Tích hợp thuật toán mật mã mới trong mạng riêng ảo OpenSwan

Nguyễn Như Tuấn, Phạm Văn Hưởng, Phạm Quốc Hoàng

***Tóm tắt*— Bài báo này trình bày phương pháp tích hợp thuật toán mật mã mới vào giải pháp mã nguồn mở OpenSwan để xây dựng mạng riêng ảo (VPN). OpenSwan là bộ công cụ mã nguồn mở đang được sử dụng rộng rãi để triển khai VPN, đặc biệt là trong các hệ thống điện toán đám mây. Mặc dù, có nhiều thuật toán mật mã đã được tích hợp trong OpenSwan, nhưng trong một số trường hợp cụ thể, người sử dụng muốn dùng một thuật toán bảo mật riêng, không có sẵn trong OpenSwan, để bảo vệ dữ liệu nhạy cảm của họ. Do vậy, việc nghiên cứu, tích hợp một thuật toán mật mã mới có ý nghĩa thực tiễn và ứng dụng cao. Trên cơ sở phân tích nguyên lý hoạt động, mã nguồn hệ thống, chúng tôi đề xuất mô hình tích hợp, thay thế các thuật toán mật mã trong OpenSwan khi triển khai VPN. Mô hình này đã được triển khai thử nghiệm và kết quả thực nghiệm cho thấy phương pháp đề xuất có hiệu năng tốt và dễ tích hợp.**

***Abstract:* This paper presents a method to integrate a new cryptography algorithm into the open source system OpenSwan for building a virtual private network (VPN). The OpenSwan plays an important role in confidential data in the virtual private network, specially for building a cloud computing system. Although there are several cryptography algorithms installed in OpenSwan, users need to use the other new algorithm in some typical cases for protecting their sensitive data so the studying to install a new cryptography algorithm in OpenSwan is necessary and useful. Based on analyzing priciple operating and source code of system, we proposed the model used to replace a crypto algorithm in OpenSwan. Experiment results show that the proposed method is easy and good performance.**

***Từ khóa*— Mạng riêng ảo; OpenSwan; IPSec; Giao thức trao đổi khóa mạng; Tổ hợp mật mã kết hợp xác thực.**

***Keywords*— VPN; OpenSwan; IPSec; IKE; AEAD (Authenticated Encryption with Associated Data).**

Bài báo được nhận ngày 05/12/2016. Bài báo được nhận xét bởi phản biện thứ nhất vào ngày 14/12/2016 và được chấp nhận đăng vào ngày 17/01/2017. Bài báo được nhận xét bởi phản biện thứ hai vào ngày 14/12/2016 và được chấp nhận đăng vào ngày 26/12/2016.

I. GIỚI THIỆU

Ngày nay, với sự phát triển mạnh mẽ của công nghệ thông tin, việc sử dụng Internet trong các lĩnh vực kinh tế - xã hội ngày càng trở nên quan trọng và phổ biến. Các thông tin trao đổi qua mạng Internet ngày càng đa dạng cả về nội dung và hình thức, trong đó có nhiều thông tin cần được bảo mật ở mức cao, đòi hỏi tính chính xác và độ tin cậy. VPN là một giải pháp phù hợp và phổ biến để giải quyết vấn đề này. Trong thực tế, có nhiều công cụ để triển khai VPN, bao gồm cả các công cụ thương mại (ví dụ như của các hãng Cisco, Juniper...) hay các công cụ mã nguồn mở (như OpenVNN, OpenStrong, OpenSwan...). Việc sử dụng các công cụ VPN thương mại sẽ được đảm bảo về chất lượng, độ tin cậy từ phía nhà cung cấp. Tuy nhiên, người sử dụng khó có thể nắm bắt được các thành phần của hệ thống, các công nghệ cũng như các chức năng được cài đặt trong hệ thống. Do vậy, người dùng cũng rất khó kiểm soát được các mã độc cũng như mã cửa hậu được cài đặt trong hệ thống.

Hệ thống VPN có thể được triển khai theo các giao thức tại các tầng khác nhau trong mô hình mạng phân tầng. Theo mô hình TCP/IP, các giao thức xây dựng VPN phổ biến bao gồm: SST/TLS ở tầng ứng dụng, IPSec ở tầng liên mạng, L2TP và PPTP ở tầng truy cập mạng ([1, 2]).

Hệ thống VPN triển khai theo IPSec được sử dụng phổ biến vì khả năng bảo mật tốt hơn SSL/TLS và có tốc độ xử lý cao hơn L2TP và PPTP. Mỗi hệ thống VPN sử dụng IPSec gồm hai thành phần chính là thành phần quản lý, phân phối khóa và thành phần triển khai IPSec.

Thành phần triển khai IPSec có hai giao thức là AH (Authentication Header) và ESP (Encapsulating Security Payload). Thành phần quản lý và phân phối khóa gồm các giao thức như: ISAKMP ([3]), Oakley ([4]), KLIPS ([5]), Photuris ([6]) và SKEME ([7]). Mỗi hệ thống VPN hỗ trợ một số giao thức, thuật toán băm như SHA1, SHA2, MD5 và thuật toán mật mã như 3DES, AES....

Đối với các thuật toán mật mã đã được biên dịch và đóng gói trong các công cụ VPN thương mại, người sử dụng sẽ gặp khó khăn khi muốn phân tích các thành phần của hệ thống cũng như can thiệp vào hệ thống để thay thế các hệ mật đặc thù. Mặc dù các hệ mật được tích hợp sẵn như 3DES, AES,... đang được coi là có tính an toàn cao và được nhiều tổ chức tin dùng, nhưng trong nhiều hệ thống truyền tin với dữ liệu nhạy cảm, cần có độ bảo mật cao, thì nhu cầu sử dụng một hệ mật riêng, đặc thù là rất cần thiết. Để có thể làm chủ được hệ thống, công nghệ, và các thuật toán mật mã trong công cụ xây dựng VPN, việc sử dụng mã nguồn mở là một cách tiếp cận phù hợp và có tính thực tiễn cao.

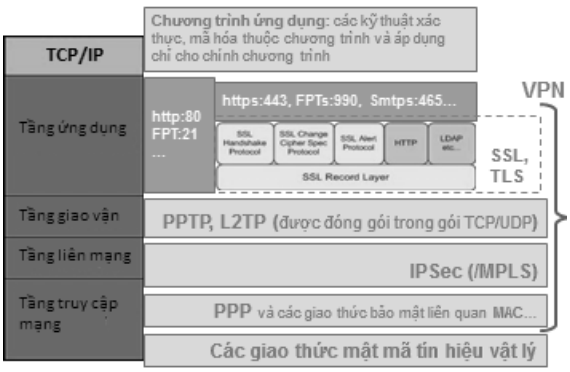
Hiện nay, công cụ xây dựng VPN mã nguồn mở OpenSwan đang được nghiên cứu, sử dụng rộng rãi. Đặc biệt, OpenSwan đang được hỗ trợ và được cài đặt mặc định trong một số hệ thống mã nguồn xây dựng điện toán đám mây phổ biến hiện nay (Ví dụ như đám mây OpenStack). Mã nguồn OpenSwan được viết bằng ngôn ngữ C, sử dụng trình biên dịch *gcc*, chạy trên môi trường Linux. OpenSwan có thành phần triển khai IPSec Netkey là giao thức IPSec nguyên thủy của nhân Linux từ phiên bản 2.6. Netkey sử dụng thư viện mật mã trong nhân Linux thông qua giao diện Crypto API để triển khai IPSec. Do đó, trong bài báo này, chúng tôi tập trung phân tích việc tích hợp, thay thế các thuật toán mật mã trong nhân Linux khi triển khai OpenSwan.

Nội dung phần còn lại của bài báo được trình bày như sau. Sau Mục giới thiệu, Mục II tổng hợp các kiến thức cơ bản về VPN và triển khai VPN dựa trên OpenSwan. Mục III trình bày mô hình đề xuất về tích hợp thuật toán mật mã mới trong OpenSwan. Mục IV tóm lược và đánh giá các kết quả thực nghiệm. Cuối cùng, Mục IV là kết luận và đưa ra một số định hướng nghiên cứu.

II. HỆ THỐNG VPN DỰA TRÊN OPENSWAN

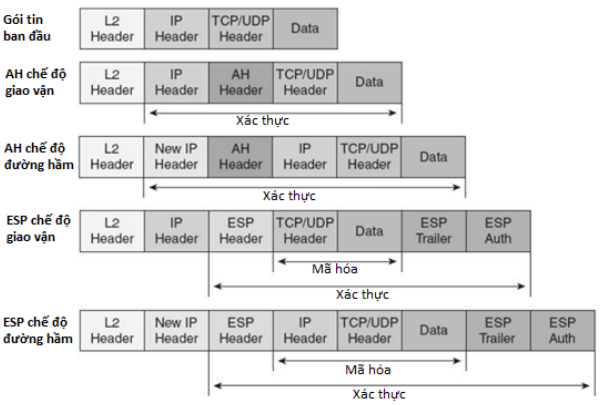
A. *Mô hình VPN dựa trên OpenSwan*

VPN sử dụng mạng công cộng như Internet, ATM/Frame Relay... làm cơ sở hạ tầng để truyền thông tin, kết hợp với các chính sách truy cập, giao thức và thuật toán mật mã để đảm bảo hệ thống hoạt động như một mạng riêng và kiểm soát được truy cập. Trong mô hình TCP/IP, có thể triển khai VPN theo các tầng khác nhau như chỉ ra trong Hình 1.



Hình 1. Các dạng VPN theo mô hình TCP/IP

Như mô tả trong Hình 1, mặc dù có thể triển khai VPN theo các giao thức ở các tầng khác nhau trong mô hình TCP/IP, nhưng việc triển khai VPN theo giao thức IPSec đang được sử dụng phổ biến. Giao thức IPSec là sự kết hợp của các chuẩn được định nghĩa trong RFC 2406, cho phép chứng thực, kiểm tra tính toàn vẹn dữ liệu, điều khiển truy cập và đảm bảo bí mật thông tin thông qua hai giao thức AH và ESP, như minh họa trong Hình 2.

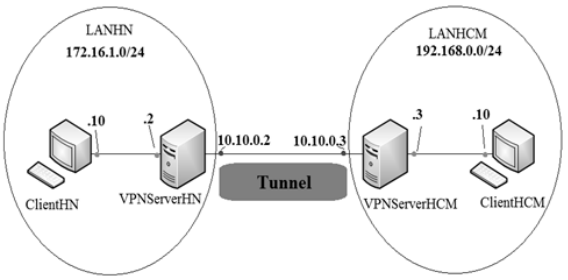


Hình 2. Tổng hợp các giao thức và các chế độ hoạt động trong IPSec

- Giao thức xác thực tiêu đề AH: đảm bảo tính toàn vẹn cho tiêu đề gói tin và dữ liệu.
- Giao thức đóng gói tải tin an toàn ESP: thực hiện mã hóa và đảm bảo tính toàn vẹn cho gói dữ liệu nhưng không bảo vệ tiêu đề cho gói IP như giao thức AH.

Giao thức IPsec sử dụng thành phần trao đổi khóa IKE (Internet Key Exchange) để thỏa thuận một tổ hợp an toàn SA (Security Association) giữa hai thực thể và trao đổi các thông tin về khóa. Hiện nay, giao thức IKE đang được sử dụng trong hầu hết các ứng dụng thực tế để đảm bảo truyền tải thông tin an toàn trên diện rộng.

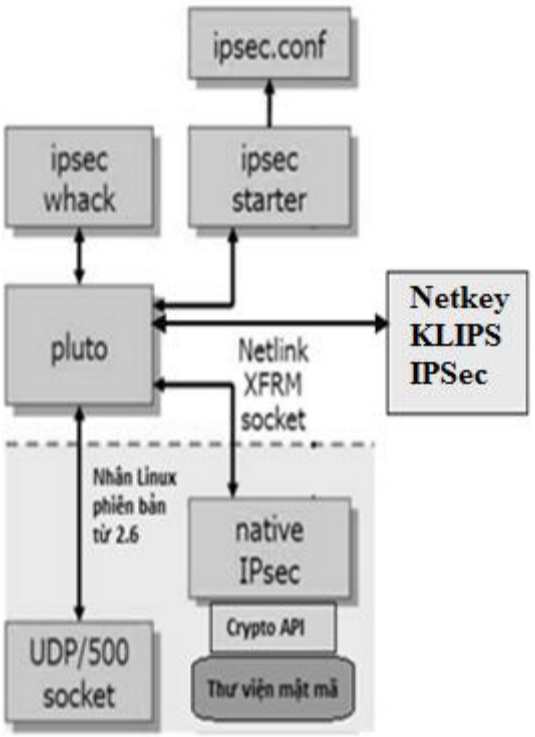
OpenSwan là bộ phần mềm VPN mã nguồn mở, cài đặt giao thức IPSec trong các hệ điều hành họ Linux (Redhat, Ubuntu...). Có thể triển khai VPN từ gói cài đặt hoặc từ mã nguồn OpenSwan theo mô hình truy cập từ xa (remote) hoặc mô hình kết nối các mạng (site to site) như minh họa trong Hình 3. Theo đó, các máy tính giữa hai mạng LANHN và LANHCM sẽ kết nối với nhau thông qua đường hầm (tunnel) nối giữa hai máy chủ VPNServerHN và VPNServerHCM.



Hình 3. VPN kết nối các mạng dựa trên OpenSwan

B. Kiến trúc mã nguồn và nguyên lý hoạt động

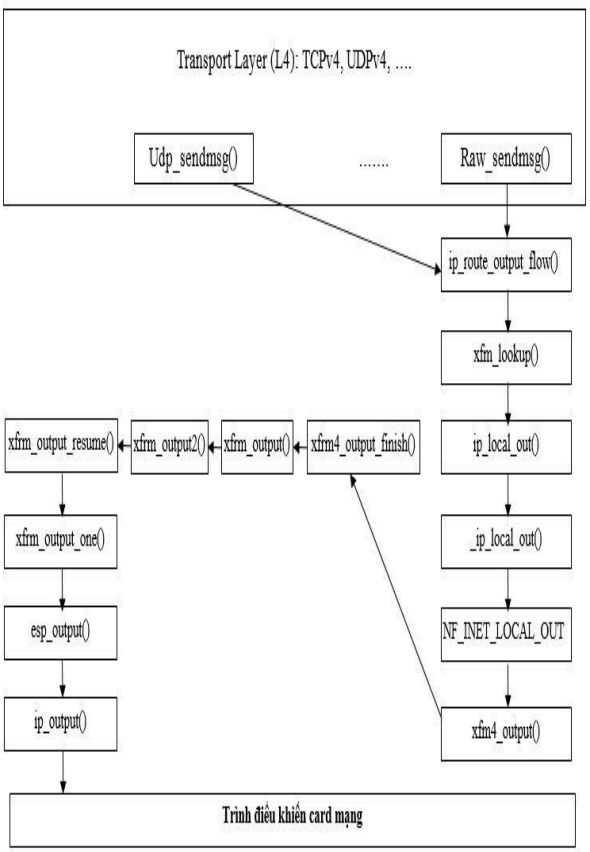
OpenSwan cũng như các công cụ xây dựng VPN theo IPSec nói chung, gồm hai thành phần chính là hệ thống phân phối khóa và hệ thống IPSec. Cấu trúc mã nguồn của OpenSwan được mô tả như trong Hình 4.



Hình 4. Cấu trúc mã nguồn OpenSwan

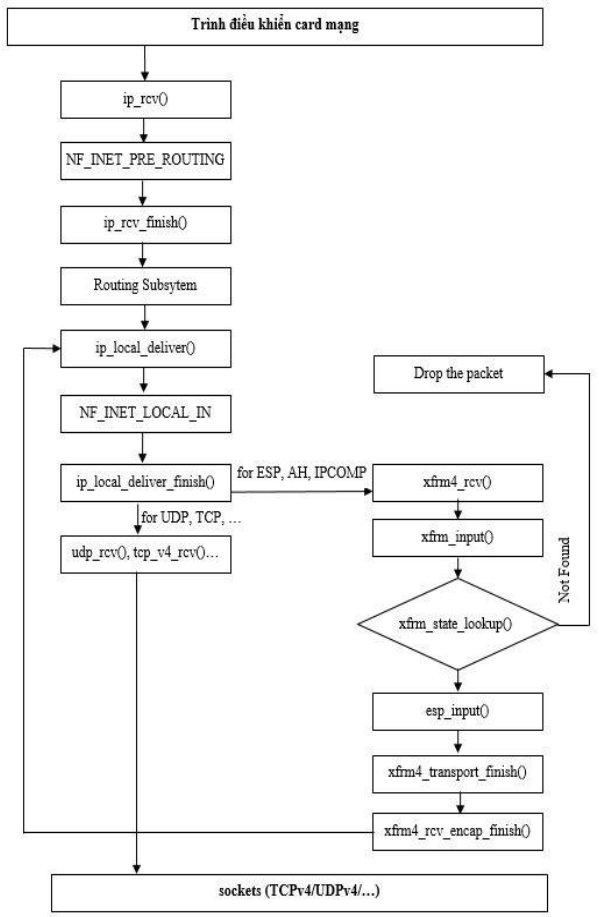
Trong đó, Pluto là thành phần phân phối khóa, KLIPS (có thể được thay thế bằng Netkey) là thành phần IPSec. Người triển khai có thể lựa chọn Netkey hoặc KLIPS khi triển khai VPN. Netkey là thành phần IPSec ra đời sau KLIPS, hướng đến sử dụng cho cả IPv4 và IPv6. Netkey được tích hợp trong nhân Linux từ phiên bản 2.6, sử dụng thư viện mật mã của nhân thông qua giao diện *Crypto API* ([8-12]).

Quá trình phân phối khóa của Pluto có thể thực hiện theo giao thức IKE hoặc sử dụng khóa chia sẻ trước. Sau giai đoạn phân phối khóa, Netkey sẽ thực hiện quá trình truyền thông có bảo mật theo IPSec. Quá trình phân tích gói, mã hóa và gửi gói tin theo giao thức ESP trong mã nguồn Netkey được thực hiện theo lược đồ gọi hàm như trong Hình 5, theo đó, hàm *esp_output()* sẽ gửi dữ liệu đã mã hóa cho hàm *ip_output()* trước khi đưa tới trình điều khiển card mạng.



Hình 5. Lược đồ gọi hàm của IPSec trong nhân Linux khi gửi tin

Quá trình nhận gói tin, phân tích và giải mã gói tin theo giao thức ESP được thực hiện theo lược đồ gọi hàm như trong Hình 6 ([3, 4, 6, 7]), dữ liệu đã mã hóa được giải mã tại hàm *esp_input()*.



Hình 6. Lược đồ gọi hàm của Netkey trong nhân Linux khi nhận gói tin

C. *Thư viện mật mã trong OpenSwan*

Thư viện mật mã trong nhân Linux được cài đặt theo API mật mã mức nhân để cung cấp các hàm băm, thuật toán mã hóa, giải mã, sinh số ngẫu nhiên,... API mật mã mức nhân coi mỗi thuật toán như một "quá trình chuyển đổi". Chương trình cài đặt của quá trình chuyển đổi là mã nguồn thực tế của thuật toán hoặc giao diện đến phần cứng khi thực hiện cứng hóa thuật toán. Cấu trúc chứa các đối tượng chuyển đổi gọi là một quy trình điều khiển hệ mật. Theo đó, quá trình sử dụng hệ mật trong nhân Linux được tiến hành như sau:

- Khởi tạo quy trình điều khiển hệ mật;
- Thực thi các thao tác mật mã thông qua các API trong điều khiển hệ mật;
- Hủy bỏ quy trình điều khiển hệ mật.

API mức nhân cung cấp giao diện cài đặt của các thuật toán mật mã; mỗi thuật toán mật mã có thể được gọi theo các khuôn mẫu khác nhau. Mỗi khuôn mẫu chỉ ra cách thức thực thi một thuật toán mật mã. Ví dụ, thuật toán mã hóa AES và hàm băm SHA1 có thể được thực hiện thông qua các khuôn mẫu như *ecb(AES)*, *cmac(AES)*, *rfc4106(GCM(AES))*, *hmac(SHA1)*, *authenc (HMAC(SHA 1))* và *cbc(AES)*. Theo đó, các khuôn mẫu sử dụng thuật toán được tổng hợp như sau ([13-16]):

<khuôn mẫu>(<thuật toán đơn>)

<khuôn mẫu 1>(<khuôn mẫu 2>(<thuật toán đơn>))

Mỗi lược đồ mật mã AEAD (Authenticated Encryption with Associated Data) cài đặt bằng ngôn ngữ C theo khuôn mẫu *gcm(aes)* trong nhân Linux bao gồm các tệp tin: *gcm.c*, *aes-generic.c*, *ctr.c*, *ghash-generic.c*, *seqiv.c*. Cấu trúc API mật mã trong nhân Linux được sử dụng để cài đặt các hệ mật gián tiếp, theo các tầng khác nhau. Tầng IPSec kích hoạt quá trình thực hiện các thao tác mật mã như lược đồ gọi hàm trong Hình 7, theo các bước sau:

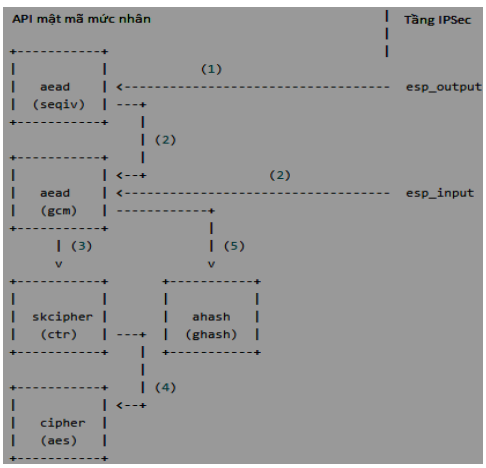
1) *esp_input()* gọi *crypto_aead_encrypt()* để kích hoạt thao tác mật mã. Cài đặt SEQIV được đăng ký như một hệ mật GIVCIPHER trong *crypt_rfc4106_alloc()*.

2) SEQIV triệu gọi lược đồ mật mã AEAD thông qua API với điều khiển hệ mật GCM.

3) Cài đặt kiểu GCM AEAD gọi thực thi hệ mật khóa đối xứng qua SKCIPHER API.

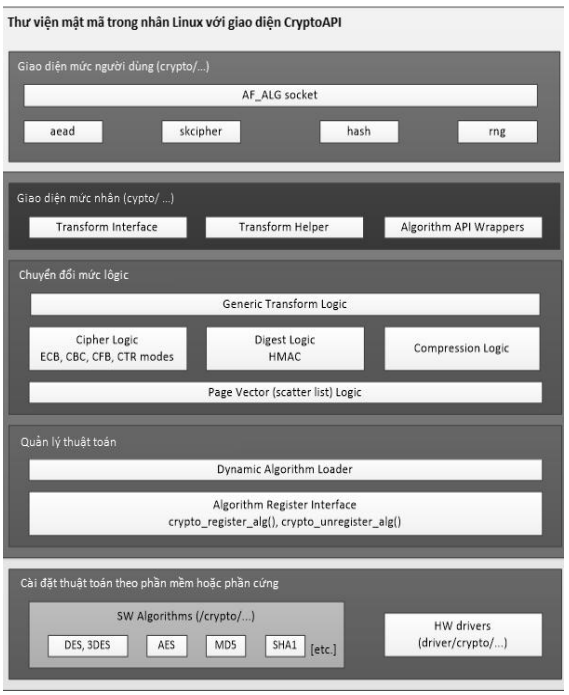
4) Hệ mật khóa đối xứng với khuôn mẫu CTR(AES) thực thi thuật toán mật mã qua điều khiển hệ mật để mã hóa khối.

5) Cài đặt GCM AEAD thực thi thuật toán băm thông qua giao diện AHASH.



Hình 7. Lược đồ AEAD trong nhân Linux

Kiến trúc thư viện mật mã trong nhân Linux được mô tả như trong Hình 8.

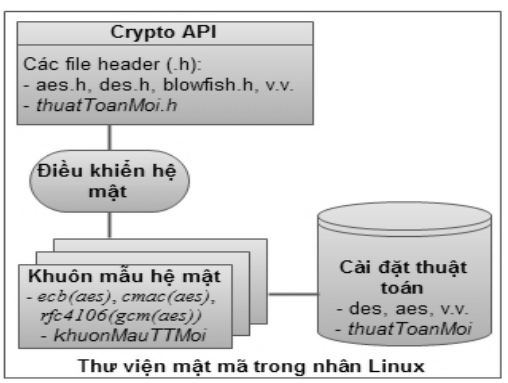


Hình 8. Kiến trúc thư viện mật mã trong nhân Linux

III. TÍCH HỢP THUẬT TOÁN MẬT MÃ MỚI TRONG OPENSWAN

A. Mô hình tích hợp thường được sử dụng

Việc can thiệp và làm chủ hệ thống VPN dựa trên OpenSwan, cụ thể như việc tích hợp hoặc thay thế các thuật toán mật mã hiện tại bằng một thuật toán mật mã riêng của người sử dụng là nhu cầu cần thiết đối với nhiều tổ chức trên thế giới. Phương pháp tích hợp thường được sử dụng hiện nay được giới thiệu trong Hình 9.



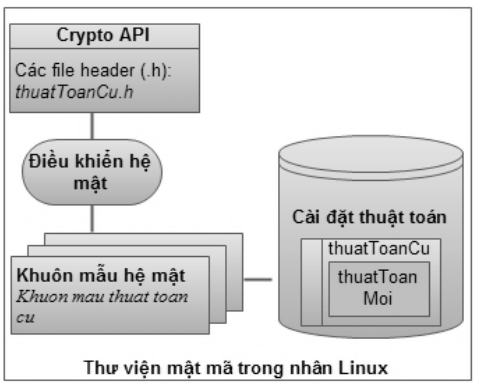
Hình 9. Mô hình tích hợp thuật toán mật mã mới

Trong mô hình này, để cài đặt và bổ sung một thuật toán mật mã mới vào thư viện mật mã trong nhân Linux, cần thực hiện các công việc như: định nghĩa giao diện gọi hàm mức người dùng, giao diện gọi hàm mức nhân thông qua các tệp tin *.h*, các hàm chuyển đổi mức logic, các hàm đăng ký thuật toán và cài đặt cụ thể của thuật toán. Đây là phương pháp thường được sử dụng để tích hợp mô-đun mật mã mới vào nhân Linux. Tuy nhiên, phương pháp này tồn tại một số hạn chế sau: Làm thay đổi kiến trúc hệ thống; Làm tăng độ phức tạp khi sử dụng với lược đồ mật mã AEAD; Khó tích hợp với các chương trình ứng dụng; Phải biên dịch lại cả nhân Linux và chương trình ứng dụng. Đặc biệt, khi kết hợp các hệ thống, với các chế độ làm việc khác nhau, việc tích hợp theo cách này sẽ phải chỉnh sửa mã nguồn, biên dịch lại toàn bộ các hệ thống liên kết với nhân. Ví dụ, khi cài đặt hệ điều hành đám mây OpenStack, kết hợp với phần mềm VPN OpenSwan, và IPSec trong nhân Linux, cần phải chỉnh sửa mã nguồn OpenStack, OpenSwan, IPSec và biên dịch, tích hợp lại cả ba hệ thống này.

B. Mô hình tích hợp đề xuất

Trên cơ sở phân tích mô hình hoạt động của OpenSwan với Netkey và kiến trúc của thư viện mật mã trong phần trước, cũng như theo phân tích về phương pháp tích hợp hiện đang được sử dụng ở trên, chúng tôi đề xuất mô hình tích hợp thuật toán mật mã mới như mô tả trong Hình 10.

Trong mô hình này, kiến trúc thư viện mật mã được giữ nguyên, thuật toán mới được thay thế cho nội dung của một thuật toán đã có; hệ thống API vẫn dùng như thuật toán gốc. Ví dụ, để tích hợp thuật toán blowfish, chúng tôi giữ nguyên kiến trúc API của thuật toán AES và cài đặt nội dung của AES theo blowfish. Theo đó, khi thực thi mã hóa, giao diện sử dụng là thuật toán AES nhưng nội dung cài đặt là thuật toán blowfish.



Hình 10. Mô hình tích hợp thuật toán mật mã mới bên trong thuật toán đã có

Theo phương pháp tích hợp theo mô hình trong Hình 10, toàn bộ kiến trúc hệ thống không thay đổi, vấn đề tích hợp chỉ thực hiện cục bộ trong hàm cài đặt thuật toán cũ nên độ phức tạp

thấp. Đồng thời, khi tích hợp thuật toán mới, không cần chỉnh sửa mã nguồn, biên dịch lại các hệ thống, chương trình đang sử dụng thuật toán cũ mà chỉ cần dịch lại nhân Linux hoặc dịch lại mô-đun mật mã và cài đặt vào nhân. Đây là một ưu điểm nổi bật của phương pháp này khi liên kết thuật toán mật mã mới với một phần mềm mã nguồn đóng; khi đó không thể chỉnh sửa mã nguồn phần mềm nên không thể tích hợp được theo phương pháp trước. Tích hợp theo phương pháp này là cách tiếp cận đơn giản, hiệu quả và ít rủi ro. Tính đúng đắn của mô hình tích hợp được chúng tôi chứng minh cụ thể trong *Phần C*.

C. *Tính đúng đắn, hiệu quả của mô hình đề xuất*

Tính đúng đắn của mô hình đề xuất

Tính đúng đắn của giải pháp đã được chứng minh thông qua các luận điểm sau: Toàn bộ kiến trúc của hệ thống không thay đổi; Các thông tin cấu hình không thay đổi; Các nguyên mẫu hàm và hệ thống API không thay đổi. Ví dụ, thay nội dung của thuật toán 3DES bằng thuật toán GOST (một thuật toán mật mã của Nga không được cài đặt mặc định trong OpenSwan) tương đương với việc đặt tên mới cho hàm cài đặt thuật toán GOST. Đồng thời, tính đúng đắn cũng được chúng tôi kiểm chứng bằng thực nghiệm, thông qua 3 bước thực hiện giải pháp và kiểm tra kết quả dưới đây.

Bước 1. Triển khai hệ thống VPN OpenSwan với thuật toán mật mã 3DES trong hệ thống đám mây OpenStack; Thực hiện ghi thống kê ra bộ nhớ đệm trong nhân Linux; Thực hiện trao đổi dữ liệu giữa hai site của VPN; Thực hiện đọc bộ đệm trong nhân để kiểm tra bản mã, bản rõ của gói tin; Thực hiện kiểm tra bản mã, bản rõ bằng một chương trình bên ngoài để khẳng định thuật toán 3DES đã mã dữ liệu và ghi trong bộ đệm.

Bước 2. Giữ nguyên toàn bộ cấu hình hệ thống mạng; Thay nội dung của hàm cài đặt thuật toán 3DES bằng cài đặt của thuật toán GOST.

Bước 3. Biên dịch nhân; Triển khai hệ thống Openswan trên nhân mới; Triển khai hệ thống VPN; Thực hiện trao đổi thông tin giữa hai site của hai đầu VPN; Thực hiện ghi bản mã, bản rõ ra bộ đệm trong nhân; Thực hiện kiểm tra bản mã, bản rõ với khóa tương ứng bằng chương trình bên ngoài chạy thuật toán mật mã GOST để khẳng định thuật toán GOST đã mã dữ liệu lưu trong bộ đệm.

Tính hiệu quả của mô hình đề xuất

Đây là giải pháp đơn giản và là cách tiếp cận nhanh, ít rủi ro nhất vì các lý do: Không làm thay đổi kiến trúc hệ thống; Hệ thống cũ dựa trên 3DES đã vận hành tốt nên thay GOST trong 3DES cũng vận hành tốt; Hệ thống API không thay đổi; Kích thước khối mã của 3DES và GOST giống nhau, đều bằng 8 byte nên không phải xử lý đầu vào.

Chứng minh hiệu năng:

Gọi t_1 là thời gian thực hiện tiến trình người dùng trên OpenStack, t_2 là thời gian thực hiện các hàm trong OpenSwan, t_3 là thời gian thực hiện các hàm trong nhân Linux, t_4 là thời gian thực hiện GOST, t_{12} là thời gian chuyển từ tiến trình của OpenStack sang tiến trình OpenSwan, t_{23} là thời gian chuyển tiến trình của Openswan cho tiến trình mã hóa của Netkey trong nhân Linux, và t_{34} là thời gian thực thi các tiến trình trung gian bao gồm gói thuật toán mật mã thực như *cgm, aead, ecb, cbc*.... Tổng thời gian T được tính từ khi các máy ảo trao đổi thông tin trong OpenStack đến khi thực hiện xong hàm mã hóa lõi được đánh giá theo công thức (1).

$$T = t_1 + t_2 + t_3 + t_4 + t_{12} + t_{23} + t_{34} \quad (1)$$

Với thuật toán 3DES ban đầu, tổng thời gian được đánh giá theo công thức (2). Với thuật toán GOST cài đặt trong vỏ 3DES, tổng thời gian được đánh giá theo công thức (3).

$$T_{3DES} = t_1 + t_2 + t_3 + t_{4\text{-}3DES} + t_{12} + t_{23} + t_{34} \quad (2)$$

$$T_{GOST} = t_1 + t_2 + t_3 + t_{4\text{-}GOST} + t_{12} + t_{23} + t_{34} \quad (3)$$

Do kiến trúc hệ thống không thay đổi, luồng dữ liệu không thay đổi, hệ thống các lời gọi hàm thông qua các API trong không gian người dùng và các lời gọi hệ thống trong không gian nhân không thay đổi, nên công thức (2) và (3) chỉ khác nhau ở $t_{4\text{-}3DES}$ và $t_{4\text{-}GOST}$. Nghĩa là thời gian thực thi trong giải pháp tích hợp thuật toán GOST trong thuật toán 3DES so với hệ thống ban đầu chỉ phụ thuộc vào thời gian thực thi của thuật toán GOST.

D. *Quy trình tích hợp theo mô hình đề xuất*

Quy trình tích hợp một thuật toán mật mã mới vào hệ thống mã nguồn mở OpenSwan và nhân Linux để triển khai hệ thống VPN theo mô hình đề xuất như sau:

- Phân tích thuật toán mật mã cần tích hợp vào hệ thống: Tùy theo yêu cầu của người sử dụng mà thuật toán mật mã cần tích hợp có các tham số về dữ liệu rõ đầu vào,

dữ liệu mã đầu ra, cũng như khóa mã khác nhau. Việc tích hợp sẽ thuận lợi hơn nếu thuật toán mật mã cần sử dụng có các tham số phù hợp với chuẩn tham số của một số thuật toán có sẵn, ví dụ như trùng với các tham số của hệ mật AES, 3DES....

- Lựa chọn thuật toán đã có trong hệ thống để thay thế thuật toán mới: Từ kết quả phân tích thuật toán cần tích hợp ở bước trên, bước này sẽ xác định một thuật toán có các tham số tương đồng nhất trong hệ thống để việc tích hợp là thuận lợi nhất.
- Tương thích vào / ra: Viết lại các hàm vào / ra để thuật toán mật mã mới tương thích với API của thuật toán gốc. Ví dụ, thuật toán mới có đầu vào là khối 64 bit, khi thay vào thuật toán AES 128 bit chỉ cần chuyển 128 bit thành các khối đầu vào của thuật toán mới, hoặc nếu thuật toán mới sử dụng khóa mã 256 bit và khối 64 bit, khi thay vào thuật toán 3DES cần hiệu chỉnh lại tham số khóa thành 256 bit.
- Tích hợp và kiểm tra: Lập trình tích hợp thuật toán mật mã mới, biên dịch lại nhân Linux; biên dịch lại mã nguồn OpenSwan. Kiểm tra hệ thống đã làm việc với thuật toán mật mã mới thông qua việc kiểm soát luồng dữ liệu và đặt các log để ghi thông tin ra bộ đệm trong nhân Linux.

IV. ĐÁNH GIÁ KẾT QUẢ THỰC NGHIỆM

Trên cơ sở mô hình và quy trình thay thuật toán mật mã mới như đề xuất ở trên, chúng tôi đã thực nghiệm thay đổi một số thuật toán khác nhau để kiểm tra tính đúng đắn cũng như hiệu suất của hệ thống.

A. Mô tả thực nghiệm

Chúng tôi đã tiến hành tích hợp thuật toán mật mã mới là thuật toán GOST của Nga vào trong hệ thống OpenSwan theo giải pháp đề xuất như mô tả trong Hình 10. Mã nguồn nhân Linux có tích hợp thuật toán mật mã mới và mã nguồn OpenSwan được biên dịch và triển khai trên máy chủ Ubuntu, phiên bản nhân 3.16.37. Thực nghiệm được triển khai theo mô hình VPN giữa hai mạng như trong Hình 3, trong đó máy chủ cài Ubuntu phiên bản 14.04 và các máy trạm cài hệ điều hành Windows 8, Ubuntu Desktop.

B. Triển khai thực nghiệm

Các bước chính để triển khai hệ thống VPN với thuật toán mật mã mới theo giải pháp trong Hình 10 như sau. Đầu tiên, chúng tôi tiến hành tải nhân Linux phiên bản 3.16.37 và tiến hành tích hợp thuật toán mật mã GOST vào trong thuật toán 3DES; sau đó biên dịch nhân, cài đặt nhân theo các câu lệnh trong Bảng 1.

BẢNG 1. CÁC LỆNH THAY NHÂN

STT	Câu lệnh	Mô tả
1	apt-get update apt-get install build-essential	Cài đặt các gói cần thiết để dịch nhân
2	apt-get install libncurses5-dev make menuconfig	Cài đặt gói cấu hình nhân trong giao diện văn bản. Khi cấu hình nhân, cần lựa chọn NETfrm để sử dụng trong IPSec.
3	Make	Biên dịch nhân
4	make modules	Biên dịch các mô-đun phụ thuộc
5	make modules_install	Cài đặt các mô-đun phụ thuộc
6	make install	Cài đặt nhân

Sau khi tích hợp thuật toán mật mã mới, biên dịch và thay thế nhân, chúng tôi tiến hành biên dịch và cài đặt OpenSwan phiên bản 2.6.38 từ mã nguồn theo các lệnh sau:

```
make programs
make install
```

Tiếp theo, thực hiện các lệnh Linux với người dùng *root* để cấu hình hệ thống OpenSwan như sau:

```
for f in /proc/sys/net/ipv4/conf/*/send_redirects; do
echo 0 > $f; done
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 0 > $f; done
echo 0 > /proc/sys/net/ipv4/ip_forward
```

C. Đánh giá kết quả thực nghiệm

Sau khi triển khai thực nghiệm như trong phần trước, chúng tôi đã tích hợp thuật toán mật mã mới vào vỏ API của thuật toán hiện tại, biên dịch và thay thế nhân *Linux 3.16.37* cho nhân *3.13.0-32*, biên dịch mã nguồn *OpenSwan 2.6.38* với nhân mới và triển khai hệ thống VPN kết nối giữa hai mạng. Hệ thống VPN hoạt động thông suốt trên cả hai giao thức AH và ESP khi sử dụng phần mềm *Wireshark* để bắt và phân tích gói tin.

Về hiệu năng và mức tiêu tốn dung lượng bộ nhớ khi thực hiện tích hợp một thuật toán mới theo giải pháp thông dụng trong Hình 9 và giải pháp đề xuất trong Hình 10 là ngang nhau. Do cả hai giải pháp này đều có độ sâu đường gọi hàm như nhau, nên quá trình cấp phát, thu hồi bộ nhớ

cho các thành phần cục bộ có kích thước và thời gian tương ứng trên đường gọi hàm.

Để đánh giá thời gian thực thi của thuật toán GOST so với 3DES trong giải pháp đề xuất và hệ thống ban đầu, chúng tôi đã viết các đoạn chương trình ghi lại thời gian lập mã và giải mã khi thực hiện thuật toán 3DES và khi thực hiện thuật toán GOST ban đầu. Sau đó, thực hiện VPN với cả hai hệ thống để ghi thông tin thống kê ra bộ đệm trong nhân. Sau khi thực hiện xong, chúng tôi tiến hành đọc bộ đệm trong nhân để so sánh thời gian thực thi GOST và 3DES ban đầu trong trường hợp thực hiện lệnh ping với kích thước gói tin thông thường và ping với kích thước gói tin tối đa. Thời gian được tính đến nanô giây. Theo kết quả tổng hợp từ thông tin thống kê trong bộ đệm nhân, tốc độ thực hiện của GOST lớn hơn hai lần so với thuật toán 3DES, điều này là phù hợp với thực tế, do thuật toán 3DES có số vòng lớn hơn.

V. KẾT LUẬN

Bài báo này đã phân tích nguyên lý hoạt động và mã nguồn OpenSwan để triển khai hệ thống VPN theo IPSec. Trong OpenSwan, thành phần Netkey sử dụng triển khai IPSec hướng đến sử dụng cho cả IPv4 và IPv6, được tích hợp trong nhân Linux từ phiên bản 2.6 trở lên. Thành phần Netkey sử dụng thư viện mật mã trong nhân thông qua giao diện Crypto API. Trên cơ sở này, chúng tôi đã đề xuất mô hình tích hợp thuật toán mật mã mới trong OpenSwan. Mô hình đề xuất tích hợp thuật toán mới vào trong vỏ một thuật toán đã có trong nhân mà không làm thay đổi kiến trúc thư viện mật mã trong nhân, nên không ảnh hưởng đến các hệ thống hiện tại. Quá trình triển khai đơn giản hơn nhiều so với mô hình đang được sử dụng, đồng thời cũng làm tăng khả năng che giấu thuật toán thực tế được sử dụng. Kết quả thực nghiệm đã cho thấy tính đúng đắn và tính hiệu quả của giải pháp đề xuất. Thuật toán mới, cần tích hợp vào hệ thống, có thể là một thuật toán đã được công bố, nhưng chưa có trong OpenSwan, hoặc là một thuật toán hoàn toàn mới, do một tổ chức tự xây dựng để sử dụng riêng cho mô hình của họ.

Từ những kết quả bước đầu khả quan, chúng tôi sẽ tiếp tục nghiên cứu, thực nghiệm để tối ưu giải pháp tích hợp mật mã mới cho OpenSwan và tối giản mã nguồn OpenSwan để biên dịch cho các thiết bị di động, các hệ thống nhúng. Đồng thời, chúng tôi cũng hướng đến nghiên cứu về mật mã hạng nhẹ để tích hợp vào OpenSwan biên dịch cho các hệ thống nhúng.

TÀI LIỆU THAM KHẢO

[1]. "Architecture", Journal of Advances in Computer Networks, Vol. 2, No. 4, 2014.

[2]. M. Fahandezh, M. Bondy, and S. Erfani, "A framework for implementing IPSec functional architecture", in Proc. Canadian Conference on Electrical and Computer Engineering, pp. 71-76.

[3]. D. Maughan, B. Patrick, and M. Schertler, "Internet Security Association and Key Management Protocol (ISAKMP)", Internet-Draft, IPSEC WG, 1996 (File: draft-ietf-ipsec-isakmp-06.txt)

[4]. HK. Orman, "The OAKLEY Key Determination Protocol", Internet-Draft, IPSEC WG, 1996 (File: draft-ietf-ipsec-oakley-01.txt).

[5]. A. Aziz, T. Markson, and H. Prafullchandra, "Simple Key-Management For Internet Protocols (SKIP)", Internet-Draft, IPSEC WG, 1996 (File: draft-ietf-ipsec-skip-07.txt).

[6]. P. Karn and WA. Simpson, "The Photuris Session Key Management Protocol", Internet-Draft, IPSEC WG, 1996 (File: draft-ietf-ipsec-photuris-11.txt).

[7]. H. Krawczyk, "SKEME: A Versatile Secure Key Exchange Mechanism for Internet" in Proceedings of SNDSS '96, IEEE (1996) pp.114-127.

[8]. Robert Love, "Linux Kernel Development", Addison-Wesley Professional, 3rd Edition, pp. 440, 2010 (0672329468).

[9]. Daniel P. Bovet, "Understanding the Linux Kernel", O'Reilly Media, 3rd Edition, pp. 944, 2005.

[10]. Wolfgang Mauerer, "Professional Linux Kernel Architecture", Wrox, 1st Edition, pp. 1368, 2008 (ISBN-13: 978-0470343432).

[11]. James Morris, "The Linux Kernel Cryptographic API", Linux Journal, 2003.

[12]. Abdulellah A. Alsaheel and Ahmad S. Almogren, "A Powerful IPSec Multi-Tunnels" .

[13]. Jean-Luc Cooke and David Bryson, "Strong Cryptography in the Linux Kernel", Discussion of the past, present, and future of strong cryptography in the Linux kernel, 2003.

[14]. Lei Gong, "A new framework of cryptography virtio driver", Huawei Technologies Co., LTD., pp. 1-5, 2005.

[15]. Rami Rosen, "Linux Kernel Networking: Implementation and Theory",A press, p. 636, 2013.

[16]. Stephan Mueller and Marek Vasut, "*Linux Kernel Crypto API*", Technical report of The kernel development community, pp. 30, 2016.

SƠ LƯỢC VỀ TÁC GIẢ

ThS. Nguyễn Như Tuấn

Đơn vị công tác: Tạp chí An toàn thông tin, Ban Cơ yếu Chính phủ.

Email: nguyennhutuan@bcy.gov.vn

Quá trình đào tạo: Nhận bằng Kỹ sư và Thạc sĩ chuyên ngành Kỹ thuật mật mã tại Học viện Kỹ thuật mật mã năm 2000 và 2007. Hiện đang làm Nghiên cứu sinh khoá I 2014 -2018 tại Học viện Kỹ thuật mật mã.

Hướng nghiên cứu hiện nay: Kỹ thuật học máy và khai phá dữ liệu ứng dụng trong an toàn thông tin; An toàn và bảo mật trong điện toán đám mây; Bảo mật dữ liệu tầng vật lý trong mạng truyền tin không dây.

TS. Phạm Văn Hưởng

Đơn vị công tác: Học viện Kỹ thuật mật mã, Ban Cơ yếu Chính phủ.

Email: huongpv@gmail.com

Quá trình đào tạo: Nhận bằng Cử nhân công nghệ thông tin năm 2005, bằng thạc sĩ năm 2008 và bảo vệ Tiến sĩ năm 2015 tại Đại học Công nghệ - Đại học Quốc gia Hà Nội.

Hướng nghiên cứu hiện nay: An toàn và bảo mật trong điện toán đám mây; IoT; Tối ưu trong phát triển phần mềm nhúng; Ứng dụng học máy; Khai phá dữ liệu trong an toàn thông tin.

ThS. Phạm Quốc Hoàng

Đơn vị công tác: Học viện Kỹ thuật mật mã, Ban Cơ yếu Chính phủ.

Email: hoang2268@gmail.com

Quá trình đào tạo: Nhận bằng Kỹ sư An toàn thông tin năm 2008, tại FSO liên bang Nga. Nhận bằng Thạc sỹ Kỹ thuật mật mã tại Học viện Kỹ thuật mật mã năm 2014.

Hướng nghiên cứu hiện nay: An toàn và bảo mật trong điện toán đám mây; Thiết kế mã khối; Hàm băm; :ý thuyết an toàn chứng minh được và tiêu chuẩn tham số trong mật mã khóa công khai.