ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

ĐINH THỊ THÚY

NGHIÊN CỨU VÀ PHÁT TRIỂN ỨNG DỤNG JAVACARD

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

Hà Nội - 2017

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

ĐINH THỊ THÚY

NGHIÊN CỨU VÀ PHÁT TRIỂN ỨNG DỤNG JAVACARD

Ngành: Công nghệ thông tin
Chuyên ngành: Quản lý Hệ thống thông tin
Mã số: Chuyên ngành đào tạo thí điểm

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. PHÙNG VĂN ỔN
TS. LÊ PHÊ ĐÔ

Hà Nội - 2017

LỜI CẢM ƠN

Lời đầu tiên tôi xin gửi lời cảm ơn sâu sắc nhất đến thầy TS. Lê Phê Đô và thầy TS. Phùng Văn Ổn, đã tận tâm, tận lực hướng dẫn, định hướng cho tôi, đồng thời, cũng đã cung cấp nhiều tài liệu và tạo điều kiện thuận lợi trong suốt quá trình học tập và nghiên cứu để tôi có thể hoàn thành luận văn này.

Tôi xin chân thành cảm ơn đến các thầy, cô trong Bộ môn Quản lý hệ thống thông tin và Khoa Công nghệ thông tin, Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội cùng với ban lãnh đạo nhà trường đã nhiệt tình giảng dạy và truyền đạt những kiến thức, kinh nghiệm qúy giá trong suốt quá trình học tập rèn luyện tại trường.

Tôi xin gửi lời cảm ơn đến các bạn học viên lớp K22-QLHTTT, nhóm bảo mật UET đã đồng hành cùng tôi trong suốt quá trình học tập. Cảm ơn gia đình, bạn bè đã quan tâm và động viên giúp tôi có nghị lực phấn đấu để hoàn thành tốt luận văn này.

Do kiến thức và thời gian có hạn nên luận văn sẽ không tránh khỏi những thiếu sót nhất định. Tôi rất mong nhận được những sự góp ý quý báu của thầy cô, đồng nghiệp và bạn bè.

Một lần nữa xin gửi lời cảm ơn chân thành và sâu sắc.

Hà Nội, tháng 07 năm 2017

Học viên thực hiện

Đinh Thị Thúy

LỜI CAM ĐOAN

Tôi xin cam các kết quả đạt được trong luận văn **"Nghiên cứu và phát triển ứng dụng JavaCard"** do tôi thực hiện dưới sự hướng dẫn của TS. Lê Phê Đô và TS.Phùng Văn Ổn.

Trong toàn bộ nội dung nghiên cứu của luận văn, các vấn đề được trình bày đều là những tìm hiểu và nghiên cứu của cá nhân tôi hoặc là trích dẫn các nguồn tài liệu và một số trang web đều được đưa ra ở phần Tài liệu tham khảo.

Tôi xin cam đoan những lời trên là sự thật và chịu mọi trách nhiệm trước thầy cô và hội đồng bảo vệ luận văn thạc sĩ.

Hà Nội, tháng 07 năm 2017

Đinh Thị Thúy

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

DANH SÁCH CÁC TỪ VIẾT TẮT

TT	TỪ VIẾT TẮT	TIẾNG ANH	THUẬT NGỮ MẬT MÃ
1	DHP	Diffie-Helman Problem	Bài toán Diffie-Hellman
2	DLP	Discrete Logarithm Problem	Bài toán logarithm rời rạc trên trường hữu hạn
3	EC	Elliptic Curve	Đường cong elliptic
4	ECC	Elliptic Curve Cryptosystem	Hệ mật Elliptic
5	ECDLP	Elliptic Curver Discrete Logarithm Problem	Bài toán logarithm rời rạc trên đường cong elliptic
6	MOV	Menezes-Okamoto-Vanstone attack	Tấn công MOV
7	SSL	Secure Sockets Layer	Là tiêu chuẩn của công nghệ bảo mật, truyền thông mã hoá giữa máy chủ Web server và trình duyệt (browser).
8	SSLHP	SSL Handshake protocol	Giao thức bắt tay
9	SSLRP	SSL Record Layer protocol	Giao thức lớp ghi
10	JC	Java Card	JavaCard
11	PVC	Polyvinyl chloride	Loại nhựa cứng và không có mùi
12	ABS	Acrylonnitrile, Butadiene, Styrene	Một loại nhựa nhiệt rất dẻo dai, chịu được sự va đập mạnh.
13	GSM	Global System for Mobile Communication	Hệ thống thông tin di động toàn cầu thế hệ thứ hai (2G)
14	JCWDE	Java Card Workstation Development Environment	Môi trường thát triển máy trạm JavaCard.
15	EMV	Europay, MasterCard, Visa	EMV là chuẩn thẻ thanh toán thông minh.

MỞ ĐẦU

1. Tính cấp thiết của đề tài luận văn

Ngày nay, sự hội nhập kinh tế sâu rộng đã mang đến cho người tiêu dùng Việt Nam cơ hội tiếp cận với những xu hướng hiện đại của thế giới. Con người dần chuyển sang sử dụng các dịch vụ thông minh hơn, tiện lợi hơn để đáp ứng các nhu cầu cuộc sống một cách hiện đại, tối ưu. Giờ đây người tiêu dùng có thể dễ dàng mua sắm, thanh toán các dịch vụ sinh hoạt, giao thông, y tế mà không cần phải mất thời gian và công sức tới các điểm giao dịch như trước thay vào đó là việc sử dụng một thiết bị đơn giản nhỏ gọn là thẻ thông minh. Sự phát triển nhanh chóng của công nghệ bán dẫn cho phép các nhà sản xuất chip tạo ra những con chip hay thẻ thông minh ngày càng nhỏ gọn cùng với sức mạnh tính toán cao. Tuy nhiên việc có quá nhiều nhà sản xuất chip, công việc phát triển ứng dụng cho thẻ thông minh gặp khó khăn về sự tương thích. Do đó nhu cầu về một nền tảng chung bên trong chip được đặt ra, công nghệ Java Card được phát triển để phục vụ mục đích này. Với việc tạo ra một môi trường ảo chung trên tất cả các hệ điều hành hỗ trợ JavaCard, công nghệ này đã giúp cho việc phát triển ứng dụng chip trở nên dễ dàng giúp tiết kiệm thời gian nghiên cứu phát triển.

Hình thức mua sắm trực tuyến đang ngày càng phổ biến và người tiêu dùng sẽ dễ dàng chọn lựa, sở hữu những món hàng yêu thích hay săn tìm các chương trình giảm giá, khuyến mãi hấp dẫn khi sở hữu thẻ tín dụng. Thẻ tín dụng là phương tiện thanh toán phù hợp với lối sống hiện đại. Tuy nhiên, quy trình đăng ký thẻ tín dụng mất khá nhiều thời gian, người tiêu dùng sau khi chuẩn bị giấy tờ, tới chi nhánh ngân hàng để đăng ký, thời gian đăng ký hạn chế trong giờ hành chính gây bất tiện cho người đăng ký thẻ mới. Ngoài ra thời gian chờ đợi thẻ cũng mất từ năm đến bẩy ngày và phải lên đúng chi nhánh nơi mình đã đăng ký để nhận thẻ.

Đi đôi với việc phổ dụng các giao dịch thông qua mạng Internet dẫn đến nguy cơ mất an toàn thông tin khi sử dụng thẻ tín dụng. Do đó, vấn đề đặt ra là làm thế nào đảm bảo an toàn thông tin trong giao dịch trực tuyến và đăng ký thẻ. Chúng ta cần có các giải pháp đảm bảo an toàn thông tin sử dụng được xây dựng dựa trên lý thuyết mật mã, an toàn bảo mật thông tin. Các nhà khoa học đã phát minh ra những hệ mật mã như RSA, Elgamal,…nhằm che dấu thông tin cũng như là làm rõ chúng để tránh sự nhòm ngó của những kẻ cố tình phá hoại. Mặc dù rất an toàn nhưng có độ dài khoá lớn nên trong một số lĩnh vực không thể ứng dụng được. Chính vì vậy hệ mật trên đường cong elliptic ra đời. Đây là hệ mật được đánh giá là hệ mật có độ bảo mật an toàn cao và hiệu quả hơn nhiều so với hệ mật công khai khác.

Ở phạm vi đề tài này tôi đặt ra vấn đề nghiên cứu ứng dụng hệ mật trên đường cong Elliptic vào bảo mật thẻ thông minh nhằm đảm bảo an toàn thông tin trong việc đăng ký thẻ trực tuyến cũng như giao dịch trực tuyến trên Internet.

Mục đích nghiên cứu:

Luận văn đề cập đến công việc thực tiễn hiện nay là việc phát triển ứng dụng cho các loại thẻ thông minh hỗ trợ công nghệ Java Card. Phần lý thuyết trình bày các kiến thức liên quan về thẻ thông minh, công nghệ Java Card, cung cấp nền tảng cơ sở cho lập trình viên trước khi xây dựng ứng dụng hay thiết kế hệ thống sử dụng công nghệ Java Card. Phần thực nghiệm sử dụng cơ sở lý thuyết ở trên để cải tiến quy trình đăng ký thẻ tín dụng bằng cách áp dụng chữ ký số trên hệ mật đường cong Elliptic vào việc đăng ký thẻ tín dụng trực tuyến nhằm bảo đảm an toàn thông tin trong thẻ tín dụng.

2. Nội dung của đề tài, các vấn đề cần giải quyết:

a. Hướng nghiên cứu:

- Công nghệ Java card
- Ứng dụng mật mã đường cong Elliptic trong bảo mật thẻ thông minh.

b. Ngoài phần mở đầu, kết luận, nội dung luận văn gồm những chương sau:

Chương 1: Tổng quan thẻ thông minh.

Chương 2: Công nghệ Java Card.

Chương 3: Mật mã đường cong Elliptic.

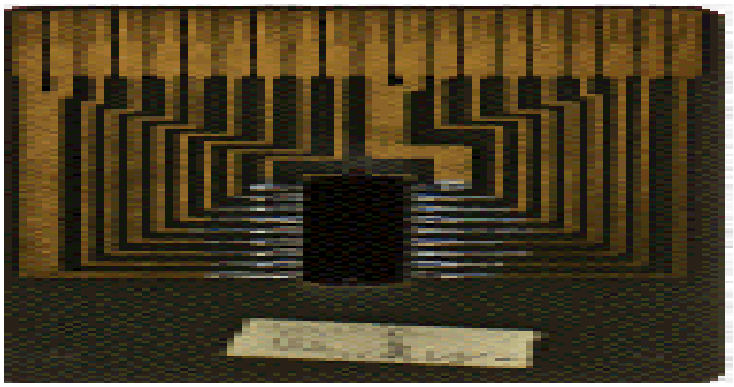
Chương 4: Ứng dụng hệ mật đường cong elliptic trong bảo mật thẻ thông minh.

CHƯƠNG 1 TỔNG QUAN THẺ THÔNG MINH

Thẻ thông minh đang được ứng dụng rộng rãi tại Việt Nam trong nhiều lĩnh vực như viễn thông, ngân hàng, thương mại điện tử, điều khiển tự động, kiểm soát người và phương tiện… Các ứng dụng của thẻ thông minh rất thiết thực và tích hợp phần mềm điều khiển bởi ưu điểm vượt trội về khả năng lưu trữ, xử lý thông tin và bảo mật dữ liệu. Chương I trình bày cái nhìn tổng quan về thẻ thông minh.

1.1 Lịch sử phát triển thẻ thông minh

Có hai ý tưởng chính đã dẫn đến sự phát triển của thẻ thông minh. Ý tưởng đầu tiên xuất hiện bởi Tiến sĩ Kunitaka Arimura đến từ Nhật Bản, ông có thiết kế tích hợp dữ liệu lưu trữ và logic số học vào một miếng silicon, ông đã nộp bản quyền cho ý tưởng vào năm 1970. Ý tưởng thứ hai là kỹ sư người Đức Helmut Gröttrup và đồng nghiệp là Jürgen Dethloff, họ đã nộp bản quyền năm 1968[6]. Bằng sáng chế thẻ chip tự động này được công bố vào cuối năm 1982[6]. Năm 1974, Roland Moreno – một nhà phát minh của Pháp, đã gắn chip lên một tấm nhựa và cấp bằng sáng chế về thẻ nhớ và thiết bị đọc nó, được đặt tên là thẻ thông minh. Moreno đã thành lập công ty Innovatron để bán ý tưởng, Moreno được biết như là cha đẻ của mạch vi xử lý (Microchip).[6]



Hình 1.1 Chip tự động

Năm 1977, ba nhà sản xuất thương mại, Bull CP8, SGS Thomson và Schlumberger đã bắt đầu phát triển các sản phẩm của thẻ thông minh[6]. Năm 1978, Bull đăng ký bằng sáng chế về bộ vi xử lý một chip tự lập trình được (SPOM-self Programmable One-chip Microcomputer)[6]. Tháng 3 năm 1979, Michel Ugon của tập đoàn Bull là người đầu tiên thiết kế bộ vi xử lý hoạt động, được biết đến như là CP8 của Bull. Nó chứa bộ nhớ lập trình 1KB, bộ vi xử lý 6805 do Motorola sản xuất. Đây có thể coi là thẻ thông minh đầu tiên kết hợp sức mạnh của bộ vi xử lý và bộ nhớ có khả năng đưa ra quyết định dựa trên nhu cầu của người dùng để sửa đổi, thêm, truy xuất hoặc xóa dữ liệu được lưu trữ. Thẻ này là một thiết kế hai chip, trong đó bộ nhớ và bộ vi xử lý là hai đơn vị riêng biệt, được chứng minh là một giải pháp không an toàn. Năm 1980 cho phép tích hợp tất cả các mạch vào trong một con chip.

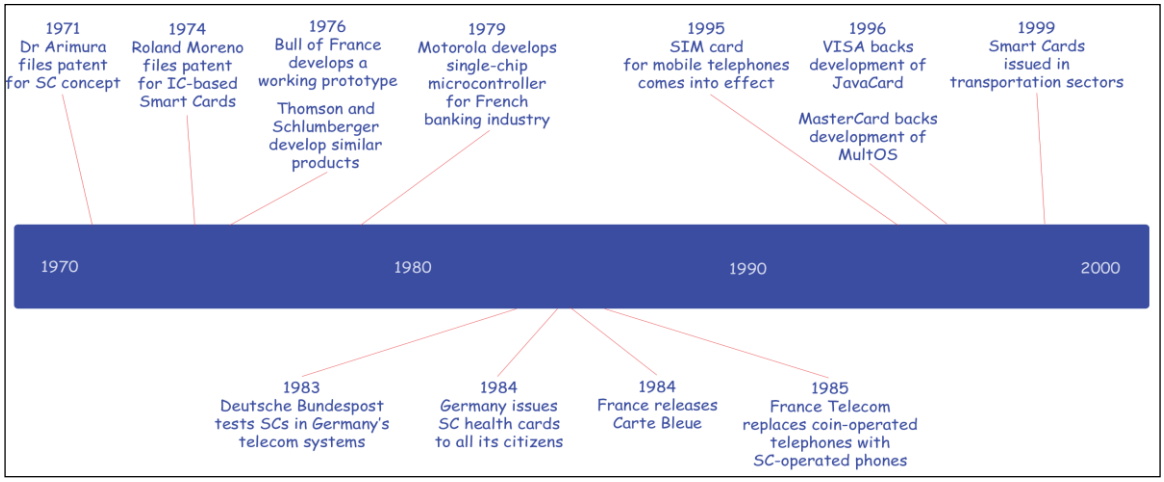
Hình 1.2 Thẻ CP8

Đến năm 1981 những chiếc thẻ thông minh đã có nền tảng ở nhiều nước Tây Âu, một số các ngân hàng Châu Âu đồng ý thành lập một cơ quan quản lý mới cho phát triển thẻ, ứng dụng và tiêu chuẩn. Tổ chức này bao gồm các tổ chức tài chính từ Bỉ, Anh, Đan Mạch, Áo, Hà Lan, và liên minh cũ các ngân hàng Pháp.

Sự bùng nổ thẻ thông minh bắt đầu vào trong thập niên 90, khi có sự xuất hiện của SIM dùng trong thiết bị điện thoại di động GSM ở châu Âu, cùng với việc mạng di động mở rộng khắp châu Âu, thẻ thông minh ngày càng trở nên thông dụng. Năm 1983 cộng đồng viễn thông yêu cầu một hệ thống điện thoại trả tiền tốt hơn do tỷ lệ gian lận và hack gia tăng đối với điện thoại công cộng sử dụng đồng xu. Hệ thống thanh toán thẻ điện thoại trả tiền thông minh đã phát triển. Schlumberger đã bắt đầu cài đặt hàng ngàn chiếc điện thoại trả tiền bằng thẻ thông minh trên khắp lục địa. Đến cuối năm họ đã cài đặt được 160.000 chiếc điện thoại. Vào năm 1984, một trong những triển khai lớn nhất của thẻ thông minh đã diễn ra tại Pháp. Ngành ngân hàng Pháp đã quyết định làm cho thẻ thông minh trở thành tiêu chuẩn cho thẻ tín dụng và thẻ ghi nợ[6].

Năm 1993, ba liên minh thẻ lớn nhất thế giới là Europay, MasterCard, Visa (EMV) thỏa thuận cùng hợp tác để xây dựng nên kỹ thuật cho việc dùng thẻ thông minh trong các thẻ thanh toán ở hai loại là thẻ tài khoản và thẻ tín dụng. Phiên bản đầu tiên của hệ thống EMV này được công bố vào năm 1994. Năm 1998, phiên bản tin cậy hơn là EMVco, chịu trách nhiệm bảo trì lâu dài hệ thống này. Mục đích của công ty EMVco là đảm bảo với các tổ chức tài chính và các đại lý rằng các chuẩn kỹ thuật dù phát triển nhưng vẫn phải giữ được tương thích với phiên bản 1998.

Hiện nay, thẻ thông minh đã và đang sử dụng cho rất nhiều lĩnh vực như làm thẻ chứng minh nhân dân, hộ chiếu điện tử, thẻ thanh toán. Tại Việt Nam thẻ thông minh hiện đang sử dụng nhiều nhất là thẻ tín dụng và sim điện thoại.



Hình 1.3 Sơ đồ lịch sử phát triển thẻ thông minh

1.2 Cấu tạo và phân loại thẻ thông minh

Thẻ thông minh được ví như một máy tính di động lưu trữ chương trình dữ liệu. Ngày nay, thẻ thông minh đang được ứng dụng rộng rãi tại Việt Nam trong nhiều lĩnh vực khác nhau như giao thông công cộng, viễn thông, ngân hàng, thương mại điện tử, điều khiển tự động, điều khiển người và phương tiện, y tế giáo dục,v.v…Các ứng dụng của thẻ thông minh rất thực tế, dễ triển khai và tích hợp phần mềm điều khiển bởi ưu thế nổi bật về khả năng lưu trữ, xử lý thông tin, bảo mật dữ liệu tốt, khả năng tích hợp linh hoạt, việc sử dụng thẻ thông minh đem lại nhiều lợi ích cho người sử dụng. Để có cái nhìn tổng quan về thẻ thông minh, cần tìm hiểu cấu tạo, nguyên lý hoạt động, các loại thẻ thông minh trên thị trường, cách thức phát triển phần mềm quản lý thẻ thông minh.

Cấu tạo và phân loại thẻ thông minh:

Thẻ thông minh thông thường có kích thước cỡ một thẻ tín dụng và được làm bằng nhựa, thường là PVC (Polyvinyl chloride – là thẻ nhựa cứng và không có mùi) đôi khi là ABS (Acrylonnitrile, Butadiene, Styrene – là một thẻ nhựa chịu được sự va đập mạnh), thẻ có thể chứa một ảnh 3 chiều tránh lừa đảo. Kích thước theo chuẩn ID-1(ISO/IEC/7810) quy định là 85,60x53,98 mm hoặc chuẩn ID-000 kích thước 25x15 mm, có bề dày mặt thẻ là 0,76 mm. Thẻ được gắn một bộ mạch tích hợp cho phép giao tiếp với hệ thống, tính toán mã hóa đảm bảo tính toàn vẹn dữ liệu. Dung lượng bộ nhớ trên thẻ thông minh khá lớn từ 64KB - 128KB (tương đương 65.536 đến 131.072 ký tự) Dữ liệu trên thẻ được truyền đến hệ thống quản trị trung tâm nhờ vào các thiết bị đọc thẻ, chẳng hạn máy đọc thẻ, ATM v.v.[3]

Phân loại thẻ thông minh: Có hai cách phân loại thẻ thông minh dựa trên công nghệ chip hay phương thức đọc dữ liệu.

Phân loại dựa trên công nghệ chip: Theo công nghệ chip được chia làm hai loại là thẻ chip nhớ (memory chip) và thẻ chip vi xử lý (microprocessor chip) được gắn trên bề mặt thẻ.[5]

- *Thẻ chip nhớ* bao gồm hai thành phần chính là thẻ nhớ cho phép có thể truy cập,

giao thức truyền thông. Ưu điểm của thẻ này là dễ sản xuất, dễ sử dụng, nhược điểm là hạn chế về bộ nhớ và tính bảo mật không cao.

- *Thẻ vi xử lý (microprocessor chip)* được cấu tạo bởi ba loại bộ nhớ, một bộ vi xử lý (CPU – Central Processing Unit), một bộ đồng xử lý mã hóa (Cryto coprocessor) và một giao diện thông tin (communication interface). Chức năng của CPU là điều khiển các bộ phận khác, xử lý thông tin và thực hiện các phép tính. Cấu tạo của CPU rất đa dạng, nhưng nói chung gồm 1 bộ xử lý (control unit) đảm nhận những chu trình cơ bản của CPU như đọc một chỉ thị và thực hiện nó, giải mã (decoding), lưu trữ (stocking), đảm nhận chức năng ALU (arithmetic and logic unit), quản lý thanh ghi, quản lý bộ nhớ (registers, RAM, ROM). ROM dùng để lưu trữ mã máy (code), dữ liệu (data), và chỉ có thể đọc, chứ không thể thay đổi nội dung. Thông tin trong ROM vẫn nguyên vẹn, ngay cả khi chúng ta ngắt card. Trong ngành thẻ thông minh, ROM được dùng để lưu trữ những ứng dụng sẽ được thực hiện bởi bộ vi xử lý. Dung lượng của ROM vào khoảng 256KB là tối đa, do thiếu không gian lắp đặt. EEPROM (Electrically Erasable Programmable Read Only Memory) giống ROM ở chỗ là thông tin lưu trữ vẫn nguyên vẹn, ngay cả khi card bị ngắt khỏi nguồn năng lượng. EPPROM có thêm 1 lợi thế là có thể cùng lúc ở mode đọc hoặc ghi. Giống ROM, dung lượng EPPROM vào khoảng vài trăm KB, do thiếu không gian. Ngày nay, sự xuất hiện của những công nghệ mới như bộ nhớ Flash, hoặc RAM sắt điện (FeRAM) với thời gian đọc, ghi, xóa ngắn hơn nhiều và kích thước của bit nhớ cũng nhỏ hơn điều đó đã tăng dung lượng nhớ của thẻ thông minh lên rất nhiều. RAM là một loại bộ nhớ nhanh và không vĩnh cửu (sẽ bị xóa khi ngắt khỏi nguồn năng lượng). RAM chỉ được sử dụng bởi bộ vi xử lý, các yếu tố bên ngoài không thể truy cập vào RAM. RAM khá đắt, và cũng chiếm nhiều không gian, nên thường dung lượng không nhiều, khoảng vài Kb. Bộ đồng xử lý mã hóa (Crypto coprocessor) để đáp ứng nhu cầu hiệu năng, một vài loại thẻ thông minh được trang bị thêm một chip điện tử. Chip này được thiết kế đặc biệt để có thể thực hiện các phép tính số học trên những số rất lớn (vài trăm đến vài nghìn bits) một cách tối ưu. Chức năng của chip này là để thực hiện các hàm mã hoá xuất hiện trong các giao thức (protocol) của thẻ thông minh. Thời kỳ đầu, chip mã hoá chỉ được trang bị trên 1 số loại thẻ thông minh vì đắt. Nhưng hiện nay chúng ta có thể tìm thấy thành phần này trên hầu như tất cả các loại thẻ thông minh.

Phân loại dựa trên phương thức đọc dữ liệu: chia làm 3 loại là thẻ tiếp xúc, thẻ không tiếp xúc và thẻ lưỡng tính[5].

- *Thẻ tiếp xúc*: là loại thẻ có một diện tích tiếp xúc thường dễ nhận diện bởi có gắn con chip (màu vàng hoặc bạc) trên thân thẻ, tiếp điểm đó có diện tích khoảng $1cm^2$, được chia thành các phần riêng biệt gồm đầu vào, đầu ra dữ liệu, tín hiệu reset (phục hồi trạng thái ban đầu của thẻ), tín hiệu xung đồng hồ, chân điện áp. Con chip này nhỏ nhưng có các chức năng không khác gì một chiếc máy vi tính, muốn đọc, ghi thông tin thì bề mặt chip phải tiếp xúc trực tiếp với đầu đọc thẻ. Loại thẻ này được sử dụng

nhiều trong tài chính và truyền thông (ví dụ sim điện thoại) vì ưu điểm giá cả rẻ, đáp ứng nhiều tiêu chuẩn về công nghệ, độ bảo mật cao. Khi được đưa vào máy đọc, chip trên thẻ sẽ giao tiếp với các tiếp điểm điện tử cho phép đọc các thông tin từ chip và viết thông tin lên nó. Thẻ thông minh loại này không có pin, năng lượng làm việc sẽ được cấp trực tiếp từ máy đọc thẻ.

Hình 1.4 Thẻ thông minh tiếp xúc và đầu đọc thẻ.

Các chuẩn ISO/IEC 7816 và ISO/IEC 7810 qui định[6]:

- Hình dạng và kích thước vật lý.
- Vị trí và hình dạng của các tiếp điểm điện tử.
- Các đặc tính điện.
- Các giao thức thông tin, bao gồm định dạng của các lệnh gửi đến thẻ và các đáp ứng từ thẻ.
- Độ tin cậy của thẻ.
- Chức năng.

- Thẻ không tiếp xúc: Là loại thẻ mà chip trên nó liên lạc với máy đọc thẻ thông qua công nghệ sóng vô tuyến RFID (Radio Frequency Identification) với tốc độ trao đổi dữ liệu từ 106 đến 848 kbit/s, thân thẻ chứa chip và đường dây ăngten được dấu ngầm. Ăngten đi vòng quanh thẻ, nó có nhiệm vụ làm trung gian nhận hoặc phát sóng radio giữa đầu đọc thẻ và chip trên thẻ. Trong thẻ có một cuộn cảm có khả năng dò tín hiệu vô tuyến trong một dải tần nhất định, chỉnh lưu tín hiệu và dùng nó để cung cấp năng lượng hoạt động cho chip trên thẻ, khoảng cách giao tiếp giữa đầu đọc thẻ và máy khoảng 10cm. Tốc độ xử lý của thẻ không tiếp xúc nhanh hơn so với thẻ tiếp xúc, vì vậy thẻ không tiếp xúc thường được ứng dụng tại những nơi cần phải xử lý nhanh như kiểm soát phương tiện công cộng, xe bus, thẻ ra vào…thẻ không tiếp xúc đắt hơn thẻ tiếp xúc, tuy nhiên độ bảo mật thông tin không an toàn bằng thẻ tiếp xúc. Về mặt cấu tạo thì thẻ không tiếp xúc khác với thẻ tiếp xúc là con chip quản lý thông tin không nằm lộ trên thẻ mà ẩn bên trong thẻ.

Hình 1.5 Thẻ không tiếp xúc

Một số ví dụ về thẻ thông minh không tiếp xúc là thẻ Octopus của Hồng Kong, và thẻ Suica của Japan Rail mà đã xuất hiện trước khi có chuẩn ISO/IEC 14443. Các hình sau cho thấy một số thẻ thông minh dùng trong giao thông công cộng và ứng dụng thanh toán điện tử [6].

Hình 1.6 Thẻ thu phí giao thông và thẻ dùng cho việc giao thông công cộng.

- *Thẻ lưỡng tính*: là thẻ kết hợp các đặc điểm của thẻ tiếp xúc và thẻ không tiếp xúc. Dữ liệu được truyền hoặc bằng phương pháp tiếp xúc trực tiếp thẻ với đầu đọc hoặc qua tín hiệu vô tuyến. Thẻ lưỡng tính đắt hơn rất nhiều so với 2 loại thẻ trên. Đầu đọc thẻ thông minh là đầu đọc dùng cho việc giao tiếp với thẻ, dữ liệu và điện năng được truyền trực tiếp hoặc gián tiếp qua công nghệ RFID từ thẻ vào máy đọc. Đầu đọc thẻ dễ dàng tích hợp vào các hệ thống khác nhau thông qua thiết bị đầu cuối. Tùy vào công nghệ sử dụng, có 2 dạng cổng kết nối là USB và COM. Nếu sử dụng kết nối USB tốc độ truyền tín hiệu đạt 12 Mbps (High speed), điện áp cung cấp thông thường 5V, 200mA.

Các ứng dụng tiêu biểu của thẻ thông minh:

✓ Định danh: Đối với các hệ thống cần xác nhận định danh được phép truy cập hệ thống như: Mạng viễn thông di động, tài khoản ngân hàng, chứng minh nhân dân điện tử, hộ chiếu điện tử hay hệ thống quản lý truy cập (Access Control) thì TTM được đại diện cho quyền truy cập các hệ thống này.

✓ Lưu trữ: Khả năng lưu trữ an toàn trên thẻ smartcard, cho phép lưu trữ những thông tin thuộc về chủ thẻ như thông tin y tế, thông tin cá nhân, chứng chỉ điện tử (thẻ bảo hiểm y tế, giấy phép lái xe điện tử, v.v...).

✓ Xác thực Offline: Ngoài các ứng dụng phổ biến nói trên, thẻ thông minh còn được dùng để kiểm tra tính xác thực thẻ thành viên không yêu cầu kết nối hệ thống trung tâm. Thẻ SAM card (Security Application Module) là một dạng của ứng dụng này. SAM card có vai trò như một cảnh sát giao thông kiểm tra người lái xe xuất trình bằng lái tại xa lộ mà không có máy tính hay kết nối cơ sở dữ liệu trung tâm. Khi đó, thẻ SAM và thẻ của người cần kiểm tra sẽ kiểm tra lẫn nhau (xác thực chéo) để kiểm tra tính trung thực trước khi thực hiện những nghiệp vụ tiếp theo. Thẻ SAM còn có khả năng đa dạng khóa (Diversify Key) đảm bảo an toàn và bảo mật trong mỗi phiên giao dịch.

1.3 Ưu nhược điểm của thẻ thông minh

Ưu điểm của thẻ thông minh: Thẻ thông minh với cấu tạo chip có nhiều ưu điểm hơn so với các loại thẻ từ khác. Ưu điểm của thẻ thông minh là:

➢ *Thẻ thông minh được ứng dụng được trong nhiều lĩnh vực*

Thẻ thông minh ứng dụng tiện lợi trong nhiều lĩnh vực như thẻ công dân, hộ chiếu điện tử, thẻ y tế - lưu trữ thông tin cần thiết như nhóm máu, sinh trắc học của người chủ thẻ, thanh toán lương - thẻ tín dụng (ATM), sinh hoạt phí hàng tháng như thẻ đổ xăng, SIM cho điện thoại di động, thẻ truyền hình cho các kênh phải trả tiền, các thẻ dùng cho thu phí giao thông tự động, thanh toán tiền xe bus, tàu, chi phí du lịch, nhà hàng quán ăn, cửa hàng buôn bán, trung tâm chăm sóc spa, bãi đỗ xe, siêu thị... Thẻ thông minh cũng dùng như ví điện tử dùng để trả tiền tại các trạm đỗ xe và các máy bán hàng tự động. Ngoài ra còn ứng dụng trong lĩnh vực an ninh cho máy tính, hệ thống mã hóa dữ liệu trên đĩa cứng có thể dùng thẻ thông minh để giữ các khóa mã bảo mật.

➢ *Tính bảo mật cao*

Tính bảo mật là ưu điểm nổi bật nhất của thẻ thông minh bởi các thành phần vật lý của con chip đều ở dạng siêu nhỏ và chúng đều có khả năng chống lại các tấn công vật lý. Còn về khả năng tấn công hay tìm cách đọc nội dung dữ liệu lưu trong thẻ bằng phần mềm đã được hệ điều hành toàn quyền điều khiển. Hệ điều hành thẻ đều phải tuân theo các tiêu chuẩn ISO-7816 với nhiều mức bảo vệ truy cập nhiều cấp, nên rất khó để tấn công dữ liệu theo con đường này. Ngoài ra, trong thẻ còn hỗ trợ các thuật toán mã hóa, các cơ chế chống nhân bản thẻ (anti-cloning) hay an toàn trong quá trình trao đổi dữ liệu, hay những thẻ đáp ứng chuẩn EMV giúp ngăn ngừa giả mạo, thẻ EMV có một bộ vi xử lý bên trong được gắn vào con chip trên thẻ, chúng bao gồm các khóa mật mã để chứng minh thẻ là bản gốc nên nếu chủ thẻ không may đánh mất thẻ thì ngân hàng chỉ cần khóa chip là toàn bộ các giao dịch với thẻ là không thể thực hiện được. Thẻ chip có thể làm giảm đáng kể việc đánh cắp hoặc sao chép dữ liệu và thông tin của chủ thẻ so với thẻ từ.

➢ *Khả năng lưu trữ thông tin lớn*

Thẻ thông minh được ví như một máy tính cá nhân thu nhỏ bởi nó có thể lưu trữ

một lượng thông tin rất lớn về cá nhân tổ chức. Việc quản lý các thông tin này cũng rất dễ dàng vì không cần phải tích hợp thêm phần mềm nào. Mặt khác thông tin lưu trên thẻ thông minh có thể dễ dàng thay đổi, xóa hoặc thêm bớt khi cần.

➢ *Khả năng xử lý thông tin nhanh*

Với công nghệ chip điện tử thẻ thông minh xử lý thông tin rất nhanh. Chính vì thế thẻ thông minh thường được ứng dụng trong những giao dịch yêu cầu về thời gian giao dịch nhanh như thanh toán phí giao thông, thẻ gửi xe…

➢ *Có nhiều dịch vụ hỗ trợ người dùng và đơn giản hóa thủ tục*

Thẻ thông minh cho phép thực hiện các giao dịch kinh doanh một cách hiệu quả theo một cách chuẩn mực, linh hoạt và an ninh mà trong đó con người ít phải can thiệp vào. Ngoài ra thẻ thông minh với giao tiếp không cần tiếp xúc đã trở nên ngày càng phổ biến trong các ứng dụng thanh toán và mua vé, điển hình là lời giải cho bài toán bán vé vận tải công cộng. Thẻ thông minh sẽ giúp đơn giản hóa thủ tục và thời gian vận chuyển bằng các phương tiện công cộng.

➢ *Sử dụng trên phạm vi quốc tế*

Hiện nay các loại thẻ thông minh visa, master card đều có thể sử dụng trên phạm vi quốc tế, người dùng có thể thanh toán mua hàng online, phục vụ việc du lịch, học tập… người dùng có thể đi bất kỳ đâu mà không cần mang quá nhiều tiền

Hạn chế của thẻ thông minh

➢ *Dễ bị mất, dễ hư hỏng*

Giống như thẻ tín dụng thì thẻ thông minh nhỏ, nhẹ và có thể dễ dàng bị mất nếu người đó không có trách nhiệm. Không giống như thẻ tín dụng, thẻ thông minh có thể được sử dụng vào nhiều mục đích có thể là cùng một thẻ do đó việc mất thẻ thông minh có thể gây nhiều bất lợi cho người sở hữu, ví dụ như nếu mất đi một thẻ debit (thẻ ghi nợ), thẻ xe buýt, chìa khóa văn phòng, v.v… điều này gây nên sự bất tiện nghiêm trọng. Ngoài ra một nhược điểm nữa của thẻ thông minh là dễ hư hỏng. Thẻ nhựa mà chip đặt trên nó là khá dẻo, dễ uốn, và do đó chip càng lớn thì càng dễ bị gãy. Thẻ thông minh thường được bỏ trong ví đây là một môi trường khá khắc nghiệt đối với chip điện tử.

➢ *Vấn đề an toàn thẻ thông minh*

Không phải tất cả các thẻ thông minh đều an toàn. Visa và MasterCard đã phát triển một tiêu chuẩn mới mục đích đưa toàn bộ ngành công nghiệp đạt được tiêu chuẩn mã hóa. Thẻ thông minh dùng để xác nhận khách hàng là một trong những cách an ninh nhất, có thể dùng trong những ứng dụng như giao dịch ngân hàng qua internet, nhưng mức độ an toàn không thể đảm bảo 100%. Trong trường hợp giao dịch ngân hàng qua internet, nếu máy tính bị nhiễm bởi các phần mềm xấu, mô hình an ninh sẽ bị phá vỡ. Phần mềm xấu có thể được viết đè lên thông tin (cả thông tin đầu vào từ bàn

phím và thông tin đầu ra màn hình) giữa khách hàng và ngân hàng. Nó có thể sẽ sửa đổi giao dịch mà khách hàng không biết. Hiện nay có những phần mềm xấu chẳng hạn như Trojan, Silentbanker. Các ngân hàng như Fortis Dexia ở Bỉ dùng một thẻ thông minh chung với một máy đọc thẻ không nối mạng nhằm giải quyết vấn đề trên. Khách hàng nhập một thông tin đánh giá từ trang web của ngân hàng, PIN của họ, và tổng số tiền giao dịch vào một máy đọc thẻ, máy đọc thẻ sẽ trả lại một chữ ký 8 chữ số. Chữ ký này sẽ được khách hàng nhập bằng tay vào PC và được kiểm chứng bởi ngân hàng. Bên cạnh việc chạy đua kỹ thuật cũng là sự thiếu hẳn một chuẩn thống nhất về chức năng và an ninh của thẻ thông minh. Để giải quyết vấn đề này, dự án ERIDANE đã được khởi động bởi The Berlin Group để phát triển một khung chức năng và an ninh cho những thiết bị bán lẻ đầu cuối dùng thẻ thông minh

➢ *Tăng nguy cơ phạm tội*

Khi được sử dụng đúng mục đích nhận dạng nó làm cho công việc của nhân viên hành pháp và các chuyên gia chăm sóc sức khoẻ dễ dàng hơn. Tuy nhiên, đối với bọn tội phạm tìm kiếm hay đánh cắp thông tin thì chúng dựa trên số lượng thông tin mà nó có thể chứa trên thẻ, những tên tội phạm có thể lấy các thông tin bất hợp pháp trên thẻ để phục vụ các mục đích cá nhân của chúng ví dụ như các tên tội phạm khi lấy cắp được thông tin thẻ chúng có thể thực hiện giao dịch bất hợp pháp hoặc rao bán thông tin bất hợp pháp của chủ thẻ v.v…

➢ *Rủi ro về quyền riêng tư*

Dùng thẻ thông minh cho giao thông công cộng cũng có một chút rủi ro về quyền tự do cá nhân, bởi vì với hệ thống như vậy thì người quản lý giao thông có thể dò theo hành trình của cá nhân. Ở Phần Lan, bộ phận bảo vệ Dữ Liệu Ombudsman cấm người quản lý giao thông của YTV thu thập các thông tin như vậy, mặc dù trong hợp đồng với YTV người chủ thẻ có quyền yêu cầu YTV cung cấp cho họ lịch trình đi mà YTV đã tính tiền cho họ. Những thông tin về lịch trình từng được dùng trong việc truy tìm thủ phạm trong vụ đánh bom Myamanni.

➢ *Rủi ro về việc phân phối thẻ thông minh*

Vấn đề cuối cùng mà các thẻ thông minh sẽ phải đối mặt trong việc khuếch tán rộng rãi liên quan đến việc bổ sung sản phẩm. Mặc dù các thẻ thông minh tương đối rẻ, nhưng đầu đọc thẻ thì không (khoảng 50 đến 200 đô la). Tuy nhiên, trong một nỗ lực để làm cho thẻ thông minh phổ biến hơn, các công ty như Netscape và Microsoft đang đề xuất đưa phần mềm vào các gói mà họ tạo ra. Nếu được sử dụng làm thẻ thanh toán, không phải mọi cửa hàng hoặc nhà hàng sẽ có phần cứng cần thiết để sử dụng các loại thẻ này, vì công nghệ này an toàn hơn, cũng đắt hơn để sản xuất và sử dụng. Do đó, một số cửa hàng có thể tính một khoản phí tối thiểu cơ bản để sử dụng thẻ thông minh để thanh toán hơn là tiền mặt.

1.4 Thách thức trong việc phát triển ứng dụng thẻ thông minh

Phát triển ứng dụng thẻ thông minh theo truyền thống là một quá trình dài và khó. Mặc dù các thẻ được chuẩn hóa về kích thước, hình dạng, và giao thức giao tiếp, các hoạt động bên trong khác nhau giữa các nhà sản xuất. Hầu hết các công cụ phát triển thẻ thông minh được xây dựng bởi các nhà sản xuất thẻ thông minh bằng cách sử dụng các công cụ ngôn ngữ lắp ráp chung và giả lập phần cứng chuyên dụng thu được từ các nhà cung cấp chip silicon. Hầu như không thể cho các bên thứ ba phát triển các ứng dụng một cách độc lập và bán chúng cho các tổ chức phát hành. Do đó, việc phát triển các ứng dụng thẻ thông minh đã được giới hạn trong một nhóm các chuyên gia giàu kinh nghiệm và chuyên viên lập trình, những người có kiến thức sâu rộng về phần cứng và phần mềm thẻ thông minh cụ thể.

Vì không có giao diện ứng dụng tiêu chuẩn cao cấp sẵn có trong thẻ thông minh, nên các nhà phát triển ứng dụng cần phải xử lý các giao thức giao tiếp mức thấp, quản lý bộ nhớ và các chi tiết khác theo thời gian của phần cứng cụ thể của thẻ thông minh. Hầu hết các ứng dụng thẻ thông minh đang được sử dụng ngày nay đều được phát triển từ đầu, đó là một quá trình mất nhiều thời gian. Thường mất một hoặc hai năm cho một sản phẩm để đi vào thị trường. Việc nâng cấp phần mềm hoặc ứng dụng đến một nền tảng khác là đặc biệt khó khăn hoặc không thể.

Hơn nữa, các ứng dụng thẻ thông minh được phát triển để chạy trên nền tảng độc quyền, các ứng dụng từ các nhà cung cấp dịch vụ khác nhau không thể cùng tồn tại và cung cấp trên một thẻ duy nhất. Công nghệ Java Card là một giải pháp để vượt qua các trở ngại cản trở việc phát triển thẻ thông minh. Nó cho phép thẻ thông minh và thiết bị hạn chế bộ nhớ khác có thể chạy các ứng dụng (được gọi là applet) được viết bằng ngôn ngữ lập trình Java. Thông thường, công nghệ Java Card xác định nền tảng thẻ thông minh an toàn, di động và nhiều ứng dụng kết hợp nhiều lợi thế chính của ngôn ngữ Java [6].

1.5 Các hình thức tấn công trên thẻ thông minh

Thẻ thông minh có khả năng bảo mật cao bởi các thành phần vật lý của con chip đều ở dạng siêu nhỏ và chúng đều có khả năng chống lại những tấn công vật lý. Tuy nhiên mạnh mẽ là thế các thẻ thông minh vẫn có những yếu điểm các hacker luôn tìm thấy cảm hứng từ các biến chip nhỏ bé, các hacker đã phát triển một loạt các kỹ thuật để quan sát và ngăn chặn các hoạt động của thẻ thông minh để có thể tước đoạt quyền truy cập thông tin, lấy các thông tin hữu ích cũng như chiếm đoạt thông tin đó. Dưới đây sẽ mô tả các cuộc tấn công trên thẻ thông minh, hiện nay có ba cuộc tấn công cơ bản: cuộc tấn công logic, cuộc tấn công phần cứng và cuộc tấn công kênh phụ (side - channel)[6].

Cuộc tấn công Logic: dựa vào những suy luận logic liên quan đên các thuật toán mã hóa hacker cố gắng khai thác lỗ hổng trong các lĩnh vực sau:

Triển khai phần mềm: Thẻ thông minh là một bộ vi xử lý, khi triển khai thẻ thông minh thì phải thực hiện các lệnh để chạy, chip hỗ trợ hàng ngàn lệnh bổ sung. Do đó chức năng này có thể bị lạm dụng cho việc thu thập dữ liệu không mong muốn cho mục đích sửa đổi.

Các lệnh ẩn: Các hệ thống thẻ thông minh hỗ trợ trên 65000 lệnh trên lý thuyết, mặc dù chỉ có một vài lệnh trong số đó là cần thiết cho một ứng dụng cụ thể, phần còn lại dễ bị lạm dụng cho một mục đích khác.

Định vị thông số và tràn bộ đệm: Nhập các giá trị tham số không hợp lệ, có thể không được phép hoặc vượt quá chiều dài, sẽ dẫn đến kết quả không mong muốn ví dụ như tràn bộ đệm.

Giao thức mã hoá, thiết kế và cài đặt: Các giao thức điều khiển mật mã và các hoạt động mật mã trên thẻ thông minh. Nếu giao thức không được thiết kế cẩn thận, những sai sót ẩn này có thể ảnh hưởng đến hoạt động của chip. Ví dụ, một số thẻ có các phương pháp dự phòng để nâng cao độ tin cậy trong trường hợp các vấn đề kỹ thuật, trong khi điều này là không an toàn và kẻ tấn công có thể có lợi từ việc tạo ra các chức năng giả tưởng**.**

Cuộc tấn công phần cứng: đòi hỏi các thiết bị hiện đại để có thể xâm nhập vào các vi mạch của thẻ (chip), hacker khai thác lỗi trong các lĩnh vực sau:

Tấn công xâm nhập (invasive attack): lớp bảo vệ của mạch sẽ bị phá bỏ, các kỹ thuật xử lý ảnh được sử dụng để quan sát các lớp và cấu trúc mạch, các tiếp xúc điện được kết nối và bộ nhớ của thẻ sẽ được đọc. Cách tấn công này đòi hỏi những thiết bị hiện đại, chính xác và đắt tiền thường chỉ có những phòng thì nghiệm cao cấp mới có được.

Tấn công nửa xâm nhập(semi-invasive attack): khác với loại tấn công trên thì trong loại tấn công này các mạch điện không được kết nối. Một ví dụ điển hình là việc dùng tia laser chiếu vào mạch để mạch hoạt động không bình thường. Qua việc xử lý các kết quả không bình thường đó thì có thể dò được một vài thông tin bí mật

Dung môi hóa học, chất tẩy, chất nhuộm: Các chất này là các chất khử có thể phát hiện các khối cấu hình của chip từ đó nó có thể phân tích và khai thác thông tin từ chip.

Kính hiển vi quang học và điện tử: Mặc dù kích thước của chip nhỏ hơn 1 micro, nhưng nó vẫn có thể nhìn thấy qua điện tử, ngay cả kính hiển vi quang học. Một chip được thiết kế cẩn thận vẫn có thể được phân tích để tiết lộ các phần hoạt động của nó, chạy mã và thậm chí cả giá trị trên bus dữ liệu

Trạm thăm dò một đầu dò nhỏ được đặt trên một đường dây tùy ý để tạo ra một kênh mới. Nếu bus dữ liệu có thể được định vị qua hai cách tiếp cận trên, tất cả các truyền dữ liệu có thể bị chặn, chẳng hạn như mã lập trình, dữ liệu lập trình, bao gồm các phím.

Tấn công qua kênh phụ: Một cuộc tấn công kênh phụ cố gắng khai thác một số hiện tượng vật lý để phân tích hoặc sửa đổi hành vi của thẻ thông minh, chẳng hạn như

thời gian thực hiện thao tác, năng lượng tiêu thụ điện, cường độ của điện trường vv…

Thời gian thực hiện thao tác: Thông qua việc phân tích thời gian mà thẻ thực hiện các thao tác (cộng, nhân, lũy thừa) ta có thể suy ra giá trị của một hay nhiều bit mà ta đang cần tìm. Tấn công đầu tiên sử dụng thông tin về thời gian thao tác là năm 1996 để tấn công các thuật toán RSA, DES và Difie-Hellman.

Năng lượng tiêu thụ điện: Lượng điện tiêu thụ bởi thẻ vào mỗi thời điểm phụ thuộc các giá trị trung gian của thuật toán vào thời điểm đó, do đó ta có thể dùng các tín hiệu về năng lượng tiêu thụ điện để tấn công. Công bố kết quả bởi Paul Kocher năm 1999.

Cường độ của điện trường: Do tồn tại một mối quan hệ giữa cường độ dòng điện và cường độ trường điện từ, vì vậy, nếu thông tin bí mật bị rò rỉ thông qua các tín hiệu về năng lượng tiêu thụ thì các thông tin này cũng sẽ bị rò rỉ thông quan các tín hiệu về trường điện từ.

CHƯƠNG 2 CÔNG NGHỆ JAVACARD

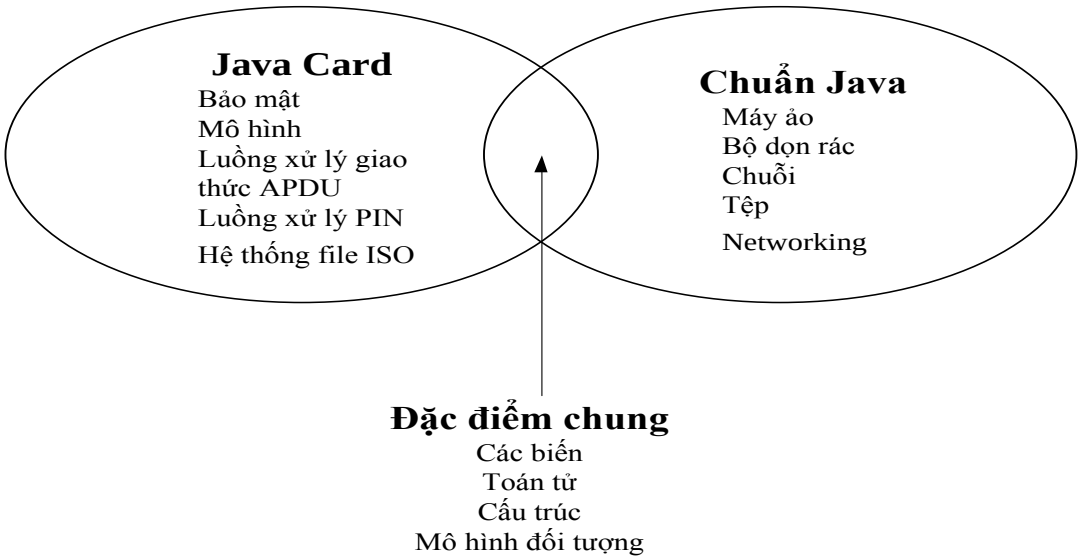
Thẻ thông minh có nhiều ưu điểm nhưng cũng có thách thức để phát triển thẻ thông minh. Các ứng dụng thẻ thông minh được phát triển để chạy trên nền tảng độc quyền, các ứng dụng từ các nhà cung cấp dịch vụ khác nhau không thể cùng tồn tại và cung cấp trên một thẻ duy nhất. Đó là trở ngại cản trở việc phát triển ứng dụng trên thẻ thông minh. Công nghệ JavaCard được hình thành nhằm giải quyết vấn đề trên. Chương 2 đi sâu vào nền tảng công nghệ JavaCard.

2.1 Giới thiệu JavaCard

JavaCard là một công nghệ cho phép mang đến cho các trình ứng dụng Java applet có thể hoạt động một cách an toàn và bảo mật trên thẻ thông minh tương tự với các bộ nhớ nhỏ của các thiết bị lưu vết. Nó là nền tảng Java nhỏ nhất hướng tới các thiết bị nhúng. JavaCard là một phần nhỏ của Java được phát triển bởi Sun, được tích hợp bên trong các thiết bị, nó đơn giản hoá việc lập trình thẻ thông minh vì các tính năng hướng đối tượng của nó. JavaCard mang đến cho người dùng khả năng lập trình cho các thiết bị mà tạo các trình ứng dụng chuyên biệt. Nó được sử dụng rộng rãi trong SIM card (trong GSM của điện thoại di động) và thẻ ATM.

Ngôn ngữ lập trình Java có nhiều ưu điểm như tính đơn giản, tính di động, mô hình bảo mật và tính hướng đối tượng. Đó cũng chính là đặc tính mà JavaCard hướng tới là khả năng tương thích và bảo mật. Tính tương thích được thể hiện ở việc JavaCard nhắm tới mục tiêu tạo ra một môi trường chuẩn cho thẻ thông minh, cho phép các ứng dụng Java có thể chạy trên các loại thẻ thông minh khác nhau. Tương tự như với môi trường Java, JavaCard thực hiện bằng việc kết hợp một máy ảo JavaCard với một bộ thư viện chung. Tuy nhiên tính tương thích bị ảnh hưởng lớn bởi sự khác nhau về kích thước bộ nhớ, khả năng xử lý và hỗ trợ của các loại thẻ khác nhau.

JavaCard có một số đặc điểm khác với Java thông thường. Về mặt ngôn ngữ lập trình, JavaCard là một phần thu gọn của Java, các cú pháp của Java, giống như tất cả các biến thể của Java và làm cho lập trình viên viết mã dễ dàng hơn vì không cần phải học một cú pháp nào khác. Tuy nhiên, cú pháp quen thuộc không có nghĩa là dễ, thật dễ dàng để bị một quan niệm sai lầm trong suy nghĩ rằng tiêu chuẩn Java và Java Card rất giống nhau, điều này có thể xảy ra do cùng cú pháp, tên tương tự và toán tử, mô hình đối tượng, nhưng trên thực tế, hai môi trường này là tương đối khác nhau từ quan điểm lập trình[6].



Hình 2.1 Các tính năng chung giữa JavaCard và chuẩn Java

JavaCard phát triển với mục đích lưu trữ các thông tin nhạy cảm, công nghệ Java Card luôn đề cao tính bảo mật và đảm bảo điều này bằng các yếu tố khác nhau như: đóng gói dữ liệu, tường lửa ngăn cách ứng dụng, mã hóa dữ liệu, tạo ứng dụng dạng Applet. Khả năng đóng gói dữ liệu cho phép dữ liệu được lưu trữ bên trong ứng dụng và các ứng dụng này được thực thi trên một máy ảo tách biệt với hệ điều hành và phần cứng của thẻ. Mỗi ứng dụng khác nhau lưu trữ trên JavaCard đều được ngăn cách bởi một tường lửa để hạn chế và kiểm tra được sự truy cập dữ liệu từ ứng dụng này sang ứng dụng khác. Khả năng mã hóa của JavaCard cho phép dữ liệu được mã hóa bằng các dạng mã hóa thông dụng sử dụng khóa như mã hóa DES, 3DES, AES hay RSA.

JavaCard là một thẻ thông minh có thể thực thi mã bytecode tương tự như Java. Do tiêu chuẩn Java quá lớn để phù hợp với thẻ thông minh, nên giải pháp cho vấn đề này là tạo ra một JavaCard với một số chuẩn Java đã được loại bỏ[6]. JavaCard được hình thành dựa trên một tập con của Java API cộng với một số lệnh thẻ đặc biệt.

Bên cạnh việc cung cấp cho các nhà phát triển môi trường phát triển quen thuộc, JavaCard còn cho phép các thẻ thông minh có thể có nhiều ứng dụng trên đó. Hầu hết các sản phẩm thẻ thông minh hiện có chỉ có một ứng dụng trên mỗi thẻ. Ứng dụng này được tự động gọi ra khi điện được cung cấp cho thẻ hoặc thẻ được cài đặt lại. Thẻ Java cho phép nhiều ứng dụng, có khả năng được viết bởi các tổ chức khác nhau, để tồn tại trên cùng một thẻ [6].

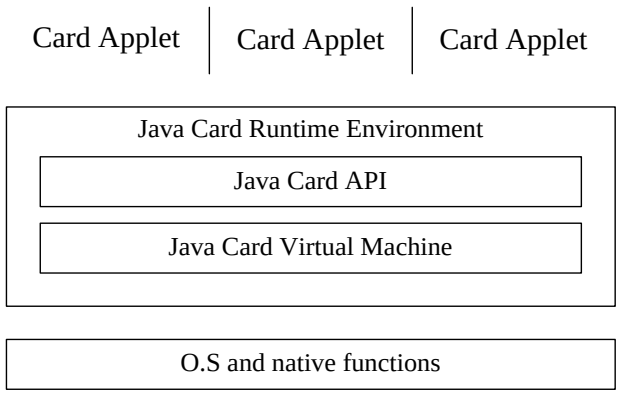
Dưới đây là những lợi ích chính của JavaCard cũng như ưu điểm của Sun:

- Đa nền tảng: JCRE (JavaCard Runtime Environment – môi trường chạy java card) phù hợp với tất cả các loại bộ xử lý thẻ thông minh.
- Tương thích: Các applet JavaCard được phát triển bởi một nhà cung cấp sẽ chạy trên bất kỳ thẻ nào phù hợp với đặc tả của JavaCard.
- Thẻ Java tuân thủ các tiêu chuẩn thẻ thông minh hiện có.
- Hỗ trợ tất cả các loại ứng dụng thẻ thông minh.

2.2 Kiến trúc JavaCard

Về mặt ngôn ngữ lập trình thì JavaCard là một phần thu gọn của Java, công nghệ JavaCard cung cấp kiến trúc để phát triển ứng dụng mở cho thẻ thông minh và nó cũng được sử dụng để phát triển các ứng dụng cho các thiết bị có bộ nhớ rất nhỏ như SIM cho điện thoại di động, thẻ ATM gắn chip.

Dưới đây là mô hình kiến trúc [7]:



Hình 2.2 Kiến trúc tổng quát của công nghệ JavaCard

Như thể hiện trong hình, máy ảo JavaCard được xây dựng trên một mạch tích hợp cụ thể. Lớp JVM (JavaCard virtual machine) giấu công nghệ độc quyền của nhà sản xuất bằng ngôn ngữ chung và giao diện hệ thống. Khung JavaCard xác định một tập các lớp API (Application Programming Interface) để phát triển các ứng dụng JavaCard và cung cấp các dịch vụ hệ thống cho các ứng dụng đó. Một ngành công nghiệp cụ thể hoặc kinh doanh có thể cung cấp các thư viện bổ sung để cung cấp dịch vụ hoặc tinh chỉnh mô hình bảo mật cho hệ thống. Các ứng dụng JavaCard được gọi là các applet. Nhiều applet có thể nằm trên một thẻ. Mỗi applet được nhận dạng duy nhất bởi AID của nó (định danh ứng dụng) như được định nghĩa trong ISO 7816[7].

Công nghệ JavaCard bao gồm các bộ thông số kỹ thuật sau:

- JavaCard API: Một giao diện lập trình ứng dụng là lớp thư viện lõi của JavaCard.
- Máy ảo JavaCard: Mô tả các đặc tính của máy ảo để xử lý các ứng dụng của JavaCard
- JCRE: (Javacard runtime environment) mô tả hành vi chi tiết về thời gian chạy,

chẳng hạn như cách bộ nhớ quản lý hay cách thực thi bảo mật.

2.3 Tập ngôn ngữ JavaCard

Do JavaCard được tích hợp vào các bộ nhớ nhỏ của thiết bị lưu vết nên nền tảng của ngôn ngữ JavaCard hỗ trợ sự chọn lựa kỹ lưỡng từ tập ngôn ngữ của Java.

Dưới đây là bảng thống kê các thuộc tính mà thư viện Java hỗ trợ [6].

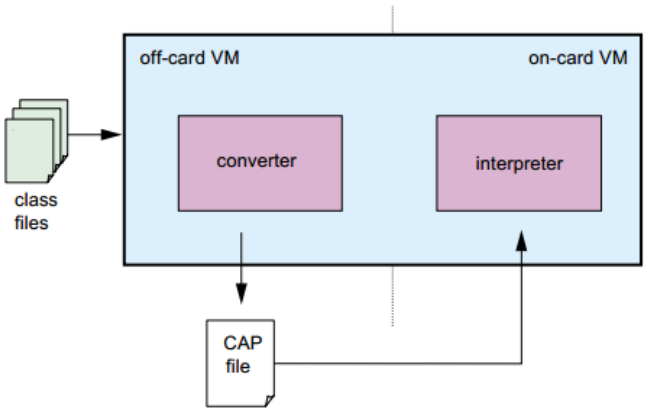
Bảng 2.1: Các thuộc tính mà thư viện hỗ trợ

Java hỗ trợ các tính năng	Java không hỗ trợ các tính năng
- Kiểu dữ liệu nhỏ: boolean, byte, short. - Mảng một chiều. - Gói java, lớp, giao diện, các ngoại lệ. - Các đặc tính hướng đối tượng trong Java: kế thừa, phương thức ảo, quá tải và tạo đối tượng động, chấp nhận phạm vi và quy tắc ràng buộc. - Các từ khóa kiểu nguyên (int) và 32-bit số nguyên kiểu dữ liệu hỗ trợ là tùy chọn.	- Kiểu dữ liệu lớn: long, double, float. - Kí tự và chuỗi. - Mảng đa chiều. - Class loading động. - Quản lý security. - Khởi tạo và hủy (bộ nhớ). - Luồng (Threads). - Object serialization (chuyển đổi trạng thái của một object). - Object cloning.

2.4 Máy ảo để chạy JavaCard

Sự khác biệt chính giữa máy ảo JavaCard (JCVM) và máy ảo Java (JVM) đó là máy ảo JavaCard thực hiện thành hai phần: một phần chạy các ứng dụng trên thiết bị đầu cuối và phần còn lại chạy trên ứng dụng thẻ[6].

JCVM chỉ hỗ trợ một tập con giới hạn của ngôn ngữ lập trình Java, nhưng nó bảo tồn nhiều tính năng quen thuộc bao gồm các đối tượng, thừa kế, các gói, tạo đối tượng động, các phương pháp ảo, các giao diện và các ngoại lệ. JCVM giảm hỗ trợ cho một số yếu tố ngôn ngữ có thể sử dụng quá nhiều bộ nhớ làm hạn chế thẻ thông minh.



Hình 2.3 Máy ảo JavaCard[6]

Máy ảo trên thẻ (on – card) interpreter (hay còn gọi là trình thông dịch JavaCard) cung cấp hỗ trợ thời gian chạy mô hình ngôn ngữ Java và cho phép độc lập phần cứng thực hiện các nhiệm vụ:

- Thực thi các đoạn mã bytecode và cuối cùng thực thi applet.
- Kiểm soát việc cấp phát bộ nhớ và tạo đối tượng.
- Đóng vai trò quan trọng trong việc đảm bảo an ninh thời gian chạy như ngăn cách tường lửa giữa các ứng dụng và hỗ trợ chia sẻ dữ liệu một cách bảo mật.

Phần máy ảo Java trên các thiết bị đầu cuối (off – card), nó chứa công cụ JCC (javacard converter) hỗ trợ việc xác thực, đóng gói, tối ưu mã hóa thường được gọi là công cụ chuyển đổi thẻ Java. Đầu ra của công cụ chuyển đổi là một tệp tin Applet được Chuyển đổi (CAP), tệp chứa tất cả các lớp trong một gói Java trong một biểu diễn nhị phân có thể chạy được. Các lớp động không được hỗ trợ trong Java Card bởi:

- Các lớp động cần phải truy cập vào những vị trí lưu trữ của tệp mô tả lớp, điều này là không khả dụng trong môi trường thẻ.
- Vấn đề an ninh (bảo mật) trong thẻ thông minh nghiêm cấm các lớp động (đối tượng ràng buộc động được cho phép).
- Thẻ thông minh hạn chế về tài nguyên, bộ nhớ.

Bộ chuyển đổi đóng gói tất cả các lớp được ứng dụng tham chiếu vào một gói ứng dụng. Lúc này bộ công cụ đóng vai trò là một bộ tiền xử lý gói ứng dụng JavaCard với các bước như sau:

- Xác thực: Kiểm tra xem gói ứng dụng đã được định dạng đúng với đầy đủ các bảng ký hiệu và kiểm tra các phạm vi ngôn ngữ lập trình có tuân thủ các đặc tả của Java Card hay không.
- Chuẩn bị: Phân bổ lưu trữ để tạo ra kiến trúc dữ liệu cho máy ảo tương ứng với các lớp, tạo các trường và phương thức tĩnh và khởi tạo các biến tĩnh thành các giá trị mặc định.
- Quyết định: Thay thế các tham chiếu ký tự tới phương thức và biến bằng tham chiếu trực tiếp khi có thể.

Thực hiện ba bước này trong bộ công cụ chuyển đổi trước khi một applet được cài lên thẻ cho phép máy ảo trên thẻ trở nên nhỏ gọn và hiệu quả hơn. Một khi ứng dụng đã được cài lên thẻ thông minh nó được coi là nạp và sẵn sàng để chạy (mặc dù một số khởi tạo có thể được yêu cầu). Sau đó, JCRE (JavaCard Run Environment) sẽ thực hiện một số thủ tục khởi tạo để tạo các tham số được khai báo với các giá trị mặc định trong giao diện. Mặc dù JCC (JavaCard converter) thực hiện càng sớm càng bắt buộc và giải quyết càng tốt, một số ràng buộc cuối cũng được hỗ trợ bởi môi trường chạy JavaCard.

Cap File: chứa một biểu diễn nhị phân thực thi các lớp trong một package Java. Một tệp CAP là một tệp JAR – là định dạng chứa tệp CAP, nó chứa một bộ thành phần được lưu trữ dưới dạng tệp cá nhân trong tệp JAR, tệp JAR chia thành các thành phần nhỏ trong mỗi thành phần nhỏ chứa nội dung của tệp CAP như thông tin về lớp, thực thi bytecode, liên kết thông tin, xác minh thông tin….Định dạng tệp CAP cho phép tối ưu hóa bằng cách sử dụng cấu trúc dữ liệu phù hợp và hạn chế gián tiếp. Nó định nghĩa một tập lệnh bytecode dựa trên tối ưu hóa từ tập lệnh Java bytecode (Mã bytecode là mã trung gian, chưa phải là mã máy)[6].

Java được tạo ra với tiêu chí "viết (code) một lần, thực thi khắp nơi" ("Write Once, Run Anywhere" (WORA)). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó. Trong công nghệ Java, tệp lớp là phần trung tâm của kiến trúc Java, nó định nghĩa tiêu chuẩn cho sự tương thích nhị phân của nền tảng Java. Do đặc tính phân bố của kiến trúc hệ thống JavaCard, tệp CAP đặt định dạng tệp chuẩn cho sự tương thích nhị phân trên nền tảng thẻ Java. Định dạng tệp CAP là hình thức trong đó phần mềm được tải lên thẻ thông minh Java. Ví dụ, các miếng CAP cho phép tải động các lớp applet sau khi thẻ đã được thực hiện. Đó là cách nó được lấy tên tệp chuyển đổi tên (CAP).

2.5 Cài đặt JavaCard và chương trình cài đặt trên thiết bị (Off-Card)

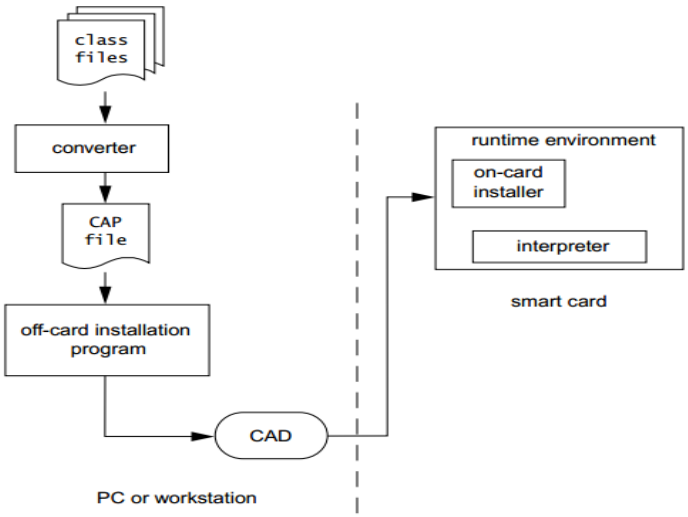
Các công cụ sau đây yêu cầu cài đặt ứng dụng JavaCard:

- Một công cụ chuyển đổi để chuyển đổi một applet JavaCard sang một định dạng cần thiết để cài đặt.
- Các công cụ xác minh ngoài thẻ để kiểm tra tính toàn vẹn của các tệp được tạo ra bởi Converter.
- Trình cài đặt thẻ gắn ngoài để cài đặt một thẻ JavaCard vào một thẻ thông minh các applet JavaCard được phát triển bằng cách sử dụng các lớp và các công cụ trên máy trạm hoặc máy tính cá nhân. Cụ thể nó cho phép nhà phát triển:
  - Biên dịch applet.
  - Theo tùy chọn, kiểm tra applet trong JCWDE (JavaCard Workstation Development Environment), và gỡ lỗi applet này. JCWDE, chạy trên máy trạm hoặc máy tính cá nhân, mô phỏng môi trường chạy JavaCard trên một máy ảo Java. Nó không phải là một mô phỏng hoàn chỉnh, ví dụ JCWDE không mô phỏng tường lửa applet của một JCVM. Tuy nhiên, JCWDE cung cấp mô phỏng cho phép kiểm tra ban đầu về một thẻ JavaCard applet. Nó cho phép chạy một applet như thể nó đã được che đậy trong bộ nhớ và đọc của một thẻ thông minh. Và quan trọng hơn, nó cho phép chạy thử nghiệm trong một máy trạm hoặc máy PC, mà không cần phải chuyển đổi các applet, tạo ra một tập tin "mask", hoặc cài đặt các applet.
  - Chuyển đổi các applet và tất cả các lớp để cài đặt vào một tập tin CAP, và có thể là một tập tin được "export". Một tập tin export được sử dụng để chuyển đổi một gói

khác nếu các gói import các class từ gói này. Không giống như JVM, xử lý một lớp cùng một lúc, đơn vị chuyển đổi của bộ chuyển đổi là một gói. Trình biên dịch Java tạo ra các tệp class từ mã nguồn, sau đó trình chuyển đổi xử lý tất cả các tệp lớp tạo thành gói Java và chuyển gói đó sang tệp CAP.

Trong công nghệ Java Card, một applet JavaCard không trực tiếp kết hợp vào một mask (mặt nạ). Tương tự như vậy, sau khi sản xuất một thẻ thông minh, một applet JavaCard không trực tiếp tải về để cài đặt vào một thẻ thông minh. Thay vào đó, đối với lớp mask (mặt nạ), một lớp applet và tất cả các lớp trong gói của nó chuyển đổi sang một tệp applet JavaCard (JCA). Tệp JCA cho bất kỳ gói nào khác được bao gồm trong mask sau đó sẽ được chuyển đổi thành định dạng tương thích với môi trường chạy đích. Đây là kết quả được chuyển đổi cho môi trường chạy đích mà được kết hợp vào mask. Cả tệp JCA và tệp CAP đều là tệp tự mô tả. Các tệp này chứa thông tin về gói được chuyển đổi [6].

Như đã đề cập trước đó, một applet JavaCard không cài đặt vào một thẻ thông minh, thay vì nó đã được cài đặt tệp CAP. Trình cài đặt thẻ ra tạo một tệp kịch bản chứa các APDU lệnh xác định phần đầu và kết thúc của tệp CAP, các thành phần và dữ liệu thành phần. Tập tin kịch bản được sử dụng làm đầu vào cho tiện ích APDUTool. Tiện ích APDUTool đệ trình các lệnh APDU vào môi trường chạy JavaCard, hoặc đến một môi trường chạy mô phỏng như JCWDE. Sau khi tập tin được thiết kế riêng, tiện ích APDUTool được chạy, chỉ định tệp kịch bản là đầu vào. APDUTool khởi động trình cài đặt trên thẻ, tải tập tin CAP. Nếu được yêu cầu trong tệp kịch bản, trình cài đặt trên thẻ sẽ tạo ra các applet được định nghĩa trong tệp CAP, để các applet có sẵn trong môi trường chạy JavaCard. Giao thức APDU gửi các lệnh APDU đến JCWDE hoặc một môi trường chạy JavaCard. Command APDU là những yêu cầu vận hành được thực hiện đối với một thẻ thông minh. Lớp APDU trong các API của JavaCard cung cấp một giao diện mạnh mẽ và linh hoạt để xử lý các APDU có cấu trúc lệnh và phản ứng phù hợp với tiêu chuẩn ISO 7816-4[6].



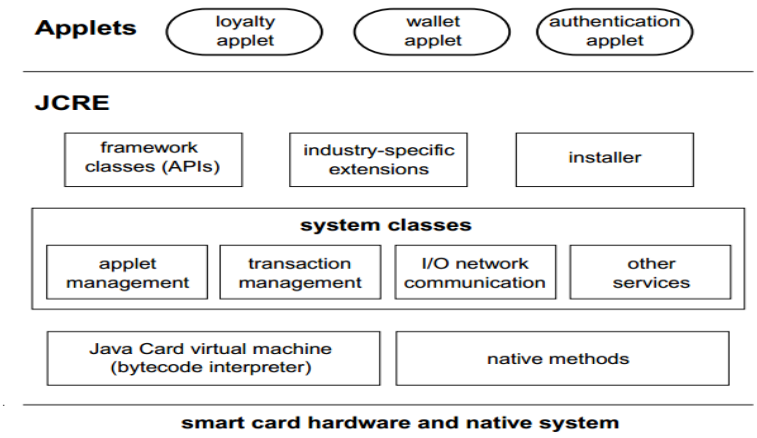
Hình 2.4 Trình cài đặt JavaCard và chương trình cài đặt ngoài thẻ.

Các lệnh APDU luôn luôn là các cặp, mỗi cặp có chứa một lệnh APDU, nó chỉ định lệnh, và một APDU phản hồi, nó sẽ gửi lại kết quả thi hành của lệnh. Trong thế giới thẻ, thẻ thông minh là những người giao tiếp phản hồi - nghĩa là họ không bao giờ bắt đầu truyền thông, họ chỉ đáp ứng với APDU từ thế giới bên ngoài. Ứng dụng đầu cuối gửi một lệnh APDU qua CAD. JCRE nhận lệnh, và chọn một applet mới hoặc chuyển lệnh tới applet hiện đang được chọn. Applet hiện đang được lựa chọn xử lý lệnh và trả về một APDU phản ứng cho ứng dụng đầu cuối. Command APDU và APDU phản hồi được trao đổi luân phiên giữa thẻ và CAD [6].

2.6 Môi trường chạy JavaCard

Trình cài đặt và trình biên dịch không phải là thành phần hệ thống duy nhất chạy trên JavaCard, nó còn nhiều các thành phần khác. Môi trường chạy JavaCard (JCRE) bao gồm các thành phần hệ thống thẻ Java chạy bên trong một thẻ thông minh. JCRE chịu trách nhiệm về quản lý tài nguyên thẻ, truyền thông mạng, thực hiện applet và trên hệ thống thẻ và bảo mật applet. Do đó, nó thực chất là hệ điều hành của thẻ thông minh.

JCRE – nền tảng chạy ứng dụng JavaCard, nằm trên phần cứng của thẻ thông minh. JCRE bao gồm máy ảo JavaCard (trình thông dịch bytecode), các lớp khung ứng dụng JavaCard (API), các phần mở rộng chuyên ngành và các lớp hệ thống JCRE. Các JCRE độc đáo tách biệt Applet từ các công nghệ độc quyền của các nhà cung cấp thẻ thông minh và cung cấp hệ thống tiêu chuẩn và các giao diện API cho các applet. Kết quả là applet dễ viết và dễ di chuyển trên các kiến trúc thẻ thông minh khác nhau. Lớp dưới cùng của JCRE chứa máy ảo Java Card (JCVM) và method native (có tính địa phương). JCVM thực thi các bytecode, kiểm soát việc cấp phát bộ nhớ, quản lý các đối tượng và thực thi bảo mật thời gian chạy, như đã giải thích trước đây. Các method native (có tính địa phương) cung cấp hỗ trợ cho lớp JCVM và lớp hệ thống lớp tiếp theo. Họ có trách nhiệm xử lý các giao thức truyền thông cấp thấp, quản lý bộ nhớ, hỗ trợ mã hoá, v.v..[6].



Hình 2.5 Kiến trúc hệ thống trên thẻ.

JCVM xử lý việc cấp phát bộ nhớ, thực hiện bytecode, quản lý đối tượng và bảo mật. Các method native (có tính địa phương) xử lý giao thức truyền thông mức thấp, quản lý bộ nhớ, hỗ trợ mã hoá và còn nhiều hơn nữa. Các lớp Hệ thống xử lý các nhiệm vụ mà một lõi hệ điều hành bình thường sẽ và gọi các method native (có tính địa phương) để thực hiện việc này. Các lớp API là các thư viện nhỏ gọn làm cho việc tạo các applet trở nên dễ dàng hơn. Trình cài đặt nạp applet vào thẻ, các phần mở rộng khác là các thư viện chuyên ngành, ví dụ như Open Platform mở rộng các dịch vụ JCRE để đáp ứng nhu cầu bảo mật đặc biệt của các ngân hàng.

JCRE được nạp vào thẻ Java tại nhà máy và vẫn duy trì ở đó cho đến khi thẻ bị phá hủy. Khi thẻ được đặt trong CAD (Card Accepting Device), nó được kích hoạt và bắt đầu sao chép dữ liệu từ chương trình từ EEPROM và ROM sang RAM nhanh hơn. Trong quá trình giao dịch, dữ liệu và các đối tượng phải được bảo toàn và được sao chép từ RAM vào EEPROM. EEPROM giữ dữ liệu khi không có điện, khi điện bị mất thẻ sẽ chuyển sang ngủ đông.

2.7 API JavaCard

Vì các ứng dụng JavaCard chủ yếu liên quan đến nhiều công ty phát hành thẻ, tính tương thích phải được thiết kế ngay từ đầu. Từ quan điểm kỹ thuật, chìa khóa là một JavaCard API đây là một lớp phần mềm cho phép ứng dụng giao tiếp với thẻ thông minh và đầu đọc của nhiều nhà sản xuất.

API có thể được coi là một thiết bị chuyên dụng hoạt động như một lớp phiên dịch giữa một ứng dụng và thẻ. API cho phép chạy ứng dụng để chọn thẻ thông minh từ nhiều nhà cung cấp. Việc mở một ứng dụng cho nhiều thẻ Java khuyến khích cạnh tranh giữa các nhà cung cấp thẻ và lợi ích của sự cạnh tranh đó - chất lượng tốt hơn và giá thấp hơn.

Một API không phải là một giao diện phổ quát mà sẽ làm việc với tất cả các thẻ Java. Thay vào đó, nó cung cấp một cách để các ứng dụng gửi lệnh tới hệ điều hành chip cụ thể của nhiều thẻ. Các lập trình viên có thể bắt đầu bằng cách phát triển một API cho hai hoặc ba thẻ và theo thời gian, mở rộng phần mềm để bao gồm một hay nhiều JavaCard. Có một giới hạn thực tế về kích thước của chương trình phần mềm có thể được lưu trữ trong một số thiết bị đầu cuối di động, nhưng API nên đủ linh hoạt để chứa các thẻ từ các nhà cung cấp cạnh tranh API cũng có thể được sử dụng để kiểm soát các phiên bản dữ liệu. Nếu cần thay đổi các yếu tố dữ liệu trên thẻ sau khi phát hành, API có thể được sử dụng để cập nhật thẻ mà không cần phải nhớ lại thẻ để định dạng lại.

JavaCard API bao gồm một tập lớp cho ứng dụng thẻ thông minh theo mô hình ISO 7816. API bao gồm ba package lõi và một package mở rộng. Ba package lõi đó là *java.lang*, *java.framework* và *java.security*. Package mở rộng đó là *java-cardx.crypto*. Có nhiều lớp nền tảng Java không hỗ trợ trong JavaCard API, ví dụ trong Java có một

số class về GUI interface, network I/O và hệ thống file I/O màn hình là không được hỗ trợ trong JavaCard. Đó là lý do tại sao thẻ thông minh không có hiển thị nó sử dụng giao thức mạng và cấu trúc file hệ thống.

Cụ thể hơn về các gói :

- Gói java.lang

Gói *java.lang* của JavaCard là một tập hợp con nghiêm ngặt của gói *java.lang* tương ứng trên nền tảng Java. Các lớp được hỗ trợ là Object, Throwable và một số lớp ngoại lệ liên quan đến máy ảo, như được trình bày trong bảng dưới. Đối với các class được hỗ trợ như đối tượng JavaCard Objectclass chỉ định nghĩa một constructor mặc định. *Java.lang package* cung cấp hỗ trợ ngôn ngữ Java cơ bản. Lớp Objectdefines một gốc cho phân cấp lớp JavaCard, và lớp Throwable cung cấp chung cho tất cả các ngoại lệ. Các lớp ngoại lệ được hỗ trợ đảm bảo ngữ nghĩa thống nhất khi một lỗi xảy ra do vi phạm ngôn ngữ Java. Ví dụ, cả máy ảo Java và máy ảo JavaCard thông qua NullPointerException một khi một tài liệu tham khảo null được truy cập.

Bảng 2.2: Các ngoại lệ

Đối tượng (Object)	Ngoại lệ (Exception)	Throwable
RuntimeException	ArrayIndexOutOfBoundsException	ArithmeticException
ArrayStoreException	IndexOutOfBoundsException	ClassCastException
NullPointerException	NegativeArraySizeException	SecurityException

- Gói *javacard.framework*

Các *javacard.framework* là một gói thiết yếu. Nó cung cấp các lớp khung và các giao diện cho các chức năng cốt lõi của một applet JavaCard. Quan trọng nhất, nó định nghĩa một lớp cơ sở Applet, cung cấp một khuôn khổ để thực hiện và tương tác với applet với JCRE trong suốt vòng đời applet. Vai trò của nó đối với JCRE cũng tương tự như của lớp Java Applet tới một trình duyệt lưu trữ. Một lớp applet người dùng phải mở rộng từ lớp cơ sở Applet và ghi đè các phương thức trong lớp Applet để thực hiện chức năng của applet.

Một lớp quan trọng khác trong gói *javacard.framework* là lớp APDU. APDU được thực hiện bởi các giao thức truyền dẫn. Hai giao thức truyền dẫn chuẩn là T = 0 và T = 1. APDU được thiết kế là giao thức truyền dẫn độc lập. Nói cách khác, nó được thiết kế cẩn thận sao cho những phức tạp và khác biệt giữa các giao thức T = 0 và T = 1 bị ẩn đi từ các nhà phát triển applet. Các nhà phát triển Applet có thể xử lý các lệnh của APDU dễ dàng hơn bằng cách sử dụng các phương pháp được cung cấp trong lớp APDU. Applet hoạt động chính xác bất kể giao thức truyền tải cơ bản mà nền tảng hỗ trợ.

Các nền tảng có gói java.lang.System là không được hỗ trợ. Nền tảng JavaCard cung cấp lớp javacard.framework.JCSystem, cung cấp một giao diện cho hành vi của

hệ thống. JCSystemclass bao gồm một bộ sưu tập các phương pháp để kiểm soát việc thực hiện aplelet, quản lý tài nguyên, quản lý giao dịch, và chia sẻ đối tượng interapplet trên nền Java Card.

Các lớp khác được hỗ trợ trong gói javacard.frameworkpackage là mã PIN, tiện ích và trường hợp ngoại lệ. Mã PIN là viết tắt của số nhận dạng cá nhân. Đây là hình thức mật khẩu phổ biến nhất được sử dụng trong thẻ thông minh để xác thực chủ thẻ.

- Gói *javacard.security*

Gói *Javacard.security* cung cấp một khuôn khổ cho các chức năng mật mã được hỗ trợ trên nền JavaCard. Thiết kế của nó dựa trên gói *java.security*.

Gói *Javacard.security* định nghĩa một khoá chính của nhà máy sản xuất các giao diện khác nhau đại diện cho các khóa mật mã được sử dụng trong các thuật toán đối xứng (DES) hoặc bất đối xứng (DSA và RSA). Ngoài ra, nó hỗ trợ các lớp cơ sở trừu tượng RandomData, Chữ ký và MessageDigest, được sử dụng để tạo ra dữ liệu ngẫu nhiên và để tính toán thông báo thư và chữ ký.

- Gói *javacardx.crypto*

Javacardx.crypto là một gói phần mở rộng. Nó chứa các lớp và giao diện mật mã phải tuân theo các yêu cầu về quản lý xuất khẩu của Hoa Kỳ. Gói *Javacardx.crypto* định nghĩa lớp cơ sở trừu tượng cho hỗ trợ chức năng mã hóa và giải mã.

Các gói javacard.security và gói javacardx.crypto xác định các giao diện API mà applet gọi để yêu cầu các dịch vụ mật mã. Tuy nhiên, họ không cung cấp bất kỳ thực hiện. Nhà cung cấp JCRE cần cung cấp các lớp triển khai các giao diện chính và mở rộng từ các lớp trừu tượng RandomData, Signature, MessageDigest và Cipher. Thông thường, một bộ xử lý đồng bộ riêng biệt tồn tại trên thẻ thông minh để thực hiện tính toán mật mã.

2.8 Package và quy ước đặt tên Applet

Hầu hết các ứng dụng quen thuộc được đặt tên và xác định bởi một tên chuỗi. Tuy nhiên, trong công nghệ JavaCard, mỗi applet được xác định và lựa chọn sử dụng một "định danh ứng dụng" (AID). Ngoài ra, mỗi gói Java được gán một AID. Điều này có nghĩa là một gói khi nạp vào một thẻ được liên kết với các gói khác đã được đặt trên thẻ thông qua AID của chúng. Quy ước đặt tên này phù hợp với đặc tả thẻ thông minh được định nghĩa trong ISO 7816.

Một AID là một mảng các byte có thể được hiểu là hai mảng riêng biệt, như trong bảng 2.3 phần thứ nhất là giá trị 5 byte được gọi là RID (mã nguồn). Phần thứ hai là một giá trị có độ dài biến được gọi là PIX (thuộc tính mở rộng định danh). PIX có thể từ 0 đến 11 byte chiều dài. Do đó một AID có thể từ 5 đến 16 byte trong tổng chiều dài. Định dạng của nó được mô tả trong:

Bảng 2.3 Cấu trúc ADI

Định danh ứng dụng (AID)	
Nhà cung cấp ứng dụng đăng ký quốc gia (RID)	Mã nhận dạng ứng dụng độc quyền - mở rộng (PIX)
5 bytes	0 to 11 bytes

2.9 Java Card Applet

JavaCard applet không nên nhầm lẫn với các applet Java chỉ vì chúng đều là các applet được đặt tên. Một JavaCard Applet là một chương trình Java tuân thủ một tập hợp các quy ước cho phép nó chạy trong môi trường chạy Java Card. Một JavaCard Applet không được dự định chạy trong môi trường trình duyệt. Lý do tên applet đã được chọn cho các ứng dụng Java Card được nạp vào môi trường chạy Java Card sau khi thẻ đã được sản xuất. Tức là, không giống như các ứng dụng trong nhiều hệ thống nhúng, các applet không cần phải ghi vào ROM trong quá trình sản xuất. Thay vào đó, họ có thể tự động tải xuống thẻ sau đó.

Lớp applet phải mở rộng từ lớp *javacard.framework.Applet*. Lớp Applet cơ bản là siêu lớp (super class) cho tất cả các applet cư trú trên một thẻ Java. Lớp applet là một kế hoạch chi tiết để giải thích các biến và các phương thức của một applet.

2.9.1 Tiến trình phát triển Applet

Sự phát triển của một applet JavaCard bắt đầu cũng giống như với bất kỳ chương trình Java nào khác. Một nhà phát triển viết một hoặc nhiều lớp Java và biên dịch mã nguồn với một trình biên dịch Java, tạo ra một hoặc nhiều lớp. Hình 2.6 trình bày quá trình phát triển applet.

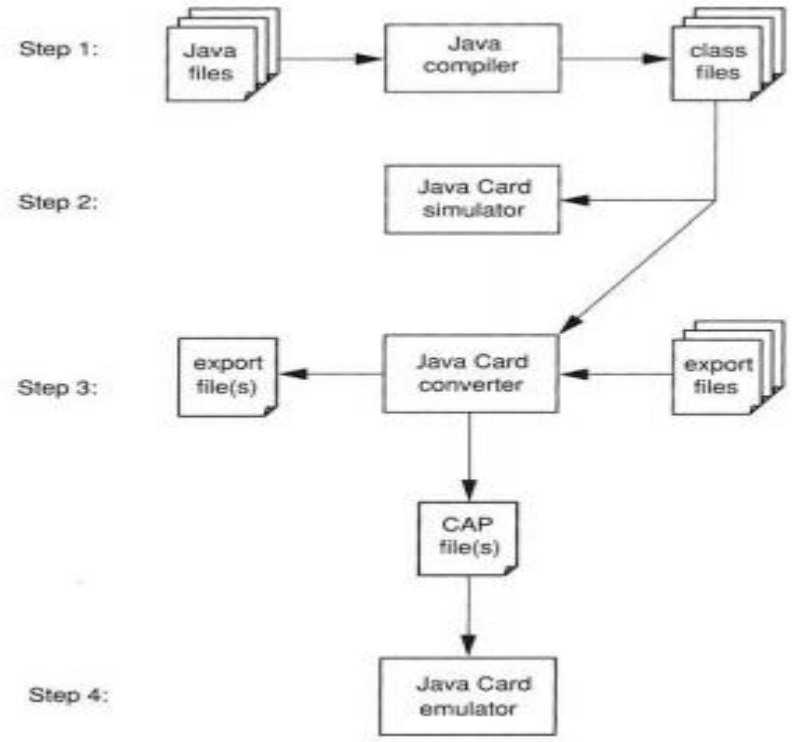
Tiếp theo, applet được chạy, thử nghiệm và gỡ lỗi trong một môi trường mô phỏng. Bộ mô phỏng mô phỏng môi trường chạy JavaCard trên máy PC hoặc một máy trạm khác.Trong môi trường mô phỏng, Applet chạy trên một máy ảo Java, và có các lớp của applet được thực thi. Theo cách này giả lập có thể sử dụng nhiều công cụ phát triển Java (máy ảo, trình gỡ lỗi, và các công cụ khác) và cho phép nhà phát triển kiểm tra hành vi của applet và nhanh chóng xem kết quả của applet không qua quá trình chuyển đổi. Trong bước này, các khía cạnh chức năng tổng thể của applet được kiểm tra. Tuy nhiên, một số tính năng thời gian chạy của máy ảo Java Card chẳng hạn như tường lửa applet và các hành của đối tượng, không thể được kiểm tra.

Sau đó, các tệp class của applet tạo nên một gói Java được chuyển đổi sang một tệp CAP bằng cách sử dụng JavaCard. Convertor – trình chuyển đổi thẻ Java không chỉ có đầu vào các tệp class được chuyển đổi mà còn là một hoặc nhiều tệp export. Khi

gói applet được chuyển đổi, bộ chuyển đổi cũng có thể export ra một tệp export cho gói đó. Một tệp CAP file hoặc tệp export, export ra một gói Java. Nếu một applet bao gồm một số gói, một CAP file và tệp export được tạo ra cho mỗi gói.

Trong bước tiếp theo, các tệp CAP đại diện cho applet được nạp và thử nghiệm trong môi trường mô phỏng. Trình mô phỏng cũng mô phỏng môi trường chạy JavaCard trên máy PC hoặc máy trạm. Tuy nhiên, trình mô phỏng là một công cụ thử nghiệm phức tạp hơn. Nó bao gồm một thực hiện máy ảo Java Card. Các hành vi của applet thực hiện trong giả lập nên được giống như hành vi của nó chạy trong một thẻ thực. Trong giai đoạn phát triển này, không chỉ applet được tiếp tục thử nghiệm. Mà còn là hành vi thời gian chạy của applet được đo. Trình gỡ lỗi cho phép nhà phát triển thiết lập các điểm ngắt hoặc chương trình đơn bước, theo dõi thời gian thực thi của sự thay đổi applet trong môi trường thời gian chạy của JavaCard đã mô phỏng hoặc giả lập.

Khi applet được thử nghiệm và sẵn sàng để tải xuống một thẻ thực, applet, đại diện bởi một hoặc nhiều CAP, được tải và cài đặt trong thẻ thông minh Java [6].



Hình 2.6 Tiến trình phát triển Applet[6]

2.9.2 Cài đặt applet

Việc cài đặt aplet được thực hiện tại nhà máy hoặc tại văn phòng của người phát hành và cũng có thực hiện sau khi phát hành, thông qua quá trình cài đặt an toàn (nếu nhà sản xuất thẻ xác định). Quá trình này bao gồm việc tải xuống một applet được ký kỹ thuật số, mà JCRE xác minh là hợp pháp, trước khi cài đặt applet. Các aplelet được cài đặt thông qua các tải không thể chứa các cuộc gọi phương thức tự nhiên do chúng không được tin cậy.

Applet gọi các phương thức tự nhiên phải được cài đặt tại nhà máy hoặc một môi trường tin cậy khác. Điều này được thực hiện vì lý do an ninh, vì các cuộc gọi nội bộ vượt qua khuôn khổ an ninh công nghệ Java và do đó phải được tin cậy cao trước khi được phép trên thẻ sau khi cài đặt JavaCard làm nền tảng. Không tương tác trực tiếp với thiết bị chấp nhận thẻ hoặc ứng dụng ngoại lệ. Các lớp được cài đặt có thể tương tác trực tiếp với JCRE hoặc với các lớp được cài đặt khác. JCRE chọn một applet và sau đó chuyển các APDU sang các applet đã chọn. Về bản chất, JCRE bảo vệ các nhà phát triển từ CPU thẻ thông minh, CAD và các giao thức truyền thông ISO cụ thể được sử dụng. JCRE cũng dịch các ngoại lệ không được ném ra bởi các lớp hoặc câu lệnh trả về bình thường trong các phương thức applet thành các giá trị trả về chuẩn ISO[6].

Kho lưu trữ cho một applet đã cài đặt không thể được phục hồi. Nếu một phiên bản mới hơn của applet được cài đặt, nó chiếm một vị trí lưu trữ mới và phiên bản trước đó của applet trở nên không thể truy cập được. Các applet thẻ Java cũng có thể được thực hiện không thể truy cập bằng cách loại bỏ tham chiếu của nó từ bảng đăng ký applet JCRE. Một khi các tài liệu tham khảo được gỡ bỏ, applet không còn có thể đạt được.

Việc cài đặt thẻ JavaCard làm cho các thành viên tĩnh của nó được khởi tạo. Công nghệ Java Card hỗ trợ khởi động tĩnh. Bộ khởi tạo không thể thực thi mã phần mềm Java, cũng không thể đặt thành viên tĩnh vào một giá trị không cố định (biến). Cài đặt cũng kết quả trong một cuộc gọi đến khởi tạo phương thức của applet (không giống như các applet Java).

Các ứng dụng đang chạy trong một thẻ thông minh Java giao tiếp với các ứng dụng máy chủ ở phía CAD bằng cách sử dụng APDU. Đối với mỗi lệnh APDU, applet đầu tiên giải mã giá trị của mỗi trường trong tiêu đề. Để thực thi các lệnh và đọc dữ liệu, applet có thể thực hiện các chức năng yêu cầu của lệnh bằng APDU. Đối với APDU phản ứng (command), applet cần định nghĩa một tập các từ trạng thái để cho biết kết quả xử lý lệnh APDU tương ứng. Trong quá trình xử lý thông thường, applet trả về từ trạng thái thành công. Nếu lỗi xảy ra, applet phải trả lại một từ trạng thái khác với thành công để biểu thị trạng thái bên trong của nó hoặc chẩn đoán lỗi.

Cài đặt Applet đề cập đến quá trình nạp các lớp applet trong một tệp CAP, kết hợp chúng với trạng thái thực thi của môi trường chạy Java Card, và tạo một ví dụ applet để đưa applet vào trạng thái có thể lựa chọn và thực thi. Trên nền tảng JavaCard, đơn vị tải và cài đặt là tệp CAP. Một tệp CAP bao gồm các lớp tạo nên gói Java. Một applet tối thiểu là một gói Java với một lớp duy nhất có nguồn gốc từ lớp

javacard.framework Applet. Một applet phức tạp hơn với một số lớp có thể được tổ chức thành một gói Java hoặc một tập các gói Java.

Để tải một applet, trình cài đặt thẻ sẽ thực hiện: thẻ sẽ lấy tệp CAP và chuyển đổi nó thành một chuỗi các lệnh APDU, mang nội dung tệp CAP. Bằng cách trao đổi các lệnh APDU với chương trình cài đặt không thẻ, trình cài đặt trên thẻ sẽ viết nội dung tệp CAP vào bộ nhớ liên tục của thẻ và liên kết các lớp trong tệp CAP với các class khác nằm trên thẻ. Trình cài đặt cũng tạo và khởi tạo bất kỳ dữ liệu nào được sử dụng nội bộ bởi JCRE để hỗ trợ applet này. Nếu applet yêu cầu một vài gói để chạy, mỗi tệp CAP được nạp vào thẻ.

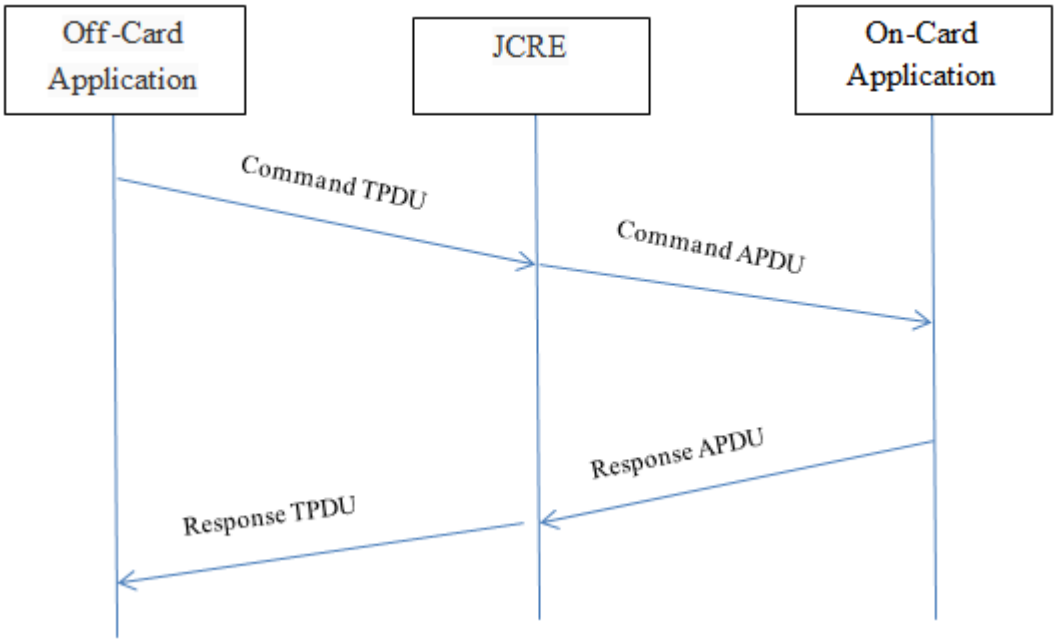
Bước cuối cùng trong quá trình cài đặt applet, trình cài đặt tạo ra một ví dụ applet và đăng ký với JCRE. Để làm như vậy, trình cài đặt sẽ gọi phương thức cài đặt:

public static void install (byte[] bArray, short offset, byte length)

Phương pháp cài đặt là một phương pháp của applet, tương tự như phương pháp chính trong một ứng dụng Java. Một applet phải thực hiện phương pháp cài đặt. Trong phương pháp cài đặt, nó gọi constructor của applet để tạo và khởi tạo một ví dụ applet. Các tham số cài đặt được gửi tới thẻ cùng với tệp CAP. Sau khi applet được khởi tạo và đăng ký với JCRE, nó có thể được chọn và chạy. JCRE xác định một applet đang chạy (một ví dụ applet), sử dụng AID. Applet này có thể tự đăng ký với JCRE bằng cách sử dụng mặc định AID được tìm thấy trong tệp CAP hoặc có thể chọn một tệp tin khác. Các thông số cài đặt có thể được sử dụng để cung cấp một AID thay thế JCRE là một môi trường đơn luồng. Điều này có nghĩa là chỉ một applet đang chạy cùng một lúc. Khi một applet được cài đặt lần đầu, nó ở trạng thái không hoạt động. Applet sẽ hoạt động khi nó được lựa chọn rõ ràng bởi một ứng dụng máy chủ lưu trữ Applet, giống như bất kỳ ứng dụng thẻ thông minh, là các ứng dụng có khả năng phản hồi. Sau khi được chọn, applet chờ đợi một ứng dụng đang chạy ở phía máy chủ để gửi một lệnh. Applet sau đó thực hiện lệnh và trả về một command với máy chủ lưu trữ[6].

2.10 Phương thức truyền nhận, trao đổi dữ liệu

Ứng dụng Java Card thực hiện trao đổi dữ liệu với ứng dụng trên thiết bị đầu cuối thông qua đơn vị dữ liệu giao thức truyền tải(Transmission protocol data unit - TPDU) và đơn vị dữ liệu giao thức ứng dụng (Application protocol data unit - APDU). Luận văn tập trung vào xem xét cấu trúc của APDU và phương thức trao đổi TPDU, APDU trong hai giao thức giao tiếp T=0 và T=1 của JavaCard. Quy trình trao đổi TPDU, APDU giữa hai ứng dụng trên thẻ và trên thiết bị đầu cuối được thể hiện trong hình 2.7.



Hình 2.7 Trao đổi thông tin giữa ứng dụng trên thẻ và ứng dụng trên thiết bị đầu cuối

APDU

- *Cặp câu lệnh - phản hồi*

Một đơn vị dữ liệu giao thức ứng dụng là một câu lệnh APDU hoặc một phản hồi APDU. Một tiến trình trong giao thức ứng dụng bao gồm việc gửi lệnh APDU, xử lý nội dung nhận được và trả về APDU phản hồi. Hai APDU đó tạo thành cặp câu lệnh - phản hồi[6].

Bảng 2.4 Cấu trúc câu lệnh APDU

Command APDU						
Header (required)				**Body (optional)**		
CLA	INS	P1	P2	L_c	Data Field	L_e

Bảng 2.5 Cấu trúc APDU phản hồi

Response APDU		
Body (optional)	**Trailer (required)**	
Data Field	SW1	SW2

Câu lệnh APDU bắt buộc phải chứa phần mở đầu gồm 4 byte: CLA, INS, P1, P2 và có thể có thêm phần thân với độ dài tùy biến. APDU phản hồi bắt buộc phải chứa phần mã trạng thái (Status Word - SW) gồm 2 byte: SW1, SW2 và có thể có thêm phần thân với độ dài tùy biến.

- Các trường dữ liệu trong cặp câu lệnh - phản hồi

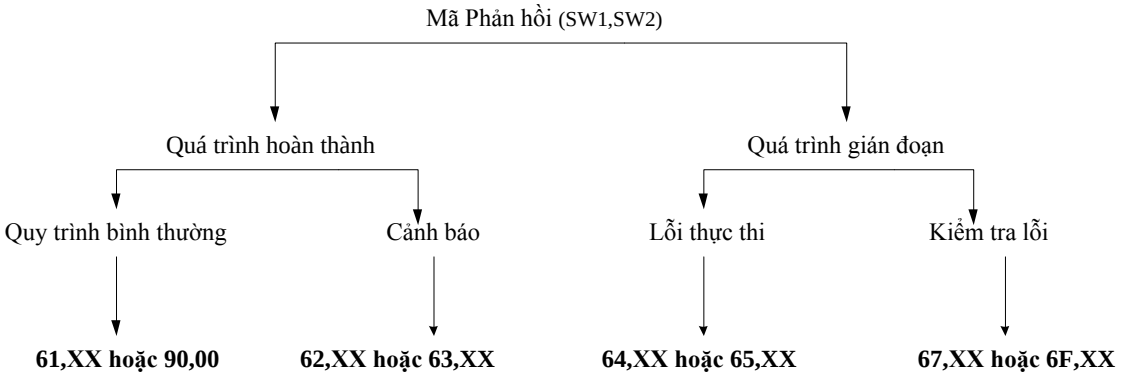
Mỗi một cặp câu lệnh - phản hồi có thể mang theo một trường dữ liệu câu lệnh/phản hồi

- *CLA*(1 byte): Trường bắt buộc này xác định một lớp hướng dẫn cụ thể cho từng ứng dụng. Các giá trị CLA hợp lệ được định nghĩa trong tiêu chuẩn ISO 7816-4.
- *INS*(1 byte): Trường bắt buộc này chỉ ra một hướng dẫn cụ thể trong lớp hướng dẫn được xác định bởi trường CLA. Tiêu chuẩn ISO 7816-4 xác định các hướng dẫn cơ bản để sử dụng để truy cập vào dữ liệu trên thẻ khi nó được cấu trúc theo hệ thống tệp tin trên thẻ như được định nghĩa trong tiêu chuẩn. Các chức năng bổ sung đã được chỉ định ở nơi khác trong tiêu chuẩn, một số là các chức năng bảo mật. Bạn có thể xác định các giá trị INS riêng biệt cho ứng dụng của mình chỉ khi sử dụng một giá trị byte CLA thích hợp, theo tiêu chuẩn
- *P1*(1 byte): trường bắt buộc này định nghĩa tham số chỉ dẫn 1. Bạn có thể sử dụng trường này để xác định trường INS, hoặc cho dữ liệu đầu vào.
- *P2*(1 byte): Trường bắt buộc này định nghĩa tham số chỉ dẫn 2. Bạn có thể sử dụng trường này để xác định trường INS, hoặc cho dữ liệu đầu vào.
- *Lc* (1 byte): trường tùy chọn này là số byte trong trường dữ liệu của lệnh.
- trường *Data* (biến, Lc số byte): trường tùy chọn này giữ dữ liệu lệnh.
- *Le* (1 byte): trường tùy chọn này chỉ định số byte tối đa trong trường dữ liệu của phản hồi dự kiến.

APDU phản hồi có trường bắt buộc:

- *Trường Data* (chiều dài thay đổi, được xác định bởi Le trong lệnh APDU): Trường tùy chọn này chứa dữ liệu trả về bởi applet.
- *SW1 (1 byte):* Trường bắt buộc này là từ trạng thái 1.
- *SW2 (1 byte):* Trường bắt buộc này là từ trạng thái 2

Các giá trị của từ trạng thái được định nghĩa trong tiêu chuẩn ISO 7816-4:



Hình 2.8 Mã trạng thái phản hồi

CHƯƠNG 3 MẬT MÃ TRÊN ĐƯỜNG CONG ELLIPTIC

Vấn đề đảm bảo An toàn thông tin đang được đặt lên hàng đầu trong bài toán giao dịch không an toàn trên Internet. Chúng ta cần có các giải pháp đảm bảo an toàn thông tin điều đó được xây dựng dựa trên lý thuyết mật mã, an toàn bảo mật thông tin. Các nhà khoa học đã phát minh ra những hệ mật mã như RSA, Elgamal,…nhằm che dấu thông tin cũng như là làm rõ chúng để tránh sự nhòm ngó của những kẻ cố tình phá hoại. Mặc dù rất an toàn nhưng có độ dài khoá lớn nên trong một số lĩnh vực không thể ứng dụng được. Chính vì vậy hệ mật trên đường cong elliptic ra đời, hệ mật này được đánh giá là hệ mật có độ bảo mật an toàn cao và hiệu quả hơn nhiều so với hệ mật công khai khác. Chương 3 trình bày những kiến thức về mật mã trên đường cong Elliptic.

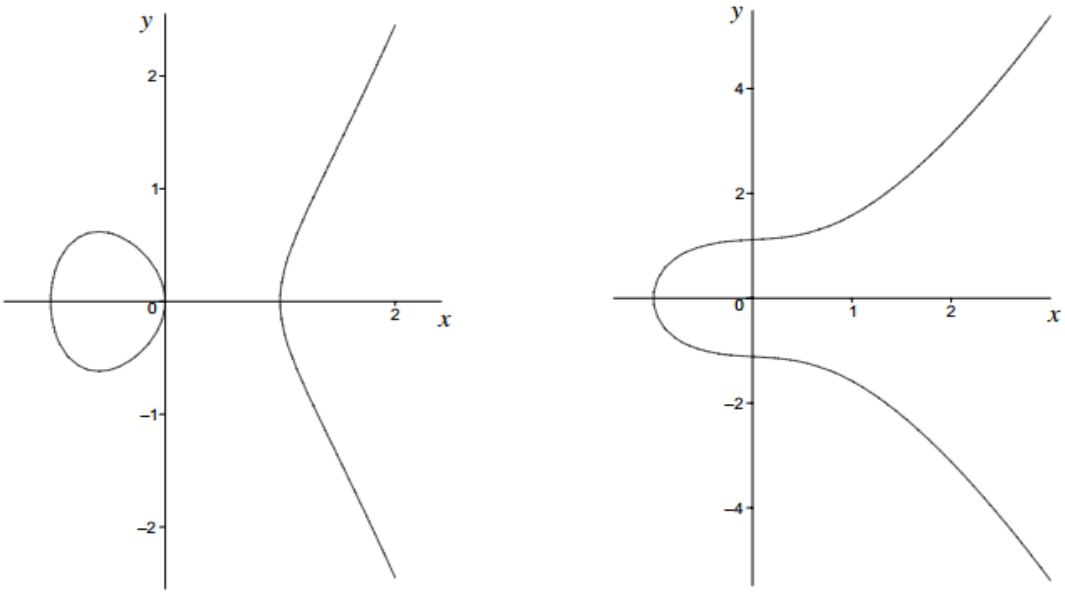
3.1 Cơ sở lý thuyết

a) Khái niệm đường cong Elliptic theo công thức Weierstrass

Đường cong elliptic E trên trường K là tập hợp các điểm (x,y) ∈ KxK thỏa mãn phương trình:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \ (a_i \in K \text{ và } 4a_4^3 + 27a_6^2 \neq 0)[4]$$

Một số ví dụ về đường cong Elliptic:



Hình 3.1 Một số ví dụ về đường cong Elliptic.

b) Đường cong Elliptic trên trường số thực R

Trên trường số thực R, đường cong Elliptic xác định bởi tập hợp các điểm (x,y) € R^2 thỏa mãn[4]:

$$y^2 = x^3 + ax + b, \text{ thỏa mãn điều kiện } 4a^3 + 27b^2 \neq 0$$

Các đa thức dạng $x^3 + ax + b$ có rất nhiều, do đó chúng ta phải chọn thêm một điểm *O* (điểm vô cực.

c) Đường cong elliptic trên trường F_p (p là số nguyên tố)

Cho p là số nguyên tố, (p>3). Một đường cong Eliptic trên trường F_p được định nghĩa bởi dạng:

$y^2 = x^3 + ax + b$, (1) Trong đó a, b ∈ F_p và $4a^3 + 27b^2 \not\equiv 0 \ (mod\ p)$.

Tập E(F_p) bao gồm tất cả các cặp điểm (x, y), với x, y ∈ F_p thỏa mãn phương trình (1) cùng với một điểm O – gọi là điểm tại vô cực.

$S = \{(x,y): y^2 = x^3 + ax + b, x, y \in F_p\} \cup \{O\}$[4].

Số lượng điểm của E(F_p) là #E(F_p) thỏa mãn định lý Hasse:

$$p + 1 - 2\sqrt{p} \leq \#E(Fp) \leq p + 1 + 2\sqrt{p}\ [4].$$

d) Đường cong Elliptic trên trường hữu hạn F_2^m

Một đường cong Eliptic E trên trường F_{2^m} được xác định bởi phương trình có dạng:

$$y^2 + xy = x^3 + ax^2 + b, (2)$$

Trong đó a, b ∈ F_{2^m}, và b≠0. Tập E(F_{2^m}) bao gồm tất cả các điểm (x,y), x, y ∈ F_{2^m} thỏa mãn phương trình (2) cũng với điểm O là điểm tại vô cực[4].

$S = \{(x,y): y^2 + xy = x^3 + ax^2 + b, x, y \in F_{2^m}\} \cup \{O\}$.

Số lượng điểm của E(F_{2^m}) là #E(F_{2^m}) thỏa mãn định lý Hasse:

$$p + 1 - 2\sqrt{p} \leq \#E(F_P) \leq p + 1 + 2\sqrt{p}.$$

Trong đó $p = 2^m$. Ngoài ra #E(F_{2^m}) là số chẵn.

a. Các phép toán trên đường cong Elliptic

Phép cộng:

Giả sử P(x_1,y_1) và Q(x_2,y_2) là hai điểm của E. Nếu $x_1=x_2$ và $y_1=-y_2$ thì ta định nghĩa P + Q =O. Ngược lại thì P + Q = (x_3,y_3) ∈ E trong đó : $x_3=\lambda^2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$. Với: P + O = O + P = P, ∀ P ∈ E(F_p)

Nếu P = (x, y) ∈ E(F_p), thì (x,y) + (x,-y) = O. Điểm (x, -y) được ký hiệu là -P và –P ∈ E(F_p)

Cho hai điểm P =(x_1, y_1) và Q = (x_2, y_2) ∈ E(F_p), nếu $P \neq \pm Q$ thì P+Q=(x_3,y_3) được xác định như sau:

Đặt $\lambda = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)$

$x_3 = \lambda^2 - x_1 - x_2$

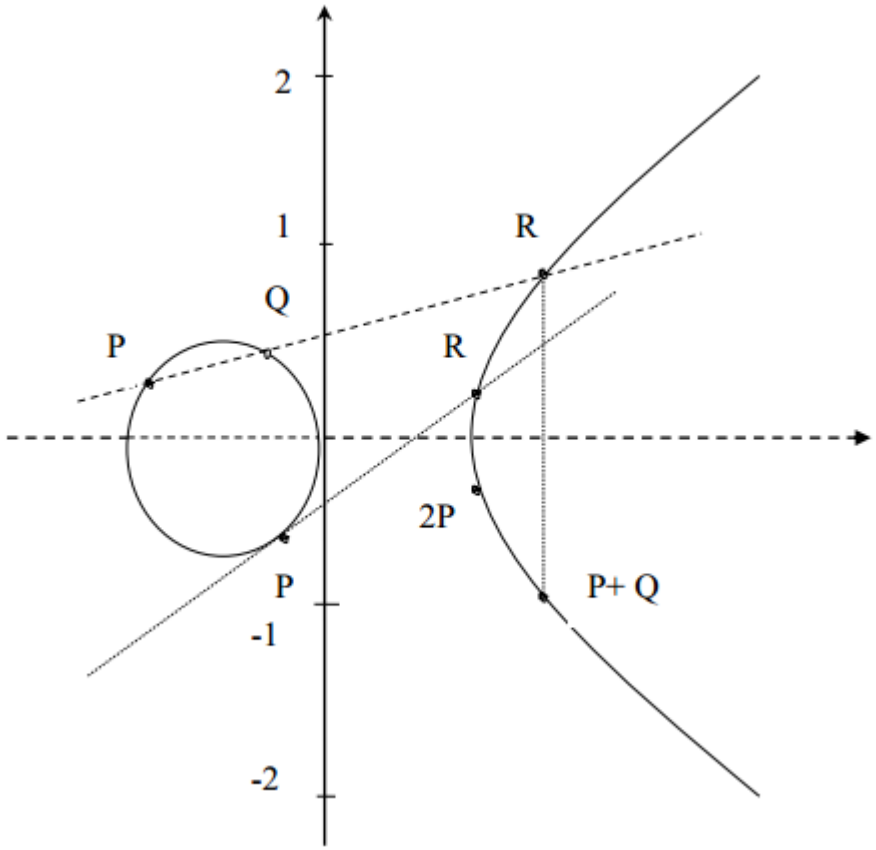
$y_3 = \lambda(x_1 - x_3) - y_1$.

Cho P =(x_1, y_1) ∈ E(F_p), P≠ -P. Khi đó 2P = (x_3,y_3) và được xác định như sau:

$x_3 = \lambda^2 - x_1 - x_2$

$y_3 = \lambda(x_1 - x_3) - y_1$

P = Q và y_1 = 0 thì P + Q = *O[4]*.



Hình 3.2 Phép cộng trên đường cong elliptic

- Phép nhân

Nếu cộng hai điểm P, Q ∈ E(R) với P = Q thì đường thẳng L sẽ là tiếp tuyến của đường cong elliptic tại điểm P. Trường hợp này điểm –R sẽ là giao điểm còn lại của L với E. Lúc đó R = 2P. Phép nhân Kp nhận được bằng cách thực hiện lặp k lần phép cộng[4].

3.2 Những chú ý để lựa chọn đường cong Elliptic phù hợp

Việc chọn một đường cong elliptic thế nào ảnh hưởng đến tốc độ, tính hiệu quả, độ dài khóa và tính an toàn của hệ mật mã trên đường cong này. Dù *E, K* và điểm cơ sở *P* ∈ *E* cố định và công khai nhưng việc chọn các tham số này phù hợp là bước quan trọng nhất.

3.2.1 Trường K

Trước hết chúng ta xem xét sự ảnh hưởng của trường *K* đến cấu trúc nhóm của *E*(*K*) và các hệ mật mã trên *E*(*K*). Một đường cong elliptic trên một trường hữu hạn tạo thành nhóm Abel được sử dụng trong mật mã học. Một ví dụ là việc chọn trường F_{2^r} giúp thực hiện các phép tính nhanh và dễ dàng triển khai được trên các thiết bị cứng. Tuy nhiên, các đường cong trên trường F_{2^r} có thể bị tấn công bởi MOV, trong khi các đường cong trên trường F_p (*p* là số nguyên tố lớn) lại chống lại được kiểu tấn công này. Rõ ràng, các đường cong elliptic trên trường nguyên tố F_p và trên trường F_{q^n} có các tính chất giúp chúng có thể thực thi được trên các thiết bị mà vẫn đảm bảo an toàn.

Một chú ý nữa là việc tính số điểm trên #*E*(*K*). Với #*E*(*K*) thích hợp có thể là điều kiện cho phép thực hiện tấn công Pohlig – Hellman. Có thể dùng thuật toán đơn định thời gian đa thức Schoof để tính #*E* trên trường hữu hạn F_q với đặc số khác 2, 3. Tốc độ của thuật toán Schoof phụ thuộc vào kích thước và đặc số của trường *K*. Ví dụ với *r* nhỏ, tính #*E*(F_{2^r}) có thể nhanh hơn một chút so với tính #*E*(F_p) với *p* lớn hơn đáng kể so với 2^r, nhưng khi *r* tăng thì tính #*E*(F_{2^r}) mất nhiều thời gian hơn tính #*E*(F_p).

3.2.2 Dạng của đường cong elliptic

Trước hết, chúng ta cần xem các dạng đường cong elliptic. Có 2 lớp đường cong elliptic được dùng trong các hệ mã hóa.

Supersingular Curve Menezes và Vanstone đã tìm ra các ưu điểm của các đường cong elliptic supersingular cho các hệ mật mã, đặc biệt trên trường F_{2^r}. Một đường cong elliptic trên trường hữu hạn có *q* phần tử được gọi là supersingular nếu $t^2 = 0, q, 2q, 3q,$ hoặc $4q$ với *t* được định nghĩa trong định lý Hasse, $t = q + 1 - \#E(F_q)$, $|t| \leq 2\sqrt{q}$. Tuy nhiên, các đường cong supersingular có thể bị tấn công bằng MOV.

Nonsupersingular Các đường cong nonsupersingular có bất biến *j* khác 0. Ưu điểm của các đường cong nonsupersingular là nó cung cấp độ bảo mật tương đương như các đường cong supersingular nhưng với các trường nhỏ hơn. Độ dài khóa ngắn giúp chúng có thể được triển khai trên các thiết bị như smart card. Hơn nữa, các đường cong nonsupersingular có thể chống lại tấn công MOV, ví dụ với nhóm con cyclic cỡ 2^{160}.

3.2.3 Phương pháp lựa chọn

Có một vài phương pháp để lựa chọn các đường cong elliptic. Phương pháp tự nhiên nhất là chọn ngẫu nhiên. Chọn ngẫu nhiên một đường cong elliptic *E* trên trường *K* và một điểm cơ sở $P \in E$. *K* được chọn và cố định trước. Phương pháp chọn ngẫu nhiên Koblitz cho các đường cong elliptic trên trường F_q (với *q* lớn) như sau:

- Phương pháp lựa chọn ngẫu nhiên Kobliz :

(1) Chọn ngẫu nhiên 3 phần tử từ F_q là x, y, a

(2) Tính $b = y^2 – (x^3 + ax)$

(3) Kiểm tra $4a^3+ 27b^2 \neq 0$ để đảm bảo phương trình $x^3+ ax + b =0$ không có nghiệm kép.

(4) Nếu điều kiện trên không thoả mãn quay lại bước 1.

(5) Còn lại, đặt P = (x, y) và đường cong $y^2= x^3+ ax +b$ là đường cong cần chọn.

Tuy nhiên phương pháp này có thể tạo ra các đường cong không đảm bảo một số yêu cầu định trước. Một kỹ thuật cải tiến là xây dựng các đường cong với các tính chất cho trước. Cũng có thể chọn những đường cong để tạo các hệ mã hóa không phụ thuộc vào bài toán EDLP, chẳng hạn các hệ elliptic dựa trên RSA. Các hệ mật mã elliptic làm việc với các nhóm con cylic của *E* với phần tử sinh là điểm *P*. Vì vậy, việc lựa chọn *P* phù hợp là rất quan trọng.

3.3 So sánh RSA và ECC

Sự khác nhau của RSA và ECC đã được so sánh bởi Robshaw và Yin đã chỉ ra rằng thời gian mã hóa của ECC nhanh gấp 8 lần so với RSA, còn giải mã và ký nhanh gấp 6 đến 7 lần. Certicom Corporation đã chứng minh rằng, nếu có các lựa chọn đường cong elliptic phù hợp thì tốc độ thực hiện của các hệ mật trên ECC nhanh gấp 10 lần so với RSA. Certicom đưa ra kết luận này khi thực thi các thuật toán trên đường cong elliptic trên trường hữu hạn nhị phân. Các so sánh tập trung vào 3 đặc điểm:

✓ **Độ an toàn (Security)**. Hệ mật đó đã được sử dụng rộng rãi bao lâu và tính an toàn của nó đã được nghiên cứu thế nào?

✓ **Tính hiệu quả (Efficiency)**. Độ phức tạp tính toán khi thực hiện.

✓ **Không gian lưu trữ (Space requirements)**. Không gian cần thiết để lưu trữ khóa cũng như các tham số khác của hệ mặt đó.

Độ an toàn

Độ an toàn của thuật toàn RSA phụ thuộc vào độ khó của bài toán IFP. Khi $n = p.q$ với p,q là các số nguyên tố lớn, càng lớn thì độ an toàn càng cao. Tuy nhiên, tốc độ thực hiện thuật toán tỷ lệ theo hàm mũ khi tăng dần độ lớn của p,q.

Bảng 3.1. Độ dài của khóa giữa RSA và ECC khi ở cùng mức an toàn

Thời gian tấn công (trong MIPS năm)	Kích thước khóa RSA (theo bit)	Kích thước khóa ECC (theo bit)	Tỷ lệ
10^4	512	106	5:1
10^8	768	132	6:1
10^{11}	1024	160	7:1
10^{20}	2048	210	10:1
10^{78}	21000	600	35:1

Nỗ lực lớn nhất để phá vỡ hệ mã ECC được đánh giá là nỗ lực để giải bài toán ECC2K – 108. Nó yêu cầu 4 tháng với gần 9500 máy và 1300 nhân viên từ 40 nước. Kết quả là bài toán ECC2K-108 đòi hỏi nỗ lực gấp 5 lần so với bài toán ECC2-97 được giải vào tháng 9 năm 1999. Việc tấn công ECC2K-108 dựa trên thuật toán

Pollard Rho (được phát minh độc lập bởi các chuyên gia ở Certicom – Rob Gallant, Rob Lambert, Scott Vanstone và nhóm của Harley). Nỗ lực để giải bài tóan ECC2K-108 cao gấp 50 lần so với bài toán 512 bit – RSA (khoảng 50 x 8000 MIPS năm)

Tính hiệu quả

Theo kết quả so sánh thực hiện năm 1998 bởi Certicom về thời gian thực hiện các thuật toán trên RSA 1024 bit và trên ECC 163 bit trên máy 167-MHz Ultra Spare chạy Solaris 2.5.1 như sau (ECNRA – thuật toán Nyberg-Rueppel trên EC, ECDSA – thuật toán DSA trên EC):

Bảng 3.2. So sánh thời gian thực hiện giữa RSA và ECC

Thuật toán	163 bit ECC (ms)	1024 bit RSA (ms)
Sinh cặp khóa	3.8	4708.3
Ký	2.1(ECNRA) 3.0(ECDSA)	228.4
Xác minh chữ ký	9.9(ECNRA) 10.7(ECNSA)	12.7
Trao đổi khóa Diffie-Hellman	7.3	1654.0

Không gian lưu trữ

Các hệ mật trên EC cung cấp tính an toàn tương đương với RSA với khóa có độ dài nhỏ hơn rất nhiều. Kết quả so sánh của Certicom về các tham số khóa như sau:

Báng 3.3. So sánh độ dài khóa của RSA và ECC

	Các tham số hệ thống	Khóa công khai (bit)	Khóa bí mật (bit)
1024 bit RSA	Không xác định	1088	2048
160 bit ECC	481	161	160

RSA hiện nay vẫn đang là hệ mặt mã khóa công khai phổ biến nhất. Hội nghị 2005 ở Toronto về ECC và ứng dụng, cùng hội nghị năm 2006 về ECC của Certicom với chủ đề "The Cryptography of today and tomorrow" đã cho thấy ECC sẽ là hệ mật thay thế RSA trong tương lai gần.

3.4 Mật mã trên đường cong elliptic

Hệ mật dựa trên đường cong Elliptic (ECDSA/ECC) là một giải thuật khoá công khai. Hiện nay, hệ mật RSA là giải thuật khoá công khai được sử dụng nhiều nhất, nhưng hệ mật dựa trên đường cong Elliptic (ECC) có thể thay thế cho RSA bởi mức an toàn và tốc độ xử lý cao hơn.

Ưu điểm của ECC là hệ mật mã này sử dụng khoá có độ dài nhỏ hơn so với RSA. Từ đó làm tăng tốc độ xử lý một cách đáng kể, do số phép toán dùng để mã hoá và giải mã ít hơn và yêu cầu các thiết bị có khả năng tính toán thấp hơn, nên giúp tăng tốc độ và làm giảm năng lượng cần sử dụng trong quá trình mã hoá và giải mã. Với cùng một độ dài khoá thì ECC có nhiều ưu điểm hơn so với các giải thuật khác, nên trong một vài năm tới có thể ECC sẽ là giải thuật trao đổi khoá công khai được sử dụng phổ biến nhất.

Hệ mật đường cong Elliptic dựa trên độ khó khi biết được điểm P và Q và phải tìm ra giá trị k. Bên cạnh công thức của đường cong Elliptic, thì một thông số quan trọng khác của đường cong Elliptic là điểm G(còn gọi là điểm cơ sở). Điểm G đối với mỗi đường cong elliptic là cố định, trong hệ mật mã ECC thì một số nguyên lớn k đóng vai trò như một khoá riêng, trong khi đó kết quả của phép nhân giữa k với điểm G được coi như là khoá công khai tương ứng.

✓ *Sơ đồ hệ mã hóa dựa trên đường cong Elliptic*

Giả sử Alice và Bob muốn trao đổi thông tin mật cho nhau trên cơ sở đường cong Elliptic, thì Alice và Bob chọn đường cong Elliptic E với các hệ số a, b, modulo p và điểm khởi tạo G, G có bậc là n (nG=0).

Alice hình thành khóa mật và khóa công cộng qua các bước sau:

1. Chọn d là số ngẫu nhiên làm khóa mật thỏa mãn .
2. Công bố khóa công cộng Q=d*G.

Quá trình mã hóa:

Bob muốn gửi thông tin mật m cho Alice, Bob thực hiện:

1. Chọn số ngẫu nhiên k . Và tính điểm $Gk(x_1,y_1)=k*G$.
2. Tính giá trị Qk(x,y)=k*Q.
3. Để mã hóa, Bob chọn tọa độ của điểm Qk để mã hóa. Ví dụ như chọn tọa độ x, và mã hóa thông điệp m: c=m.x (mod p).
4. Gửi cặp ($Gk(x_1,y_1)$, c) cho Alice.

Quá trình giải mã:

Nhận được cặp ($Gk(x_1,y_1)$, c) từ Bob. Alice tiến hành giải mã qua các bước sau:

- Tính PBk(x',y')=d*Gk

- Chọn tọa độ x của điểm H_k và tìm phần tử nghịch đảo của x' là và tính giá trị m bằng biểu thức: m= $x'^{-1}.c$

Chúng ta kiểm tra tính đúng đắn của hệ. Từ bước 1 của quá trình giải mã chúng ta thấy d * Gk=k.d *G= k* Q= Qk(x,y), nên x'=x.

Và quá trình giải mã là đúng.

3.5 Chữ ký số trên hệ mật đường cong Elliptic

3.5.1 Sơ đồ chữ ký ECDSA

[2]Để thiết lập sơ đồ chữ ký ECDSA (Elliptic Curve Digital Signature Algorithm), cần xác định các tham số: lựa chọn đường cong E trên trường hữu hạn Fq với đặc số p sao cho phù hợp, điểm cơ sở G ∈ E(Fq).

a. Sinh khóa

1. Chọn số ngẫu nhiên d trong khoảng [2, n-1] làm khóa bí mật.
2. Tính Q=dG làm khóa công khai.

b. Ký trên bản rõ m

1. Chọn một số ngẫu nhiên k, $2 \leq k \leq n-1$
2. Tính $kG = (x_1, y_1)$.
3. Tính $r = x_1 \bmod n$. Nếu r=0, quay lại bước 1.
4. Tính $k^{-1} \bmod n$
5. Tính $s = k^{-1}(m + dr) \bmod n$. Nếu s = 0, quay lại 1.
6. Chữ ký trên thông điệp m là (r,s)

c. Kiểm tra chữ ký

1. Kiểm tra r và s có là các số tự nhiên trong khoảng [2, n-1] không.
2. Tính $w = s^{-1} \bmod n$
3. Tính $u_1 = mw \bmod n$ và $u_2 = rw \bmod n$.
4. Tính $X = u_1G + u_2Q = (x_x, y_y)$
5. Nếu X = O thì phủ nhận chữ ký . Ngược lại tính $v = x_x \bmod n$
6. Chữ ký chỉ được chấp nhận nếu v = 4

d. Xác thực

Nếu chữ ký (r,s) trên m là đúng thì $s = [k^{-1}(m + dr)] \bmod n$.

$k \equiv s^{-1}(m + dr) \equiv s^{-1}m + s^{-1}rd \equiv wm + wrd \equiv u_1 + u_2d \pmod n$.

Vì vậy, $u_1G + u_2G = (u_1 + u_2d)G = kG$ và vì vậy v = r.

Ví dụ:

+ Chuẩn bị tham số: Chọn đường cong elliptic E: $y^2 = x^3 + x + 1$ trên trường Z_{23}, điểm ở vô cùng O, điểm cơ sở G(17, 3).

Tính bậc n của G:

Ta có 2G = (13, 16); 4G = (5, 19); 6G = (17, 20) = -G; 7G = G + (-G) = O

Suy ra n = 7 là bậc của G.

+ Tạo khóa:

1. Chọn số ngẫu nhiên d = 6 làm khóa bí mật.

2. Tính $Q(x_Q, y_Q) = 6G = (17, 20)$ làm khóa công khai.

+ Ký số trên bản rõ:

1. Chọn một số ngẫu nhiên $2 \in [2, 6]$

2. Tính điểm 2G = (13, 16)

3. Tính r = 13 mod 7 = 6.

4. Tính $2^{-1} \pmod 7 = 4$ vì 2*4 mod 7 = 1

5. Tính $s = 4 (5 + 6*6) \bmod 7 = 3 \neq 0$

6. Chữ ký trên thông điệp m là (6, 3)

+ Kiểm tra chữ ký:

1. r = 6 và s = 3 trong khoảng [2, 6].

2. Tính $w = 3^{-1} \pmod 7 = 5$ vì 3*5 mod 7 = 1

3. Tính $u_1 = 5*5 \bmod 7 = 4$ và $u_2 = 6*5 \bmod 7 = 2$

4. Tính X = 4G + 2Q = (5, 19) + 2 (17, 20) = (5, 19) + (13, 7) = (13, 16)

5. Vì $X \neq O$ nên tính v = 13 mod 7 = 6.

6. v = r nên chữ ký là đúng!

3.5.2 Sơ đồ chữ ký Nyberg – Rueppel

[2]Giả sử E là một đường cong Elliptic trên trường Zp (p>3 và nguyên tố) sao cho E chứa một nhóm con cyclic H trong đó bài toán logarith rời rạc là "khó".

Với $P=Z_p^* x Z^*_p$, $C = E x Z_p^* x Z_p^*$ ta định nghĩa $K=\{(E,Q,a,R):R=aQ\}$ với $Q \in E$. Các giá trị α và R là công khai, a là bí mật.

Với K=(E,Q,a,R) chọn số ngẫu nhiên $k \in Z_{|H|}$. Khi đó, với $x = (x_1,x_2) \in Z_p^* x Z_p^*$ ta định nghĩa $sig_k(x,k) = (c,d)$, trong đó:

1. $(y_1,y_2) = kQ$

2. $c = y_1 + hash(x) \bmod p$

3. $d = k – ac \bmod p$

4. $ver_K(x,c,d) = true \leftrightarrow hash(x) = e$

5. $(y_1,y_2) = dQ + cR$

6. $e = c – y_1 \bmod p$

3.5.3 Sơ đồ chữ ký mù Harn trên ECC

[2] Harn đã công bố một sơ đồ chữ ký mù tựa như sơ đồ ECDSA năm 1994. Chữ ký mù là chữ ký thực hiện trên một văn bản mà người ký hoàn toàn không biết nội dung. Điều này thực hiện được vì người trình ký đã sử dụng một phương pháp nào đó để che dấu nội dung của văn bản gốc để người ký không biết. Để người ký yên tâm, người xin cấp chữ ký phải chứng minh tính hợp lệ của nội dung đã bị che dấu.

a. Sinh khóa:

Chọn các tham số cho đường cong Elliptic:

1. Chọn số nguyên tố p và số nguyên n

2. Với 2 phần tử a_1,a_2 của $GF(p^n)$, xác định phương trình của E trên $GF(p^n)$ ($y^2 = x^3 + a_1x + a_2$ trong trường hợp p>3) với $4a_1^3 + 27a_2^2 \neq 0$

3. Với 2 phần tử x_G và y_G trong GF(p^n) xác định một điểm G=(x_G,y_G) trên E(GF(p^n)) (G ≠ O với O là điểm gốc)

4. Giả sử điểm G có bậc q

Việc sinh khóa bao gồm:

(1) Chọn một khóa bí mật d là số nguyên ngẫu nhiên trong [2, q – 1]

(2) Tính khóa công khai Q, là một điểm trên E sao cho Q = dG

b. Ký mù

Giả sử Bob yêu cầu Alice ký lên một văn bản m_o mà m là đại diện của văn bản này (m = H(mo) với H là một hàm băm nào đó). Giao thức ký được thực hiện như sau:

1. Alice sinh ra cặp khóa $(\overline{k}, \bar{R})$ theo cách sau: chọn ngẫu nhiên $\overline{k} \in [2, q-1]$ và tính $\bar{R}=\bar{k}G = (x_{\bar{k}}, y_{\bar{k}})$. Đặt $\bar{r} = x_{\bar{k}}$ rồi gửi $\bar{r}$ và $\bar{R}$ cho Bob
2. Bob chọn các tham số làm mù a,b ∈ [1, q-1], tính R trên E sao cho R = a $\bar{R}$ + bG = (x_k,y_k) và tính r = c(x_k) và $\overline{m}$ = (m+r)a^{-1} - $\bar{r}$. Sau đó gửi $\overline{m}$ cho Alice ($\overline{m}$ là m sau khi đã bị làm mù)
3. Alice tính $\bar{s}$ = d($\overline{m} + \bar{r}$) + $\overline{k}$ $(mod\ q)$, rồi gửi $\bar{s}$ cho Bob
4. Bob nhận được $\bar{s}$, xóa mù để có được chữ ký s trên m bằng cách tính s =a$\bar{s}$ +b cặp (r,s) là chữ ký trên m

3.5.4 Sơ đồ chữ ký mù bội Harn trên ECC

[2] Đa chữ ký hiểu là chữ ký được tạo thành bởi nhiều người ký. Có văn bản cần được ký bởi một số người thay vì một người nhằm bảo đảm tính an toàn. Những người ký không biết về nội dung văn bản ký.

a. Sinh khóa: Việc chọn các tham số cho đường cong elliptic tương tự như sơ đồ chữ ký Harn. Giả sử rằng có t người ký là U_i với i=1,….,t. Việc sinh khóa được thực hiện qua các bước:

1. Mỗi người ký U_i chọn ngẫu nhiên một khóa bí mật d_i là một số nguyên thuộc [2, q – 1].
2. Khóa công khai của người ký U_i là điểm: $Q_i = d_iG = (x_{di}, y_{di})$, i=1,….,t
3. Khóa công khai cho tất cả người ký là: $Q = Q_1 + \ldots. + Q_t = dG = (x_d,y_d)$ với $d= d_1+ \ldots. + d_t$ (mod q)

b. *Ký mù trên m*:

1. Người ký U_i sinh một lần cặp $(\overline{k_i}, \overline{R_i})$ bằng cách chọn ngẫu nhiên $\overline{k_i} \in$ [2,q-1] và tính $\overline{R_i} = \overline{k_i}G = (x_{ki},y_{ki})$. U_i đặt $\overline{r_i} = x_{\overline{ki}}$, i=1,…,t rồi gửi $\overline{r_i}$ và $\overline{R_i}$ cho ban thư ký
2. Ban thư ký chọn các tham số làm mù a,b ∈ [1,q-1], tìm điểm R trên E sao cho R=a$\bar{R}$ + bQ = (x_k,y_k) trong đó $\bar{R} = \overline{R_1} + \ldots.. + \overline{R_t}$ và Q= Q_1 + …..+ Q_t. Ban thư ký tính r=c(x_k) (modq) và $\overline{m}$ =(m+r+b)a^{-1} -$\bar{r}$. Sau đó gửi $\overline{m}$ và $\bar{r}$ đến cho từng người ký U_i
3. U_i tính chữ ký $\overline{s_i}$ = d_i ($\overline{m} + \bar{r}$) + $\overline{k_i}$ (mod q), i=1,…,t, gửi $\overline{s_i}$ tới ban thư ký
4. Ban thư ký tính $\overline{s_i}$G - ($\overline{m} + \bar{r}$)Q_i =(x_{ei},y_{ei}) và kiểm tra $\bar{r}_I$ có bằng x_{ei} (mod q)

i=1,….,t. Chữ ký mù nhóm ECC là cặp (r,s) trong s =$\bar{r}$a(mod q) và

$\bar{s}$ =$\bar{s}_i$ +….+ $\bar{s}_t$ (mod q)

3.6 Đánh giá các tấn công hệ mật trên đường cong Elliptic

3.6.1 Phương pháp Baby step - Giant step

Phương pháp Babystep – GiantStep là phương pháp tấn công đầu tiên lên hệ mật mã ECC, và thực hiện với thời gian là hàm mũ. Nó giải bài toán DLP trong trường nguyên tố Zp được mở rộng cho bài toán EDLP. Bài toán Tìm k sao cho kG = Q trên E(Fq) với #E(Fq) = N, giả sử k tồn tại thực sự.

Thuật toán:

1. Chọn số nguyên m > $\sqrt{N}$.
2. Tính mG.
3. Với i = 0 đến i = m-1 tính (và lưu lại) iG.
4. Với j = 0 đến j = m-1 tính (và lưu lại) Q – jmG.
5. Sắp xếp danh sách trong bước 3 và 4 theo một thứ tự nhất định.
6. So sánh các danh sách ở các bước 3 và 4 cho đến khi tìm được cặp i, j thỏa mãn iG = Q – jmG.
7. Kết quả trả lại là k ≡ i + jm (mod N).

- *Đánh giá*: đối với các nhóm điểm đường cong elliptic cấp N, phương pháp này cần khoảng $\sqrt{N}$ bước tính và $\sqrt{N}$ bộ nhớ.

3.6.2 Phương pháp Pohlig – Hellman

Định nghĩa: Giả sử $p-1=\prod_{i-1}^{k} p_i^{Ci}$, p_i là số nguyên tố đặc biệt. Giá trị a = $\log_\alpha \beta$ được xác định một cách duy nhất theo modulo p-1. Trước hết nhận xét rằng, nếu có thể tính a mod p_i^{Ci} với mỗi i, 1 ≤ i ≤ k, thì:

$$p\text{-}1 \equiv 0 \pmod{q^c}$$

$$p\text{-}1 \neq 0 \pmod{q^{c+1}}$$

có thể tính a mod (p-1) theo định lý phần dư. Để thực hiện diều đó ta giả sử rằng q là số nguyên tố.

Thuật toán Pohlig - Hellman để tính log$\alpha\beta$ mod qc:

1. Tính $\gamma = \alpha^{(p-1)/q} \bmod p$ với 0 ≤ i ≤ q-1
2. Đặt j = 0 và $\beta_j = \beta$
3. **While** j ≤ c-1 **do**
4. Tính $\delta = \beta_j^{(p-1)/q\, j+1} \bmod p$
5. Tìm i sao cho $\delta = \gamma_i$
6. a_j = i
7. $\beta_{j+1} = \beta_j\, \alpha^{-aj\, qj} \bmod p$
8. j = j +1

- Phương pháp Pohig – Hellman nó thực hiện tốt tất cả các ước nguyên tố của N là nhỏ. Nếu ước nguyên tố lớn nhất xấp xỉ độ lớn của N thì phương pháp này rất khó áp dụng. Do đó các hệ mật dựa trên logrith rời rạc thường chọn bậc của nhóm có chứa một thừa số nguyên tố lớn.

3.6.3 Tấn công MOV

Phương pháp tấn công MOV (được đưa ra bởi Menezes, Okamoto, và Vanstone) làm yếu bài toán logarit rời rạc trên đường cong elliptic $E(F_q)$ thành bài toán logarith rời rạc trên trường F_q^m với m nào đó. Khi đó có thể tấn công bằng tấn công chỉ số, nhất là khi m nhỏ. Vì bài toán logarithm rời rạc trong các trường hữu hạn có thể bị tấn công bởi phương pháp tính chỉ số và nó giải nhanh hơn bài toán logarithm rời rạc trên đường cong elliptic với điều kiện là trường F_q^m không lớn hơn nhiều so với trường F_q.

Tấn công MOV chỉ ra không phải loại đường cong elliptic nào cũng có thể được sử dụng, bằng cách đưa ra việc giải quyết bài toán logarit rời rạc trên đường cong elliptic trở thành bài toán logarit rời rạc trên một trường hữu hạn mở rộng với độ mở rộng tùy thuộc vào loại đường cong. Do đó, các đường cong siêu lạ rất được ưa dùng ở giai đoạn đầu lại trở nên rất yếu vì ở đó độ mở rộng chỉ cao nhất là 6. Nét đặc biệt trong tấn công MOV là việc sử dụng phép ghép cặp (pairings) trên các đường cong elliptic, với những kết quả khá mới mẻ trong lý thuyết số. Không chỉ phục vụ cho việc phá mã, việc sử dụng phép ghép cặp sau đó đã trở nên cực kỳ hữu hiệu trong việc xây dựng mã.

3.6.4 Phương pháp Index và Xedni

Thuật toán tính chỉ số ngược đầu tiên là nâng các điểm P_1 , P_2 ,..., P_n , sau đó chọn một đường cong Elliptic E (Q) chứa các điểm đã nâng và hy vọng rằng chúng phụ thuộc tuyến tính. Nghĩa là thỏa mãn quan hệ $\sum_{i=1}^{r} ni\ Pi \ = 0$. Tuy nhiên, xác suất để chúng phụ thuộc tuyến tính là nhỏ.

Thuật toán Index và Xedni có thời gian chạy là hàm mũ và không hiệu quả trong thực tế. Do thuật toán Index và Xedmi vướng phải hai bài toán khó bởi vì:

- Bài toán tìm được phép nâng theo nghĩa tạo ra các điểm nâng phụ thuộc tuyến tính bài toán này khó do xác suất các điểm nâng phụ thuộc tuyến tính là rất nhỏ.
- Giải phương trình quan hệ tuyến tính cũng rất khó vì độ cao của các điểm nâng là rất lớn.

3.6.5 Các tấn công dựa trên giả thuyết Diffie – Hellman

Tấn công này chỉ ra rằng nếu p-1 có một ước d thỏa mãn $\approx \sqrt{p}$ thì khóa bí mật có thể được tính với $O(\sqrt[4]{p})$ bộ nhớ. Nếu p+1 có một ước d thỏa mãn $d \approx \sqrt[3]{p}$ thì khóa bí mật có thể được tính là $O(\log p.\sqrt[3]{p})$ phép toán sử dụng $O(\sqrt[3]{p})$ bộ nhớ.

3.6.6 Các tấn công cài đặt

Kiểu tấn công cài đặt thứ nhất là dựa trên điểm không hợp lệ của đường cong Elliptic. Nó được áp dụng trong một số giao thức cụ thể như mã hóa tích hợp đường cong Elliptic hoặc giao thức trao đổi khóa ECDH một pha. Nếu trong quá trình nhận và xử lý một điểm trên đường cong mà không thực hiện việc kiểm tra xem nó có thực sự nằm trên đường cong đã cho hay không thì lược đồ có thể bị tấn công.

Dạng tấn công thứ hai là kiểu tấn công phân tích năng lượng để khám phá khóa bí mật. Hiệu quả của các kiểu tấn công này phụ thuộc vào cách cài đặt cụ thể.

3.7 Chuẩn tham số cho hệ mật Elliptic

Trên thế giới hiện nay các chuẩn tham số cho hệ mật Elliptic được đưa ra trong các chuẩn:

- ISO 15496-5
- ANSI X9.62
- FIPS PUB 186-3
- Certicom SEC1 version 2.0

➢ Tất cả các chuẩn đưa ra đều có các đặc điểm chung tiêu chuẩn như sau:

- Về ngưỡng an toàn: tiêu chuẩn này đánh giá khả năng thám mã tại thời điểm đưa ra chuẩn
- Modulo P: P là số nguyên tố được sinh ra theo phương pháp tất định hoặc xác suất, có thể ở dạng đặc biệt nhằm tăng tốc độ tính toán, có độ dài theo bit và được chọn bằng 2*bit với bit ở ngưỡng an toàn.
- Độ dài khóa: Tiêu chuẩn này phải đảm bảo độ dài khóa phải có ước nguyên tố lớn N. Độ lớn của N phải đảm bảo cho độ phức tạp của bài toán ECDLP là lớn hơn ngưỡng an toàn.
- Tiêu chuẩn tránh các đường cong yếu: tuyệt đối không sử dụng các đường cong siêu kỳ dị và bất quy tắc
- Tiêu chuẩn MOV: Sử dụng bài toán ECDLP và bài toán DLP trên trường hữu hạn mở rộng
- Giả thuyết Diffie-Hellman: Tiêu chuẩn này nói về các kiểu tấn công phụ thuộc vào phân rã của $N\pm1$

➢ Các điểm khác nhau: ngoài các điểm chung thì có các điểm khác nhau nhất định như sau:

- Định lượng về giá trị cận của MOV.
- Chuẩn ISO và SEC1 v2-2009 đưa ra điều kiện về giả thuyết Diffie-Hellman như sau:

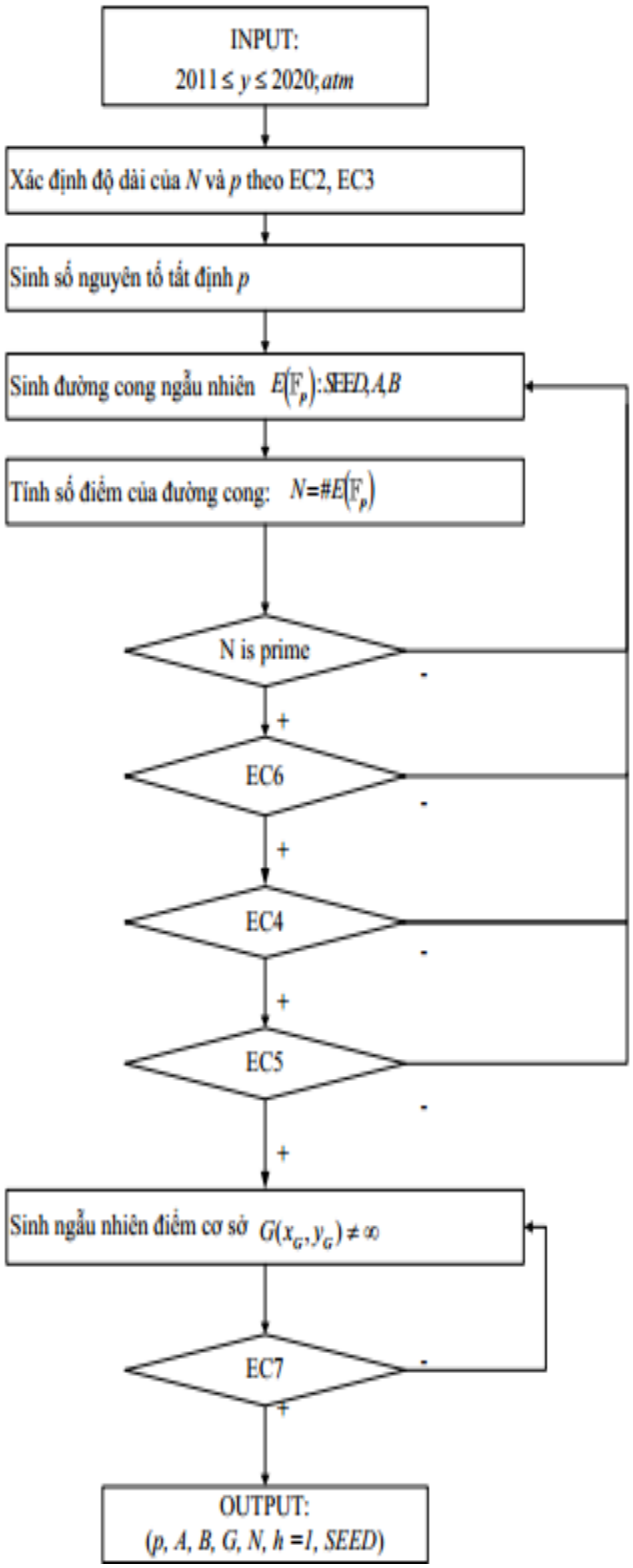
+ ISO: Độ dài khóa của đường cong có ước nguyên tố N sao cho ước d và e của $N\pm1$ thỏa mãn điều kiện $d,e \notin [(\log N)^2, \sqrt{N})]$

+ SEC1: Độ dài khóa của đường cong có ước nguyên tố N thỏa mãn điều kiện mỗi số $N\pm1$ phải có một ước nguyên tố lớn r sao cho Log N (r) > 19/20.

3.8 Sinh tham số cho hệ mật Elliptic

3.8.1 Tham số miền của đường cong Elliptic

- Thuật toán sinh tham số miền của đường cong Elliptic[4]:



Hình 3.3 Thuật toán sinh tham số miền đường cong elliptic

- Mô tả thuật toán:

- Input: Số năm *y* trong khoảng 2011 -2020 và biến *atm* (bằng 1 xác định mức an toàn là ATM1, bằng 2 xác định ATM2).

- Output: Bộ tham số miền (Fp, A,B,G,N,h, SEED)

1. Dựa vào y và biến xác định mức an toàn atm để tính các giá trị cụ thể của độ dài khóa N và modulo p theo tiêu chuẩn EC2 và EC3
2. Sử dụng một thuật toán tất định để sinh số nguyên tố p có thể chứng minh được theo độ dài đã xác định trong bước (1). Lưu lại p
3. Sử dụng Thuật toán 5 để sinh đường cong ngẫu nhiên có thể kiểm tra được trong Fp. Lưu lại SEED, A, B.
4. Sử dụng Thuật toán 7 để tính số điểm của đường cong được sinh ngẫu nhiên: N=#E(p).
5. Nếu N là hợp số thì quay lại bước (3).
6. B1: Kiểm tra tiêu chuẩn EC6 về ước nguyên tố của N± 1, nếu không thỏa mãn thì quay về bước (3).

   B2: Kiểm tra tiêu chuẩn EC4 về đường cong bất quy tắc: Nếu N = p thì quay về bước (3).

   B3: Kiểm tra tiêu chuẩn EC5 về điều kiện MOV, nếu không thỏa mãn thì quay về bước (3).
7. h = 1
8. Sinh ngẫu nhiên một điểm cơ sở G(x_G , y_G) ≠ ∞ .
9. Kiểm tra điểm cơ sở theo tiêu chuẩn EC7(x_G ≠ 0,3 x_{G_2} ≠ 0(mod p),5x_{G_4}+ 2 A x_{G_2} - 4 Bx_G + A_2 ≠ 0(mod p) .Nếu không thỏa mãn thì quay về bước (8).
10. Trả về (p, A, B, G, N, h, SEED)[4].

3.8.2 Sinh và kiểm tra cặp khóa đường cong Elliptic

Thuật toán : Sinh cặp khóa cho hệ mật Elliptic

+ Input: Bộ tham số miền(Fp, A,B,G,N,h, SEED)
+ Output: Output: (Q – điểm công khai, d – khóa bí mật)

1. Sinh $d \in R$ [0, N -1]. Số nguyên d phải được giữ bí mật và phải không dự đoán được
2. Tính điểm Q =(x_Q , y_Q)= dG
3. Trả về cặp khóa là (Q,d) trong đó Q là khóa công khai, d là khóa bí mật,

Với một bộ tham số miền (F_p , A, B , G , N , h, SEED) và một khóa công khai Q có thể được kiểm tra tính hợp lệ theo thuật toán dưới đây:

Thuật toán: Kiểm tra tính hợp lệ của khóa công khai

+ Input: Tham số miền (F_p , A, B , G , N , h, SEED), khóa công khai Q
+ Output: "*Khóa công khai hợp lệ*" hoặc "*Khóa công khai không hợp lệ*"

1. Kiểm tra Q không phải là điểm trên E
2. Kiểm tra x_q,y_q ∈ F_p
3. Kiểm tra rằng $y_q^2 = x_q^3 + Ax_q + B$ trong F_p
4. Kiểm tra N_q=∞
5. Nếu bất kỳ một trong các phép kiểm tra trên thất bại trả về "khóa công khai không hợp lệ" còn không thì trả về "khóa công khai hợp lệ"[4].

3.8.3 Thuật toán kiểm tra điều kiện MOV

Thuật toán:

+ Input: Giá trị B là cận của MOV theo tiêu chuẩn EC5

+ Output: 0: Không thỏa mãn điều kiện MOV; 1: Thỏa mãn MOV.

1. t = 1, ok= 1;
2. for i = 1 to B do
   T = t.p (modN)
   If (t==1){ok=0; return ok;}
3. Return ok;

3.8.4 Thuật toán sinh đường cong ngẫu nhiên

Thuật toán : Sinh đường cong ngẫu nhiên

+ Input: Số nguyên tố p

+ Output: Chuỗi SEED và A,B∈ F_p Xác định E trên F_p

Tính trước t = $[\log_2 p\ ]$, s= [(t-1)/256], h= t -256s

1) Chọn một chuỗi bit SEED có độ dài ít nhất 256 bit. Gọi G là độ dài theo bit của SEED

2) Tính H=SHA256(SEED), gọi c_o là h bit bên phải của H

3) W_o là h bit nhận được bởi việc thiết lập bit ngoài cùng bên trái của c_o thành 0 (nhằm đảm bảo r<p)

4) Với i=1 đến s tính W_i=SHA256((SEED+i)mod 2^g)

5) W = $W_o||W_1||....||W_s$

6) Với $w_1w_2...w_t$ là các bit của W từ trái qua phải. Tính số nguyên r=$\sum_{i=1}^{t} w_i 2^{i-1}$

7) Chọn A,B ∈ F_p sao cho $rB^2 \equiv A^3 \pmod p$ (A,B không nhất thiết phải chọn ngẫu nhiên)

8) Nếu $4A^3 + 27B^2 \equiv 0 \pmod p$, chuyển sang bước 1

9) Đường cong được chọn trên F_p là E : $y^2 = x^3 + Ax + B$

10)Trả về (SEED,A,B)[4].

CHƯƠNG 4 ỨNG DỤNG CHỮ KÝ SỐ TRÊN ĐƯỜNG CONG ELLIPTIC NHẰM ĐẢM BẢO ATTT TRONG ĐĂNG KÝ THẺ TRỰC TUYẾN

4.1 Bài toán

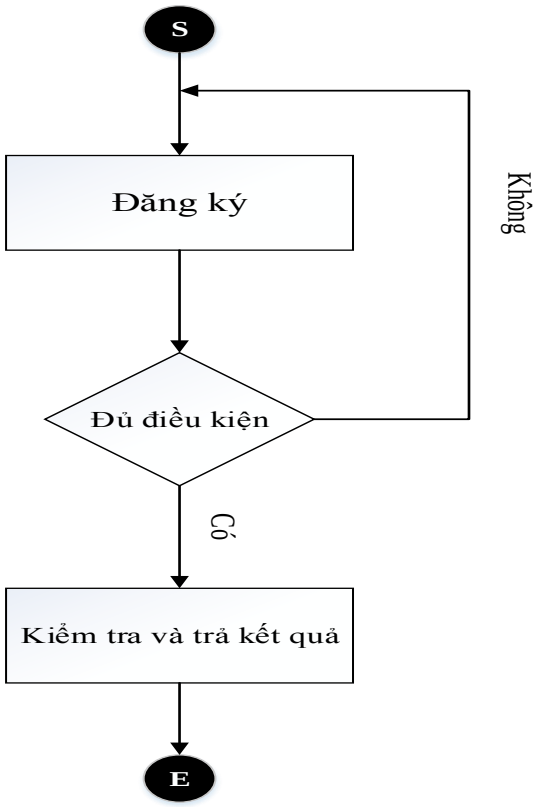
Việc phát triển thanh toán, mua hàng online bằng thẻ Ngân hàng là xu hướng phát triển tất yếu tại thị trường việt nam hiện nay. Phát triển thanh toán bằng thẻ online cũng là sự đầu tư đúng đắn theo chủ trương hạn chế giao dịch bằng tiền mặt của Ngân hàng Nhà nước. Thêm vào đó, sự đầu tư và cam kết về chất lượng của nhiều ngân hàng sẽ là điểm dựa đáng tin cậy cho các đối tượng khách hàng sử dụng thanh toán online. Tối ưu hóa mọi giao dịch với chiếc thẻ ngân hàng và cú nhấp chuột để tận hưởng cuộc sống một cách đơn giản, tiện lợi và thoải mái nhất.

Để có được những lợi ích mang lại của thẻ thông minh thì người dùng phải đăng ký để sỡ hữu được thẻ thông minh cho riêng mình. Người dùng chuẩn bị giấy tờ như chứng minh thư, ảnh, thông tin chứng minh thu nhập, mang giấy tờ đến tại chi nhánh ngân hàng muốn đăng ký mở thẻ. Sau khi ngân hàng xác nhận thông tin hợp lệ sẽ tiến hành cấp thẻ sau năm đến bẩy ngày làm việc. Thủ tục đăng ký thẻ rườm rà, mất thời gian. Người dùng cũng phải mất công chạy đi chạy lại ngân hàng để tiến hành làm thủ tục đăng kí, thời gian đăng ký cũng hạn chế trong giờ hành chính gây bất tiện cho người đăng ký thẻ mới. Ngoài ra thời gian chờ đợi thẻ cũng mất 7 ngày và phải lên đúng chi nhánh nơi mình đã đăng ký để nhận thẻ. Hiện nay có một số ngân hàng có hình thức đăng ký thẻ trực tuyến tuy nhiên việc bảo mật thông tin cho khách hàng đang còn lỏng lẻo dẫn đến mất an toàn thông tin. Vấn đề đặt ra phải đảm bảo An toàn thông tin trong giao dịch trực tuyến và đăng ký thẻ. Cần đưa các hệ mật an toàn vào quá trình mã hóa, giải mã, cũng như chứng thực chứng từ liên quan trong quá trình đăng ký cũng như giao dịch trên mạng Internet không an toàn. Hệ mật dựa trên đường cong Elliptic được đánh giá là hệ mật có độ bảo mật an toàn cao và hiệu quả hơn nhiều so với hệ mật công khai khác. Do đó trong phạm vi luận văn này đề xuất áp dụng hệ mật trên đường cong Elliptic trong quá trình đăng ký thẻ tín dụng trực tuyến.

4.2 Giải pháp kết hợp chữ ký ECDSA trong đăng ký thẻ trực tuyến

4.2.1 Quy trình đăng ký thẻ trực tuyến

Trong hệ thống đăng ký thẻ trực tuyến, người dùng phải thực hiện các bước bao gồm: đăng ký thẻ; ngân hàng thực hiện các bước: kiểm tra và trả kết quả. Quy trình được mô tả như hình:



Hình 4.1 Quy trình đăng ký thẻ trực tuyến.

Đăng ký: Trong bước này người dùng tiến hành đăng ký các thông tin trên hệ thống đồng thời cung cấp các giấy tờ liên quan.

Kiểm tra và trả kết quả: Ngân hàng xác minh tính hợp lệ của tờ khai và các giấy tờ cung cấp sau đó sẽ trả kết quả.

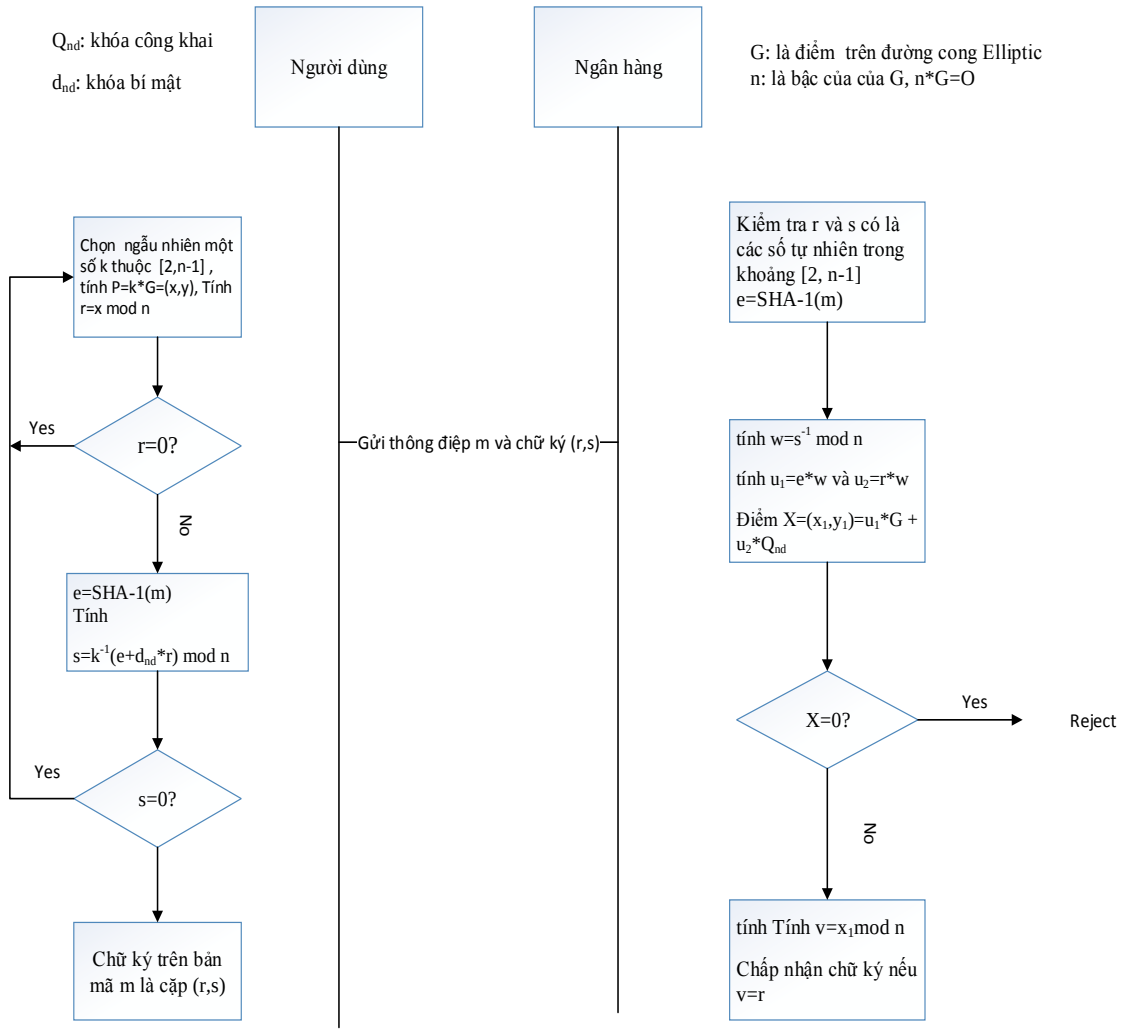
4.2.2 Chữ ký ECDSA dùng trong đăng ký thẻ trực tuyến.

Để bảo mật thông tin cho khách hàng khi đăng ký thẻ trực tuyến chúng ta sẽ ứng dụng chữ ký ECDSA vào quá trình đăng ký thẻ trực tuyến để đảm bảo rằng chính người ký là người tạo ra nó, không thể làm giả chữ ký nếu như không biết thông tin bí mật để tạo chữ ký, một khi đã ký thì người ký không thể phủ nhận chữ ký đó.

Lý do chọn chữ ký ECDSA bởi vì ưu điểm độ an toàn của nó, độ an toàn của chữ ký ECDSA dựa trên bài toán logarit rời rạc trên đường cong elliptic. Cho đến nay độ an toàn của các hệ mã hoá đường cong elliptic đã được chỉ ra là rất an toàn và hiệu quả. Đối với bài toán logarit rời rạc đường cong elliptic thì có nhiều thuật toán giải nó. Tuy nhiên chưa có thuật toán nào có độ phức tạp tính toán trong thời gian đa thức. Thuật toán giải bài toán logarit rời rạc đường cong elliptic tốt nhất hiện nay là thuật

toán Pollard's Rho, phiên bản thiết kế theo hướng tính toán song song. Theo đó với nhóm đường cong elliptic cấp n và có r máy tính cùng tính toán thì phải mất $\sqrt{\pi.n}$ /2.r phép toán. Mặt khác người ta đã phân tích và chỉ ra rằng với hệ mã hoá dựa trên bài toán logarit rời rạc đường cong elliptic có cùng độ bảo mật với hệ mã hoá dựa trên bài toán phân tích số nguyên thành các thừa số nguyên tố (như RSA) thì độ dài khoá của hệ mã hoá dựa trên đường cong elliptic có chiều dài khoá ngắn hơn rất nhiều . Chẳng hạn với hệ mã hoá RSA có chiều dài khoá là 1024 bit thì hệ mã hoá bằng đường cong elliptic chỉ cần độ dài khoá 163 bit sẽ có độ bảo mật tương đương. Và do đó việc tính toán các tiến trình đối với các hệ mã hoá đường cong elliptic là nhanh hơn rất nhiều.

Sơ đồ khối chữ ký ECDSA:



Hình 4.2 Sơ đồ thuật toán chữ ký số ECDSA

- Các bước thực hiện:

a. Sinh khóa

1. Chọn số ngẫu nhiên d trong khoảng [2, n-1] làm khóa bí mật

2. Tính Q=dG làm khóa công khai.

b. Ký trên bản rõ m
   1. Chọn một số ngẫu nhiên k, $2 \leq k \leq n-1$
   2. Tính $kG = (x_1, y_1)$.
   3. Tính $r = x_1 \bmod n$. Nếu r=0, quay lại bước 1.
   4. Tính $k^{-1} \bmod n$
   5. Tính $s = k^{-1}(m + dr) \bmod n$. Nếu s = 0, quay lại 1.
   6. Chữ ký trên thông điệp m là (r,s)

c. Kiểm tra chữ ký
   1. Kiểm tra r và s có là các số tự nhiên trong khoảng [2, n-1] không.
   2. Tính $w = s^{-1} \bmod n$
   3. Tính $u_1 = mw \bmod n$ và $u_2 = rw \bmod n$.
   4. Tính $X = u_1G + u_2Q = (x_x, y_y)$
   5. Nếu X = O thì phủ nhận chữ ký . Ngược lại tính $v = x_x \bmod n$
   6. Chữ ký chỉ được chấp nhận nếu v = 4

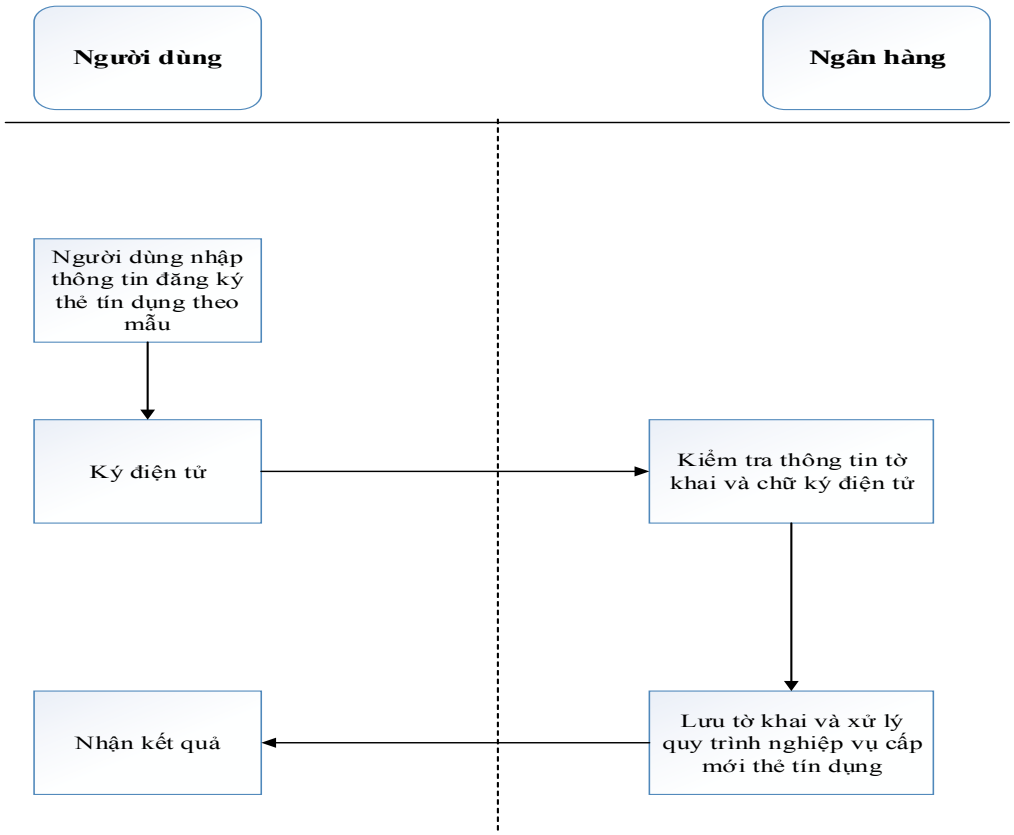
d. Xác thực

   Nếu chữ ký (r,s) trên m là đúng thì $s = [k^{-1}(m + dr)] \bmod n$.

   $k \equiv s^{-1}(m + dr) \equiv s^{-1}m + s^{-1}rd \equiv wm + wrd \equiv u_1 + u_2d \pmod n$.

   Vì vậy, $u_1G + u_2G = (u_1 + u_2d)G = kG$ và vì vậy v = r.

Quy trình ký và kiểm tra tính toàn vẹn của đăng ký thẻ trực tuyến:



Hình 4.3 Quy trình đăng ký thẻ trực tuyến sử dụng chữ ký điện tử

Bước 1. Bên ngân hàng tạo mẫu khai đăng ký thẻ và gửi cho khách hàng.

Bước 2. Người dùng khai báo trên mẫu khai đăng ký thẻ và sử dụng chữ ký số để ký lên tờ khai.

Bước 3. Để đảm bảo bí mật nội dung hợp đồng và chữ ký số, bên gửi tiến hành mã hóa cả mẫu đăng ký thẻ và chữ ký số vừa tạo bằng khóa công khai của bên nhận.

Sau đó mẫu đăng ký thẻ và chữ ký số đã được mã hóa qua Internet đến người nhận (ngân hàng). Bước này được gọi là "gói phong bì số".

Bước 4. Người nhận (ngân hàng) tiến hành "mở phong bì số" bằng cách sử dụng khóa bí mật của mình để giải mã thông điệp nhận được. Bước này đảm bảo chỉ duy nhất người nhận có thể nhận được thông điệp và chữ ký số của người gửi. Khi đó người nhận sẽ có trong tay bản đăng ký thẻ và chữ ký số của người gửi. Tiếp theo người nhận tiến hành xác thực tính toàn vẹn nội dung của hợp đồng và chữ ký số.

Bước 5. Người nhận (ngân hàng) tiến hành giải mã chữ ký số bằng khóa công khai của người gửi.

Bước 6. Người nhận (ngân hàng) tiến hành so sánh hai bản rút gọn này, nếu giống nhau chứng tỏ sự toàn vẹn của hợp đồng và chữ ký số đúng là của người gửi.

Nếu có sự khác biệt chứng tỏ đã có sự thay đổi nội dung hợp đồng hoặc chữ ký số.

4.2.3 Thiết kế chương trình

Bước 1: Ngân hàng sẽ gửi một bản đăng ký cho người dùng bao gồm các thông tin:

GIẤY ĐỀ NGHỊ
CẤP THẺ TÍN DỤNG
APPLICATION FORM

Số/No.: [Số đơn cấp thẻ chính:.....]
[Số đơn cấp thẻ phụ:.....]

Kính gửi: Ngân hàng .....

Đề nghị Ngân hàng phát hành cho tôi/chúng tôi thẻ tín dụng với các thông tin sau:

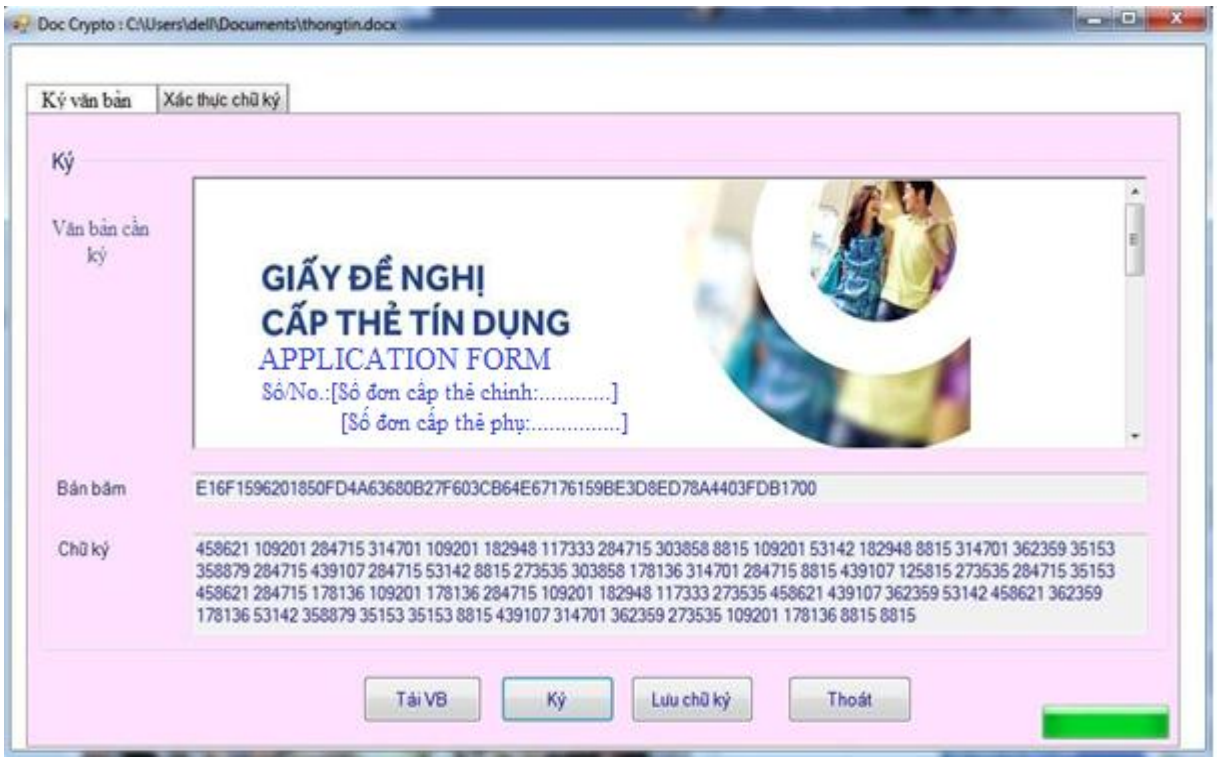
I. THÔNG TIN CÁ NHÂN (CARDHOLDER'S INFORMATION)
Quý khách vui lòng điền đầy đủ thông tin trong mục này (Please complete all fields)

Thông tin cá nhân (Personal information)	Chủ thẻ chính (Primary cardholder)	Chủ thẻ phụ (Supplementary cardholder)
Họ và tên (Fullname)	☐ Ông (Mr.) ☐ Bà (Ms./Mrs)	☐ Ông (Mr.) ☐ Bà (Ms./Mrs) .....
Ngày/ tháng/năm sinh (Date of birth dd/mm/yy)	.....	.....
Nơi sinh (Country of birth)	.....	.....

Hình 4.4 Mẫu tờ khai đăng ký thẻ trực tuyến

Khách hàng sẽ thực thiện kê khai thông tin và cung cấp giấy tờ đi kèm.

Bước 2: Tiến hành mã hóa rồi ký điện tử văn bản và gửi đến ngân hàng



Hình 4.5 Demo ký văn bản

Bước 3: Ngân hàng sẽ tiến hành giải mã và xác thực chữ ký. Kiểm tra thông tin trên mẫu khai. Nếu thông tin hợp lệ sẽ tiến hành các thủ tục đăng ký theo quy trình nghiệp vụ bên ngân hàng. Và gửi lại kết quả trong thời gian nhanh nhất qua bưu điện hoặc chuyển phát nhanh.

Như vậy quy trình đăng ký thẻ trực tuyến sử dụng chữ ký trên đường cong elliptic không chỉ tiết kiệm thời gian cho người dùng, tiết kiệm chi phí in ấn cho ngân hàng mà còn đảm bảo an toàn thông tin cho cả hai bên trong quá trình giao dịch.

KẾT LUẬN

Các kết quả đã đạt được

Thẻ thông minh đã và đang được phát triển mạnh mẽ không chỉ trên thế giới mà tại việt nam thẻ thông minh cũng đang ngày càng sôi động, hứa hẹn tạo một bước ngoặt mới cho thị trường thẻ với những ứng dụng và tiện ích vô cùng độc đáo. Ngoài các cơ hội là những thách thức không hề nhỏ, đó chính là vấn đề bảo mật thông tin ngày nay đang đặt lên hàng đầu.

Trong khóa luận này tôi đã trình bày được những kết quả sau:

+ Giới thiệu tổng quan về thẻ thông minh: khái quát lịch sử phát triển của thẻ thông minh, nêu lên cấu tạo và phân loại thẻ. Phân tích chi tiết về ưu nhược điểm của thẻ thông minh, ngoài ra thách thức trong việc phát triển thẻ thông minh.

+ Nghiên cứu về công nghệ Java Card:

- Giới thiệu về JavaCard, kiến trúc của nó, tập ngôn ngữ.
- Trình bày về máy ảo để chạy, môi trường chạy, api java card, quy ước đặt tên, ứng dụng applet
- Trình bày về các giao thức truyền nhận dữ liệu giữa thẻ và thiết bị đầu cuối

+ Mật mã đường cong Elliptic

- Trình bày cơ sở lý thuyết đường cong Elliptic
- Mật mã đường cong
- Chữ ký số trên hệ mật đường cong elliptic.
- Đánh giá tấn công trên hệ mật elliptic
- Chuẩn tham số cho hệ mật
- Cách sinh tham số cho hệ mật

+ Ứng dụng chữ ký số trên đường cong Elliptic nhằm đảm bảo ATTT trong đăng ký thẻ trực tuyến.

- Sử dụng chữ ký điện tử trên hệ mật đường cong Elliptic
- Demo được chương trình.

HƯỚNG NGHIÊN CỨU TIẾP THEO

- Tìm hiểu cải tiến công nghệ JavaCard ứng dụng vào thẻ thông minh để nâng cao tính bảo mật cho thẻ.

TÀI LIỆU THAM KHẢO

Tài liệu tiếng Việt

[1] Trịnh Nhật Tiến, *Giáo trình an toàn dữ liệu*, NXB Đại học Quốc Gia, 2008.

[2] Đào Việt Anh, luận văn thạc sỹ Nghiện cứu một số chữ ký đặc biệt trên đường cong Elliptic, 2011.

[3] ThS. Hoàng Thị Vân Anh, *Thẻ thông minh xu hướng tiêu dùng mới tại Việt Nam,* Viện nghiên cứu thương mại.

[4] ThS. Hoàng Thị Xuân, *Nghiên cứu hệ mật trên đường cong Elliptic và ứng dụng,* luận văn thạc sĩ kỹ thuật -2013.

Tài liệu tiếng anh

[5] Alfred Menezes and Scott Vanstone, *Guide to Elliptic Curve Cryptography,2004*

[6] Zhiqun Chen, Java Card Technology for Smart cards, Architecture and Programming.

[7] Smart Card technology and application by Tina Chhabra& Piya (Ray) Chindaphorn

[8] Pachtchenko, Nadejda, Evaluating elliptic curve cryptography for use on java card

[9] Marc Witteman, Java Card Security, 2003