tin:* SET phải có khả năng bảo mật thông tin, đặc biệt là thông tin về tài khoản ngân hàng khi được trao đổi qua mạng. SET còn phải có khả năng ngăn chặn người bán hàng biết số thẻ tín dụng (Credit Card Number) của người mua hàng (Cardholder), cũng là người chủ thẻ. Thuật toán mã hóa DES được dùng để thực thi chức năng này.

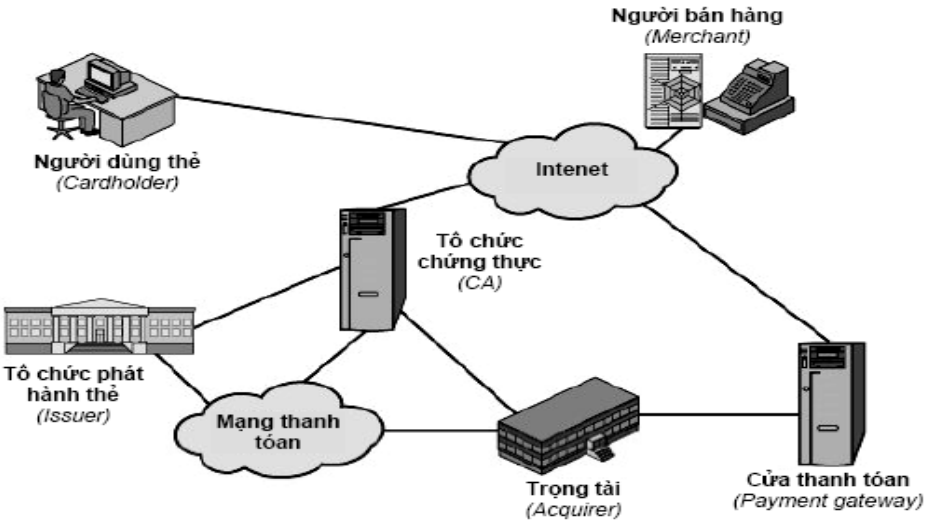
• *Xác thực đảm bảo toàn vẹn dữ liệu*: SET phải có khả năng đảm bảo an toàn dữ liệu của các thông tin về việc đặt hàng, thanh toán, thông tin cá nhân khi được gởi từ một người mua hàng đến người bán hàng. Kỹ thuật chữ ký số DSA với hàm băm SHA-1 được dùng để thực thi chức năng này (trong một số thông điệp của SET, HMAC được dùng thay cho DSA).

• *Xác thực tài khoản của người mua hàng:* SET phải có khả năng cho phép người bán hàng xác minh người dùng thẻ là chủ nhân hợp lệ của tài khoản đang đề cập. Để thực hiện chức năng này, SET dùng chuẩn xác thực X.509 version 3.

• *Xác thực người bán hàng:* SET phải có khả năng cho phép người mua hàng xác thực rằng người bán hàng có quan hệ với một tổ chức tài chính chấp nhận thanh toán qua thẻ. Chức năng này cũng được thực hiện bằng cách dùng X.509 version 3.

Các thành phần chính của SET

Trong giao thức SET có các thành phần chính được mô tả trong hình sau:



- *Người mua hàng (Cardholder):* là người dùng thẻ tín dụng để thực hiện các giao dịch thanh toán trên Internet.

- *Người bán hàng (Merchant):* là một cá nhân hay tổ chức bán hàng, hoặc một dịch vụ trên mạng (thông qua web hoặc email). Người bán hàng phải có khả năng chấp nhận thanh toán bằng thẻ, và phải có quan hệ với một tổ chức tài chính nào đó.

- *Tổ chức phát hành thẻ (Issuer):* là một tổ chức tài chính (thường là ngân hàng) phát hành thẻ tín dụng. Tổ chức này có trách nhiệm thanh toán theo yêu cầu của người dùng thẻ.

- *Ngân hàng mua lại (Acquirer):* còn gọi là trọng tài, là tổ chức tài chính cung cấp dịch vụ thanh toán, xử lý các giao dịch trực tuyến và đảm bảo về mặt tài chính cho giao dịch, có quan hệ với người bán hàng, thực hiện việc xác thực tài khoản và thanh toán của người mua hàng.

Khi có một giao dịch mua bán, trọng tài sẽ kiểm tra tài khoản của người mua hàng để thông báo cho người bán hàng biết số dư trong tài khoản của người mua là có đủ để thực hiện giao dịch hay không. Sau khi giao dịch mua bán hàng được thực hiện, trọng tài thực hiện việc chuyển tiền từ tài khoản của người mua hàng sang tài khoản của người bán hàng, đồng thời liên hệ với các tổ chức phát hành thẻ để yêu cầu chuẩn

chi cho giao dịch.

- *Cửa thanh toán (Payment Gateway):* là thành phần chịu trách nhiệm xử lý các thông báo thanh toán (Payment Message) được điều hành bởi trọng tài hoặc một tổ chức thứ 3 được chỉ định. Cửa thanh toán giao tiếp giữa SET và hệ thống thanh toán của ngân hàng để thực hiện các thao tác xác thực và thanh toán. Như vậy, người bán hàng thật ra trao đổi các thông báo với cửa thanh toán thông qua mạng Internet, sau đó, cửa thanh toán mới liên kết đến hệ thống xử lý tài chính của trọng tài.

- *Tổ chức cấp chứng nhận CA (Certification Authority):* là thành phần có chức năng tạo ra các chứng nhận (Certificate) theo chuẩn X.509v3 để cấp cho người mua hàng, người bán hàng và cửa thanh toán.

Sự thành công của SET phụ thuộc vào sự tồn tại của CA. Thông thường, CA được tổ chức theo một mô hình phân cấp với nhiều CA liên hệ với nhau.

Các bước trong một giao dịch SET

Để thực hiện một giao dịch SET thì khách hàng và người bán hàng cần phải có các thông tin của chính mình như sau:

- Khách hàng mở tài khoản tại một ngân hàng có dịch vụ thanh toán qua mạng (ví dụ MasterCard, VisaCard, …) và trở thành người mua hàng/ người chủ thẻ (Cardholder). Khi đó khách hàng nhận được một chứng thực X.509v3, được ký bởi ngân hàng bằng chữ ký số (Digital Signature), trong đó chứa khóa công khai RSA của khách hàng và ngày hết hạn.

- Người bán hàng (Merchant) phải có 2 chứng thực khác nhau chứa khóa công khai cho hai mục đích: ký nhận các thông báo (Message Signing) và trao đổi khóa (Key Exchange). Ngoài ra, người bán hàng cũng có một bản sao chứng thực của cửa thanh toán (Payment Gateway).

Khi đó, một giao dịch SET điển hình gồm các bước sau đây:

1. Khách hàng đặt hàng: thao tác này được thực hiện qua website của người bán hàng hoặc qua email.

2. Xác nhận người bán hàng: người bán hàng gởi chứng thực của mình cho người mua hàng để chứng minh tính sở hữu của mình đối với một kho hàng nào đó.

3. Thông tin đặt hàng và thanh toán được thực hiện: người mua hàng gởi thông tin đặt hàng và thanh toán cho người bán hàng cùng với chứng thực của mình. Thông tin thanh toán (số thẻ tín dụng) được mã hoá sao cho người bán hàng không thể thấy được nhưng có thể kiểm tra tính hợp lệ của nó.

4. Người bán hàng yêu cầu xác thực việc thanh toán thông qua cửa thanh toán.

5. Người bán hàng xác nhận đơn đặt hàng bằng cách gởi thông báo cho người mua hàng.

6. Người bán hàng giao hàng (hoặc bắt đầu cung cấp dịch vụ) cho người mua hàng.

7. Người bán hàng yêu cầu thanh toán thông qua cửa thanh toán.

4.2. Chữ ký song song (Dual Signature) trong SET

Khái niệm chữ ký song song

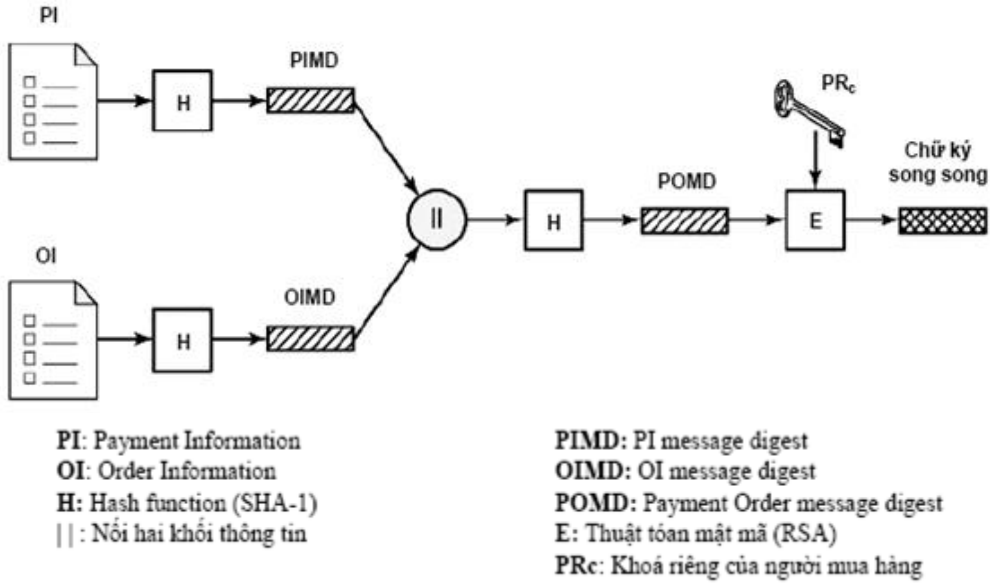
Chữ ký song song là một thuật ngữ được dùng trong SET để diễn tả sự kiện là *chữ ký*

của một người trên hai thông báo gửi cho hai người khác nhau là như nhau. Ví dụ, khi mua hàng qua mạng, khách hàng gởi *thông tin đặt hàng OI* (Order Information) với chữ ký của mình cho người bán hàng, đồng thời gởi *thông tin thanh toán PI* (Payment Information) cho ngân hàng cũng với cùng một chữ ký giống vậy.

Về nguyên tắc, ngân hàng không cần biết chi tiết về thông tin đặt hàng, và người bán hàng cũng không cần biết chi tiết về thông tin thanh toán. Chữ ký song song được sử dụng trong trường hợp này để tránh các tranh chấp xảy ra khi thông tin đặt hàng và thông tin thanh toán không khớp nhau.

Quy trình tạo và xác thực chữ ký song song

Hình sau mô tả quy trình tạo chữ ký song song trên thông tin đặt hàng (OI) và thông tin thanh toán (PI)



PI: Payment Information
OI: Order Information
H: Hash function (SHA-1)
|| : Nối hai khối thông tin

PIMD: PI message digest
OIMD: OI message digest
POMD: Payment Order message digest
E: Thuật toán mật mã (RSA)
PRc: Khoá riêng của người mua hàng

- Trước tiên, người mua hàng áp dụng hàm băm lên thông tin thanh toán PI và thông tin đặt hàng OI (dùng SHA-1), tiếp theo hai giá trị băm này được nối với nhau và được băm một lần nữa. Sau cùng, giá trị băm được mã hóa bằng khóa riêng PR_C của chính người mua hàng để tạo thành chữ ký song song DS

$$DS = E(H(H(PI) + H(OI)), PRc)$$

Chữ ký song song DS này được gửi cho người bán hàng và cả ngân hàng. Quy trình này cho thấy người bán hàng chỉ thấy H(PI) mà không thấy chính PI, ngân hàng chỉ thấy H(OI) mà không thấy chính OI, do đó sự riêng tư được bảo vệ.

- Khi người bán hàng nhận được OI, H(PI) và DS thì người bán hàng xác nhận chữ ký của người mua hàng bằng cách tính hai giá trị: H(H(PI) + H(OI)) và D(DS, PUc), trong đó PUc là khóa công khai của khách hàng. Nếu hai giá trị này bằng nhau, thì chữ ký xem như chính xác và đơn đặt hàng được chấp nhận.

- Khi ngân hàng nhận được H(OI), PI và DS thì ngân hàng xác nhận chữ ký của người mua hàng bằng cách tính hai giá trị: H(H(OI)+ H(PI)) và D(DS, PUc), trong đó PUc là khóa công khai của khách hàng. Nếu hai giá trị này bằng nhau, thì chữ ký xem như chính xác và lệnh thanh toán được chấp nhận.

4.3. Thực hiện thanh toán trong SET

Xử lý thanh toán (Payment Processing) là công đoạn quan trọng nhất trong giao thức

SET. Quá trình xử lý thanh toán gồm 3 thủ tục chính như sau:

1. Yêu cầu mua hàng (Purchase Request)

Sau khi người mua hàng hoàn tất các công việc chọn hàng và đặt mua trên mạng thì thủ tục yêu cầu mua hàng mới được bắt đầu. Chú ý rằng thao tác chọn hàng và đặt mua được thực hiện trên các kết nối bình thường (như e-mail hay web) mà không cần có sự tham gia của giao thức SET. Quá trình yêu cầu mua hàng được thực hiện thông qua 4 bản tin: *Initiate Request, Initiate Response, Purchase Request, Purchase Response*

Để gởi được các bản tin này trong SET đến người bán hàng, người mua hàng cần có một bản sao các chứng thực của người bán hàng và cửa thanh toán. Bản tin *Initiate Request* được sử dụng để yêu cầu người bán hàng cung cấp các chứng thực cần thiết cho người mua hàng.

Người bán hàng sẽ trả lời bản tin *Initiate Request* bằng một bản tin hồi đáp *Initiate Response* trong đó có chứa giá trị ngẫu nhiên (nonce) đã được tạo ra trước đó bởi người mua hàng, một giá trị ngẫu nhiên khác do người bán hàng tạo ra, nhân dạng của giao dịch hiện hành, cùng với các chứng thực của chính người bán hàng và cửa thanh toán. Tất cả các thông tin này được xác thực bởi chữ ký của người bán hàng.

Người mua hàng xác minh các chứng thực nhận được, sau đó tạo ra thông tin đặt hàng (OI) và thông tin thanh toán (PI), trong đó có chứa nhân dạng giao dịch mà người bán hàng vừa tạo ra trước đó. Người mua hàng chuẩn bị bản tin *Purchase Request*. Bản tin này chứa các thông tin sau đây:

• Các thông tin liên quan đến việc thanh toán bao gồm: PI, chữ ký song song, OIMD và một phong bì số (Digital Envelope). Các thông tin này được mã hoá bằng khoá bí mật Ks do người mua hàng tạo ra cho từng phiên giao dịch.

• Các thông tin liên quan đến đơn đặt hàng bao gồm OI, chữ ký song song, PIMD. Chú ý rằng OI được gởi đi trực tiếp mà không cần mã hoá.

• Chứng thực của người mua hàng.

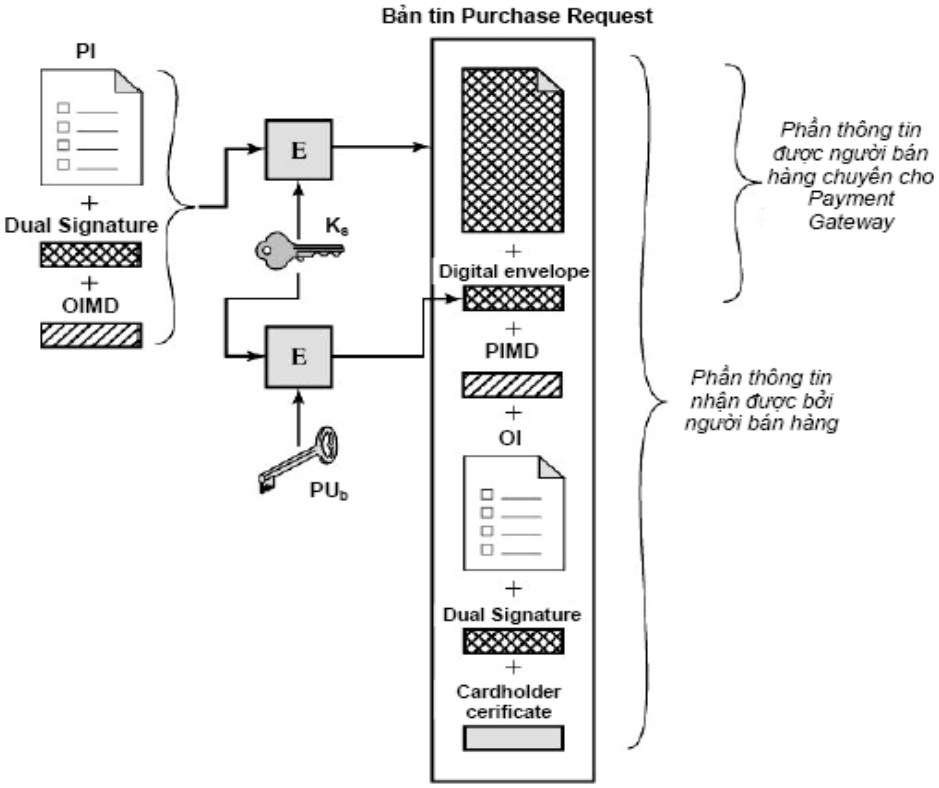
Khi người bán hàng nhận được *Purchase Request*, họ sẽ thực hiện các thao tác sau đây:

- Xác minh chứng thực của người mua hàng.

- Kiểm chứng chữ ký song song của người mua hàng.

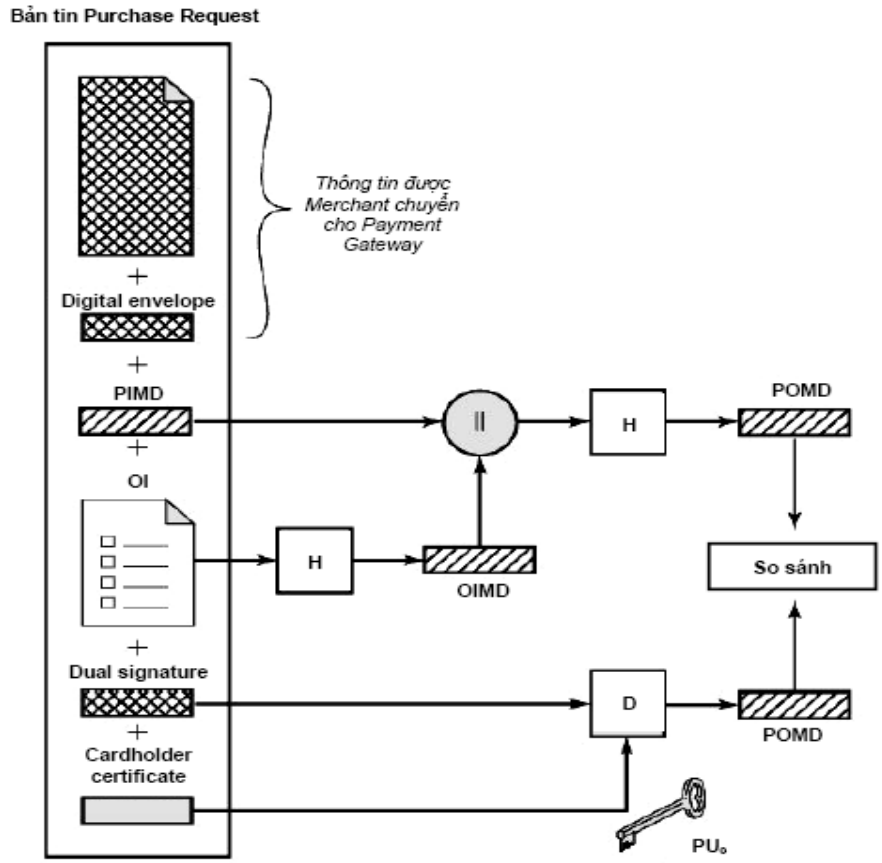
- Xử lý đơn đặt hàng và chuyển thông tin thanh toán cho cửa thanh toán để kiểm tra.

- Gởi bản tin *Purchase Response* cho người mua hàng. Bản tin *Purchase Response* chứa các thông tin để chấp nhận đơn đặt hàng và các tham chiếu đến số nhân dạng giao dịch tương ứng. Thông tin này được ký bởi người bán hàng và gởi cho người mua hàng cùng với chứng thực của người bán. Người mua hàng khi nhận được bản tin *Purchase Response* sẽ tiến hành kiểm tra chữ ký và chứng thực của người bán hàng.



2. Xác thực thanh toán (Payment Authentication)

Đây là thủ tục mà người bán hàng xác thực tính hợp lệ của người mua hàng thông qua cửa thanh toán. Quá trình xác thực nhằm bảo đảm rằng giao dịch này được chấp thuận bởi tổ chức phát hành thẻ (Issuer), và do đó người bán hàng sẽ được đảm bảo được thanh toán. Quá trình này được thực hiện thông qua hai bản tin: *Authorization Request, Authorization response*



Bản tin *Authorization Request* được người bán hàng gởi đến cửa thanh toán bao gồm các thông tin sau:

- Thông tin liên quan đến việc mua hàng, bao gồm: PI, chữ ký song song, OIMD và phong bì số.
- Thông tin liên quan đến xác thực bao gồm: nhân dạng giao dịch, được mã hoá bằng khoá bí mật do người bán hàng tạo ra và phong bì số, được mã hoá bằng khoá công khai của cửa thanh toán.
- Các chứng thực của người mua hàng và người bán hàng.

Khi nhận được bản tin *Authorization Request,* cửa thanh toán thực hiện các thao tác sau:

- Xác minh tất cả các chứng thực.
- Giải mã phong bì số của khối thông tin mua hàng.
- Xác minh chữ ký của người bán hàng.
- Giải mã phong bì số của khối thông tin xác thực.
- Xác minh chữ ký song song.
- Xác minh nhân dạng giao dịch (Transaction ID).
- Yêu cầu xác thực từ ngân hàng phát hành thẻ.

Nếu nhận được thông tin xác thực thành công từ ngân hàng phát hành thẻ, cửa thanh toán sẽ hồi đáp bằng bản tin *Authorization Response* trong đó chứa các thông tin sau:

- Thông tin liên quan đến xác thực bao gồm: khối thông tin xác thực được ký bởi cửa thanh toán và mã hoá bằng khoá bí mật do cửa thanh toán tạo ra, ngoài ra còn có phong bì số.

• Thông tin liên quan đến thực hiện thanh toán.

• Chứng thực của cửa thanh toán. Với thông tin xác thực này, người bán hàng đã có thể bắt đầu giao hàng hoặc cung cấp dịch vụ cho người mua hàng.

3. Thực hiện thanh toán (Payment Capture)

Để thực hiện thanh toán, người bán hàng thực hiện một giao dịch với cửa thanh toán, được gọi là *Capture Transaction*. Quá trình này được thực hiện thông qua hai bản tin: *Capture Request, Capture Response*.

Trong bản tin *Capture Request*, người bán hàng tạo ra thông tin yêu cầu thanh toán, trong đó có khối lượng thanh toán và nhân dạng giao dịch (Transaction ID), cùng với thông tin xác thực nhận được trước đó từ cửa thanh toán, chữ ký và chứng thực của người bán hàng.

Cửa thanh toán nhận được bản tin này, giải mã và thực hiện các bước kiểm tra cần thiết trước khi yêu cầu ngân hàng phát hành thẻ chuyển tiền cho người bán hàng. Cuối cùng, cửa thanh toán sẽ thông báo cho người bán hàng bằng bản tin *Capture Response*.