セキュリティプロトコルの変更が可能なマルチアプリケーションICカードシステムの開発と評価

内山　宏樹[†]　　梅澤　克之[††]　　洲崎　誠一[††]

Development and Evaluation of the System of Multi-Application Smart Card with Security Protocol Change Function

Hiroki UCHIYAMA[†], Katsuyuki UMEZAWA[††], and Seiichi SUSAKI[††]

あらまし　今後拡大するユビキタス環境においては，セキュリティ要件の変化やセキュリティプロトコルやアルゴリズムの脆弱性の発見に伴い，プロトコルの迅速な更新が求められることが予想される．このような要求を満たすために，ユーザが享受するサービスのセキュリティ要件，脆弱性情報，ネットワークやシステムの環境情報等から，動的にセキュリティプロトコルを生成し，そのプロトコルを様々なデバイス上に実装する技術の研究を実施した．本論文では，利用可能なリソースなどに制限があるICカードに適用可能なセキュリティプロトコルモジュールの動的生成方式の提案と，その方式を実現したシステムの評価結果を示す．

キーワード　セキュリティプロトコル，ICカード，カスタマイズ，ユビキタス

1. まえがき

近年，携帯電話，情報家電，ネットワーク対応ゲーム機等の普及に伴い，個人が利用する電子機器が様々な形で外部のネットワークと接続されるユビキタス環境がますます進展しつつある．このようなユビキタス環境では，多種多様な端末やネットワークが利用されるため，利用環境に応じた「安全」の提供が必要となる．例えば，電子決済のような高度なセキュリティが要求されるサービスや，IC乗車券のような迅速な認証が要求されるサービスなど，提供されるサービスによって様々なレベルのセキュリティが必要となる．また，実サービスの開始後にサービスが利用している認証方式等に脆弱性が発見される場合がある．例えば，SSLのセキュリティプロトコルが利用する暗号アルゴリズム（RSA暗号モード）において，脆弱性の発見に伴い仕様変更が実施された事例がある．このような背景から，セキュリティプロトコルを迅速に更新する技術の開発が急務となっている．

本論文では，CPU周波数がたかだか数MHzであり，メモリ容量が数十kByte程度しか搭載されていないICカードに適用可能なセキュリティプロトコルモジュールの動的生成方式の提案とその方式を実現したシステムの概要及び評価結果に関して示す．以下では，まず，**2.**で従来技術としてICカードの管理仕様やプロトコル実装方法について記述する．**3.**で提案するICカードのプロトコルカスタマイズ技術について記述し，**4.**と**5.**でその技術を実現する上で必要となる外部カスタマイズ方式とアクセス制御方式を記述する．そして**6.**で提案方式の評価を行う．最後に**7.**でむすびとしてまとめと今後の課題を示す．

2. 従来技術

2.1 マルチアプリケーションICカード

ICカードには，図1に示すようにネイティブICカードとマルチアプリケーションICカードの2種類のカードが存在する．ネイティブICカードとは，アプリケーションと一体化した専用OSを搭載したICカードであり，アプリケーションを書き換えることはできない．一方，マルチアプリケーションICカードとは，OSとアプリケーションが分離したマルチアプ

[†]（株）日立製作所 日立研究所，日立市
Hitachi Research Laboratory, Hitachi, Ltd., 1–1 Omika-cho 7-chome, Hitachi-shi, 319–1292 Japan
[††]（株）日立製作所 システム開発研究所，横浜市
Systems Development Laboratory, Hitachi, Ltd., 292 Yoshida-cho, Totsuka-ku, Yokohama-shi, 244–0817 Japan

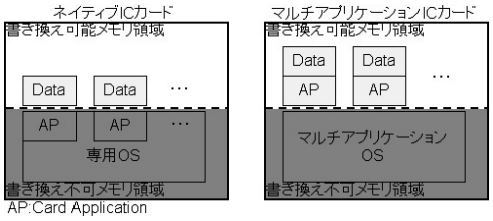

図 1　ネイティブ IC カードとマルチアプリケーション IC カード
Fig. 1　Native smart card and multi-application smart card.

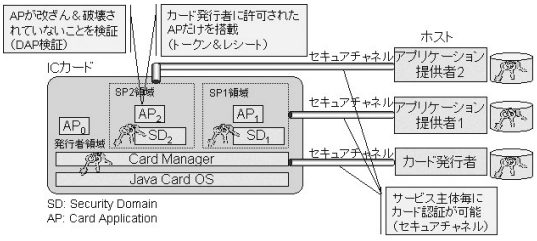

図 2　GlobalPlatform 仕様
Fig. 2　GlobalPlatform specification.

リケーション OS を搭載した IC カードであり，アプリケーションを書き換えることが可能である．また，汎用的なプログラム言語（C 言語，Java 等）でアプリケーションを開発可能であるというメリットもある [1].

2.2　IC カードの管理仕様

2.2.1　GlobalPlatform

GlobalPlatform とは，マルチアプリケーション IC カードの管理システムに関する業界標準化団体であり，マルチアプリケーション IC カードの管理仕様として，IC カードの状態管理や鍵管理等を安全に行うためのスキームを規定している [2]．ただし，GlobalPlatform 仕様単体では OS の機能を実現できないため，JavaCard [3], [4] OS などと連携して利用されている．GlobalPlatform 仕様では，図 2 に示すようにアプリケーション搭載時や削除時に IC カードにあらかじめ設定されている鍵を用いてホストとカード間で相互認証を行い，セキュアチャネルを構築する．また，アプリケーションの改ざん有無を検知する仕様も規定されている．

2.2.2　MULTOS

MULTOS とは，MasterCard が中心になって設立した MAOSCO で策定されたマルチアプリケーション IC カード OS の仕様のことを指す [5]．MULTOS 仕様は，図 3 に示すように GlobalPlatform 仕様と大きく異なり，アプリケーションのコードのハッシュ値を用いて生成するアプリケーション搭載証明書（ALC）やアプリケーション削除証明書（ADC）をカードに送付することによりホストの認証を行い，アプリケーションの搭載や削除を実現する．ALC や ADC は，鍵管理局（KMA）と呼ばれるカード発行者やアプリケーション提供者ではない第三者機関で生成される．

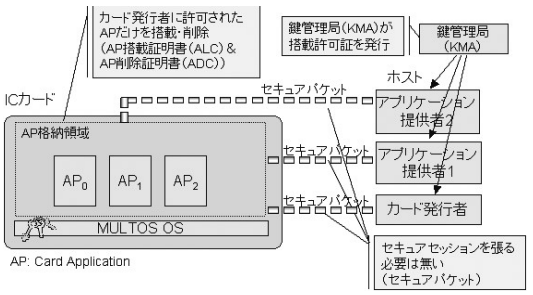

図 3　MULTOS 仕様
Fig. 3　MULTOS specification.

2.3　IC カードのプロトコル実装方法

従来，IC カードで利用するセキュリティプロトコルは以下の 2 通りの実装方法があった．

（1）　個別のアプリケーションに実装
（2）　カード OS 内に実装

（1）の方法は IC カード内の各アプリケーションに類似のプロトコル機能が実装されるため，アプリケーションの増加に伴い，IC カード内のリソースを無駄に消費してしまう可能性があった．一方，（2）の方法は，複数のアプリケーションから共通にプロトコル機能を利用可能であるため，リソースの節約になる．しかし，GlobalPlatform 仕様の場合，IC カード発行時に使用するプロトコルを決定するため，発行後にプロトコルを変更するためには IC カードの再発行が必要となる．また，変更可能なプロトコルはあらかじめ IC カードに搭載されているプロトコルに限定されており，サービスの種類等に応じてきめ細かにプロトコルを変更することは困難であった [2]．更に，MULTOS 仕様の場合にはプロトコルの変更そのものが想定されていないため，プロトコルを変更するためには，新たなプロトコルの設計から実施する必要があった [5].

また，**2.2** に示したように IC カードの管理スキームは一種類でないため，単一のホストで様々な IC カードに対してアクセスするためには外部システムを多重化する必要があった．

表 1　プロトコルカスタマイズ方式の比較
Table 1　Comparison of protocol customization method.

方式	外部カスタマイズ方式	内部カスタマイズ方式
対象	セキュリティプロトコルや暗号アルゴリズムの危殆化といった異常事態への対応	環境の変化（ユビキタス環境）への対応
利点	セキュリティプロトコルの大規模な変更が必要な場合に有効	環境パラメータに応じて，動的に，短時間でプロトコルの変更が必要な場合に有効
適用例	セキュリティプロトコルを従来想定されていないフロー形式へ変更する必要がある場合	モバイル環境，ETC 環境，ネットカフェ環境等のユビキタス環境において迅速なセキュリティプロトコルの変更が要求される場合
	ハードウェア実装されていない暗号アルゴリズムをソフトウェア実装によって追加する場合など大規模な変更が必要となる場合	ハードウェア実装されている暗号アルゴリズムの種類や鍵長の変更など比較的小規模な変更が必要となる場合

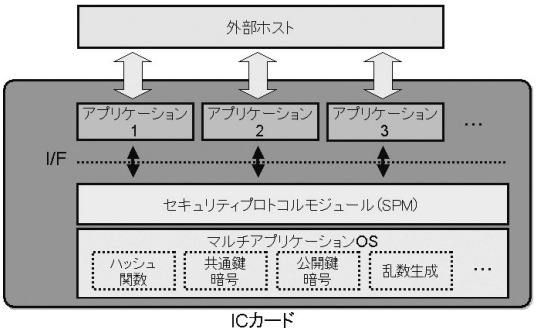

図 4　提案する IC カード内の構成
Fig. 4　Proposal architecture in the smart card.

3. IC カード向けセキュリティプロトコルカスタマイズ技術の提案

3.1　提案する IC カード内の構成

本論文で提案する IC カード内の構成を図 4 に示す．

認証プロトコル等のセキュリティプロトコル機能を提供するセキュリティプロトコルモジュール（以降，SPM と記述する）を IC カード内のアプリケーションと OS の間のミドルウェアとして配置した．SPM は OS に実装されている暗号アルゴリズム等を利用したセキュリティプロトコルの実行と，上位アプリケーションに対するアクセスインタフェースの提供を行う．上位アプリケーションは外部ホストから受信したセキュリティプロトコルコマンドを SPM に転送し，SPM 内で実行したセキュリティプロトコルの実行結果を外部ホストに送信し，認証や鍵交換を実現する．IC カード内をこのような構成にすることにより，セキュリティプロトコル機能を各アプリケーション内に実装する必要がなくなるため，IC カード内のメモリ使用量を削減することが可能になる．また，SPM の置き換えや動作変更を実施することにより，セキュリティプロトコルの動的なカスタマイズが実現可能となる．なお，提案方式を適用する際には図 4 の構成の GlobalPlatform カードや MULTOS カードを新規に発行する必要がある．GlobalPlatform 仕様にはアプリケーションが OS のプロトコル機能を利用して外部ホストと認証する仕様が存在するため，アプリケーションの修正コストは小さいと考えられるが，MULTOS 仕様の場合にはアプリケーションの構成自体を変更する必要があるため修正コストは大きくなる可能性がある．しかしながら，従来 GlobalPlatform 仕様と MULTOS 仕様で異なっていた IC カード内の構成を統一することにより，アプリケーションの管理コストは各々のスキームごとのアプリケーションを管理する場合に比べ小さくなることが予想される．よって，MULTOS 仕様においても図 4 の構成は有効となる可能性が高いと考えられる．

3.2　提案するカスタマイズ方式

SPM を用いて IC カードのセキュリティプロトコルをカスタマイズする方式として，表 1 に示す 2 種類の方式が考えられる．第 1 の方式は，IC カード外部でカスタマイズした SPM を安全に IC カードに入れ換える方式によるセキュリティプロトコルのカスタマイズ技術（以降，外部カスタマイズ方式と記述する）であり，第 2 の方式は，IC カード内部において SPM の動作を変更する方式によるセキュリティプロトコルのカスタマイズ技術（以降，内部カスタマイズ方式と記述する）である．表 1 に示すように両方式にはそれぞれ利点があるため，両者を組み合わせることにより，様々なサービスへの適用が可能となると考えられる．本論文では，暗号アルゴリズムの 2010 年問題(注1)によ

(注1)：米国では 2010 年までに政府標準暗号を次世代暗号アルゴリズムに移行することが決定している．対象となる暗号アルゴリズムは 2-Key Triple DES や RSA 1024 bit などであり，現在使用している IC カードの大半は影響を受けると予想されている．

り表面化すると考えられる IC カード内の暗号アルゴリズムの大規模な置換えに対応可能な外部カスタマイズ方式に着目し，外部カスタマイズ方式の実装方法及びカスタマイズに伴い増加する脆弱性への対策方法を検討した．また，外部カスタマイズ方式では IC カード内の SPM が動的に交換されるため，IC カード内の上位アプリケーション（以降，AP と記述する）が期待するセキュリティレベルを維持できるよう，SPM とそれを利用する AP との間の適切なアクセス制御技術（以降，アクセス制御方式と記述する）についても検討した．

以降の章では提案する二つの方式の詳細を示す．

4. 外部カスタマイズ方式

4.1 概　　要

図 5 に本研究で提案する外部カスタマイズ方式の概要を示す．

提案する外部カスタマイズ方式は，清本ら [6] により提案されたセキュリティプロトコルの自動生成技術が組み込まれたプロトコル自動生成サーバ及び太田ら [7] により提案されたセキュリティプロトコルの高速検証技術が組み込まれたプロトコル高速検証サーバと連携して動作するものである．具体的には図 5 の点線で囲まれた部分であり，コンパイラサーバ，店舗端末，IC カードから構成される．プロトコル自動生成サーバは店舗端末から，サービスの要件等を記載した「環境パラメータ」を取得し，そのパラメータに応じて IC カードの種類に依存しないプロトコル定義ファイル（XML 形式）を出力する．次に，プロトコル高速検証サーバにおいてプロトコル定義ファイルに定義されているプロトコルの安全性を検証し，XML 署名を付与する．次に，コンパイラサーバはプロトコル高速検証サーバから取得したプロトコル定義ファイルの署名を検証し，カード発行者（鍵管理局）サーバと連携して IC カードに搭載可能な格納モジュールを生成する．最後に，格納モジュールを各店舗端末に配信し，GlobalPlatform スキームや MULTOS スキームなど様々な種類の IC カードに搭載されている SPM と置き換える．このような方式を実現することにより，IC カードの管理スキームに依存せずにセキュリティプロトコルを動的にカスタマイズすることが可能となる．

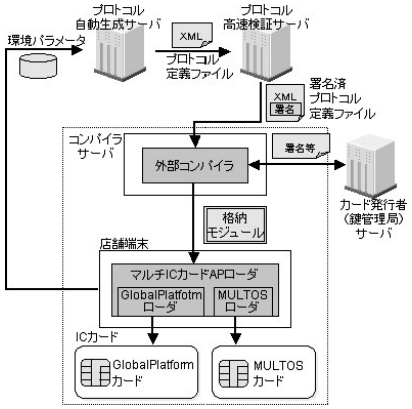

図 5　外部カスタマイズ方式の概要
Fig. 5　Outline of external customization method.

4.2 実装方式の検討

外部カスタマイズ方式を実現するために，図 5 に示す「外部コンパイラ」と「マルチ IC カード AP ローダ」を開発した．以降では各機能について記す．

4.2.1 外部コンパイラ

外部コンパイラとは，プロトコル定義ファイルから IC カードに格納可能なモジュールを生成する機能をもつものである．図 6 に外部コンパイラの概要を示す．まずはじめにプロトコル高速検証サーバより XML 署名が付与された安全性検証済のプロトコル定義ファイルを取得する．次に，取得したプロトコル定義ファイルの改ざん有無を検知するために XML 署名の検証を行い，GlobalPlatform スキームや MULTOS スキームに準拠したソースコードを出力する．次に，出力したソースコードのコンパイル，クラスファイルの生成，IC カードに搭載可能な実行モジュール形式への変換を行う．最後に生成した実行モジュールの改ざんを防止するためにカード発行者（鍵管理局）サーバに実行モジュールを送信し，各スキームに応じた改ざん防止署名や証明書を取得し，実行モジュールと組み合わせて格納モジュールとして出力する．このような手順を

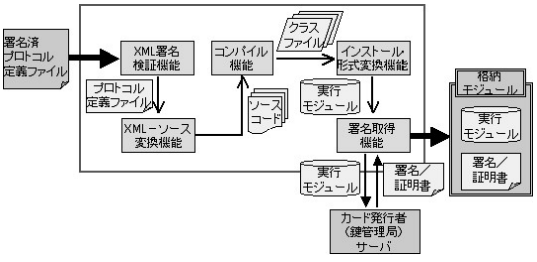

図 6　外部コンパイラ
Fig. 6　External compiler.

表 2　機能の相違点
Table 2　Differences of function.

項目	GlobalPlatform	MULTOS
AP 間機能共有方法	Sharable Interface Object	Delegation
selectingApplet	使用可	使用不可
select，deselect	あり	なし
getOutBlockSize	使用可	使用不可
checkCase	不要	必須

表 3　アルゴリズムの相違点
Table 3　Differences of algorithm.

項目	GlobalPlatform	MULTOS
PKCS1 Padding	実装済	未実装
ISO9797-1 Padding	実装済	未実装
DES Mac	実装済	未実装
SHA1withRSA	実装済	未実装

実施することにより，プロトコル定義ファイルの改ざんと実行モジュールの改ざんを防止することが可能となる．

XML-ソース変換機能においてソースコードを生成する際には，GlobalPlatform スキーム及び MULTOS スキームの様々な仕様の違いを考慮する必要がある．表 2 及び表 3 に機能やアルゴリズムに関する代表的な相違点を示す．

このような相違点に対し，機能に関しては各スキームの違いを吸収する形でソースコードを出力するように実装した．一方，アルゴリズムに関しては，MULTOS スキームには実装されていない PKCS1 パディングや DES Mac 等のアルゴリズムをソフトウェアで実現するソースを出力するようにし，SPM で使用可能なアルゴリズムの制限を極力なくすように実装した．

4.2.2　マルチ IC カード AP ローダ

マルチ IC カード AP ローダとは，外部コンパイラで生成した格納モジュールを IC カードに搭載する機能をもつのである．図 7 にマルチ IC カード AP ローダの概要を示す．まずはじめに店舗端末に挿入されている IC カードの種類（GlobalPlatform カード/MULTOS カード）を識別する．次に，外部コンパイラで生成した格納モジュールを取得する．最後に識別したカード情報をもとに搭載・削除コマンドを発行し，SPM の置換えを行う．

搭載･削除コマンドの発行にあたっては，GlobalPlatform スキーム及び MULTOS スキームの仕様の違いを考慮する必要があった．表 4 に搭載/削除に関する代表的な相違点を示す．

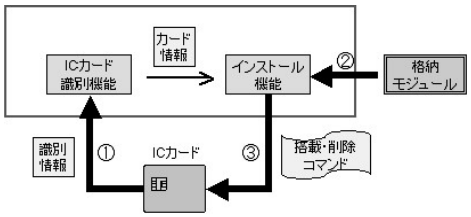

図 7　マルチ IC カード AP ローダ
Fig. 7　Application loader for multi-smart cards.

表 4　搭載/削除に関する相違点
Table 4　Differences of loading and deleting.

項目		GlobalPlatform	MULTOS
搭載	ファイル形式	CAP 形式	ALU 形式
	認証方式	セキュアチャネル	アプリケーション搭載証明書（ALC）
削除	認証方式	セキュアチャネル	アプリケーション削除証明書（ADC）

このような相違点に対して，それぞれのスキームに従った搭載・削除コマンドの生成や搭載・削除時の認証を実施するように実装した．これらのスキームに従った認証を実施することにより第三者による不正な SPM の改ざんや削除を防止することが可能となる．

5.　アクセス制御方式

5.1　概　　要

SPM のカスタマイズに伴い AP の安全性が低下することを防止するためにアクセス制御方式を提案する．図 8 にアクセス制御方式の概要を示す．

アクセス制御方式は，SPM 内に新たにアクセス制御機能を組み込むことにより実現する．アクセス制御機能とは，AP から SPM へのアクセス時に SPM が保持する安全性のレベル（SL_{SPM}）がアクセス元の AP が期待する安全性のレベル（SL_{AP1}〜SL_{APn}）を満足しているかを検証する機能である．検証の結果，満足していると判断された場合には，SPM へのアクセスを許可し，満足していないと判断された場合には，アクセスを拒否し処理を中止する．このような方式を実現することにより，プロトコルカスタマイズに伴う AP の安全性低下を防止することが可能となる．

5.2　実装方式の検討

アクセス制御方式を実現するためには，アクセス権の有無を判定する基準（セキュリティレベル）と判定アルゴリズムと実行タイミング及び実行方法に関して

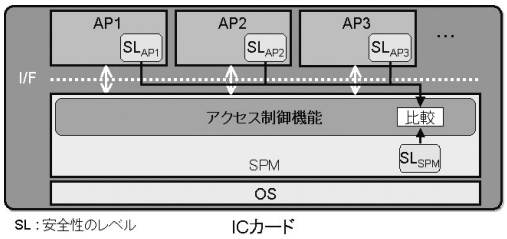

図 8　アクセス制御方式の概要
Fig. 8　Outline of access control method.

表 5　セキュリティレベル一覧
Table 5　List of security level.

項目	型	意味
entityAuthenticationAnalysys-Result		認証に関する解析結果
minimumKeyLength	int	鍵偽造確率を 2^{-k} としたときの k の値
formalResult	bool	認証プロトコル安全性
keyExchangeAnalysysResult		鍵共有に関する解析結果
knownKeyResult	bool	既知鍵攻撃安全性検証に関する解析結果
semanticResult	bool	意味論的安全性検証に関する解析結果
forwardSecrecyResult	bool	forwardSecrecy 検証に関する解析結果
formalResult	bool	鍵交換用プロトコル安全性

検討する必要がある．以降では各項目について記す．

5.2.1　セキュリティレベル

AP や SPM に設定するセキュリティレベルを表 5 に示す．これらの項目は清本ら [8] により提案されたプロトコルの安全性評価項目をもとに規定した．

セキュリティレベルは，認証プロトコルに関する評価項目と鍵交換プロトコルに関する評価項目から構成され，プロトコル自動生成サーバで生成するプロトコル定義ファイル中にその情報が記載される [6]．そこで **4.2.1** で示した外部コンパイラに対し，プロトコル定義ファイルに記載されているセキュリティレベルの取得機能，及び，SPM への格納機能を新たに追加した．これにより，セキュリティプロトコルのカスタマイズに伴い変化したセキュリティレベルを SPM に反映することが可能となる．

5.2.2　判定アルゴリズム

AP から SPM へのアクセス時に SPM のセキュリティレベルが AP が期待するセキュリティレベルを満足しているか否かを判定アルゴリズムを用いて判定する．実装した判定アルゴリズムを図 9 に示す．

APが期待するセキュリティレベル（SL_{AP}）

項目	型	値
minimumKeyLength	int	A
formalResult(entityAuth)	bool	B
knownKeyResult	bool	C
semanticResult	bool	D
forwardSecrecyResult	bool	E
formalResult(keyExchange)	bool	F

SPMのセキュリティレベル（SL_{SPM}）

項目	型	値
minimumKeyLength	int	a
formalResult(entityAuth)	bool	b
knownKeyResult	bool	c
semanticResult	bool	d
forwardSecrecyResult	bool	e
formalResult(keyExchange)	bool	f

① APが期待する鍵長以上であるか[A≦?a]
② APが期待する攻撃耐性の取得[B～Fのうち真のもの]
③ APが期待する攻撃耐性と一致しているか[B=?b など]

図 9　判定アルゴリズム
Fig. 9　Algorithm for verification.

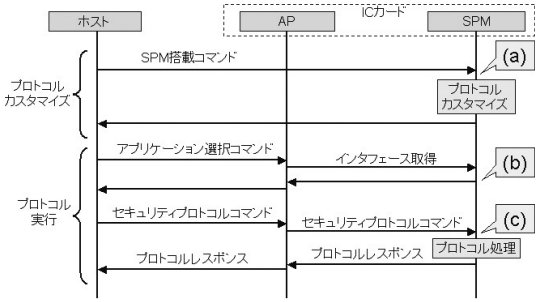

図 10　アクセス制御実行タイミング
Fig. 10　Timing for execution of access control.

まずはじめに，SPM のセキュリティレベルのうち，minimumKeyLength (注2) が AP の期待値以上であるか判定する．これは AP が期待する鍵長よりも SPM が実装する鍵長の方が長い場合には，AP の安全性を低下することにはつながらないと考えられるためである．次に，minimumKeyLength 以外の AP が期待するセキュリティレベルのうち，真である項目を抽出する．偽である項目は，SPM のセキュリティレベルによらず，常に条件を満たすことになるため，判定には利用しない．最後に，真である項目が SPM のセキュリティレベルと一致しているか検証する．なお，AP によって期待するセキュリティレベルが異なる場合には SPM のセキュリティレベルにより判定アルゴリズムの条件をクリアする AP は SPM に対してアクセス可能となる．

5.2.3　実行タイミング及び実行方法

図 10 に示すように，アクセス制御方式を実行するタイミングとしては，(a) プロトコルカスタマイズ時，(b) プロトコル実行時のアプリケーション選択コマン

(注2)：この項目は，認証プロトコルで利用する暗号アルゴリズムの鍵長を示すものであり，どのようなプロトコルにも必ず設定される項目である．

ド送信時，(c) プロトコル実行時のセキュリティプロトコルコマンド送信時の三つのケースが考えられる．

また，アクセス制御は図 10 に示すタイミングで毎回実施する方法（方法 1）と 1 回目のみアクセス制御を実施し，2 回目以降は 1 回目の結果を参照して判定する方法（方法 2）の 2 種類が考えられる．更に，方法 2 の場合には，アクセス制御結果を保管しておく主体として，“AP”，“SPM”，“ホスト” の 3 者が考えられる．これらの実行タイミングと実行方法と保管主体の組合せを網羅的に示した結果を表 6 に示す．

これらのパターンのうち，(1) については，プロトコルカスタマイズ時に処理した結果を保管しないため，AP から SPM へのアクセス時に参照することができず，アクセス制御を実現できない．また，(4)，(8)，(12) についてはホスト側であらゆる IC カードのアクセス制御結果を保管する必要があるため，現実的ではない．更に，(2)，(6)，(10) についてはアクセス制御結果を保管する機能を AP に追加する必要があるため，既に存在する AP の数を考えるとこのパターンも現実的ではない．よって，(3)，(5)，(7)，(9)，(11) が候補として残る．これらのパターンを処理時間の増加量や保管メモリ量で評価した結果を表 7 に示す．

この結果，(3)，(7)，(11) は AP の増加に伴い IC カード内で使用する保管メモリ量が増加することが判明した．よってメモリ容量に制限がある IC カードには不適切であると判断した．最後に，(5) と (9) のパターンを比較した結果，セキュリティプロトコルのフロー数に拠らず処理時間が一定となる (5) のパターンが最も適切であると判断した．

表 6　アクセス制御実行パターン
Table 6　Execution pattern of access control.

実行タイミング	方法 1	方法 2		
		AP	SPM	ホスト
(a)	(1)	(2)	(3)	(4)
(b)	(5)	(6)	(7)	(8)
(c)	(9)	(10)	(11)	(12)

表 7　アクセス制御パターンの比較
Table 7　Comparison of access control pattern.

パターン	処理時間の増加量		保管メモリ量
	カスタマイズ時	実行時	
(3)	$T_a \times N_{AP}$	—	$M \times N_{AP}$
(5)	—	$T_a \times N_p$	—
(7)	—	T_a	$M \times N_{AP}$
(9)	—	$T_a \times N_p \times N_f$	—
(11)	—	T_a	$M \times N_{AP}$

T_a　：1AP 当り必要となるアクセス制御実行処理時間
N_{AP}　：IC カード内に搭載されている AP の数
M　：1AP 当り必要となる保管メモリ量
N_p　：セキュリティプロトコルの実行回数
N_f　：セキュリティプロトコル内のフロー数

6. 評　　価

本章では，外部カスタマイズ方式及びアクセス制御方式を実装したシステムの性能評価を行う．

6.1　評価システム概要

評価に使用したシステムの概要を図 11 に示す．提案方式の有効性を評価するために，外部コンパイラを組み込んだコンパイラサーバとマルチ IC カード AP ローダ及びホスト機能を組み込んだ店舗端末を構築し，(1) コンパイラサーバにおける実行モジュール生成時間，(2) 生成された実行モジュールの容量，(3) プロトコル実行処理時間の 3 点に着目して評価を実施した．なお，今回はプロトコル高速検証サーバやカード発行者サーバとの連携は行わず，あらかじめコンパイラサーバ内に安全性検証済のプロトコル定義ファイルが格納されていることを前提とした．また，表 8 に図 11 に示したコンパイラサーバと店舗端末（ホスト）と IC カードの仕様を示す．

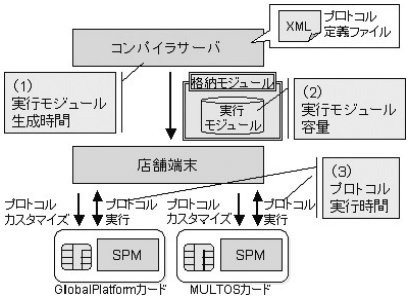

図 11　評価システム概要
Fig. 11　Outline of evaluation system.

表 8　評 価 環 境
Table 8　Evaluation environment.

項目	ホスト	IC カード	
		GlobalPlatform	MULTOS
OS	Windows XP SP2	JavaCard 2.1.1 GlobalPlatform 2.1	MULTOS 4.2
CPU	PentiumM 1.73 GHz	AE46C1	AE45C1
HDD	40 GByte	-	-
EEPROM	-	60 kByte	36 kByte
RAM	480 MByte	2.7 kByte	1.3 kByte

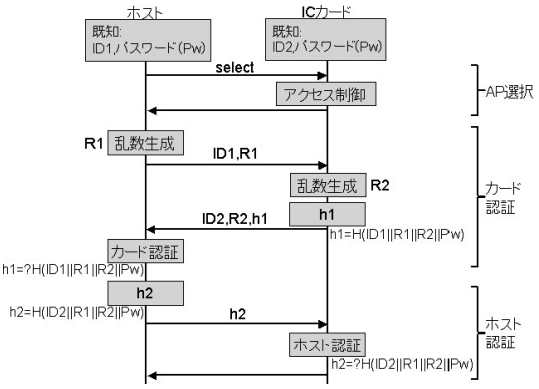

図 12　評価フロー
Fig. 12　Evaluation flow.

表 9　実行モジュール生成時間評価結果
Table 9　Evaluation result for generating modules.

スキーム	従来方式	提案方式
MULTOS	29.0 hours（254 ステップ）	1.39 s
GlobalPlatform	27.5 hours（241 ステップ）	2.04 s

6.2　評価フロー

性能評価には，図 12 に示すチャレンジアンドレスポンス認証方式のフローを利用した．本フローは，大きく三つの部分から構成される．一つ目は AP 選択フローであり，提案方式ではこの際にアクセス制御機能を実行する．二つ目はカード認証フローであり，IC カードからホストに送信する認証情報を基にホスト内で IC カード認証を行う．最後のフローはホスト認証フローであり，ホストから IC カードに送信する認証情報を基に IC カード内でホスト認証を行う．

6.3　評 価 結 果

6.3.1　実行モジュール生成時間

図 12 のフローを実現する SPM を従来のように人手により AP 内部に実装した場合（以降，従来方式と記述する）と，提案する外部カスタマイズ方式を利用して実装した場合（以降，提案方式と記述する）の実行モジュール生成時間を評価した．評価結果を表 9 に示す．なお，提案方式の測定は 10 回行い，表にはその平均値を記載している．また，人手により実行モジュールを生成する時間は，プログラムのステップ数を用いて以下の前提から算出した．

- 1400 ステップ/月 [9]
- 20 日/月
- 8 時間/日

この結果，提案方式は従来方式に比べ実行モジュールの生成時間を大幅に短縮できていることが確認できた．よって外部カスタマイズ方式が IC カードのプロトコルの動的なカスタマイズに有用であることが示されたと考えられる．

表 10　実行モジュール容量評価結果
Table 10　Evaluation result for capacity of module.

スキーム	従来方式	提案方式	増加率
MULTOS	2531 Byte	5138 Byte	203%
GlobalPlatform	3407 Byte	4025 Byte	118%

表 11　処理時間評価結果
Table 11　Evaluation result for processing time.

スキーム	処理項目		従来方式	提案方式	増減量
MULTOS	AP 選択		145 ms	213 ms	68 ms
	カード認証	受信	255 ms	386 ms	131 ms
		演算	114 ms	230 ms	116 ms
		送信	164 ms	249 ms	85 ms
	ホスト認証	受信	151 ms	304 ms	153 ms
		演算	105 ms	112 ms	7 ms
		送信	4 ms	19 ms	15 ms
	計		938 ms	1513 ms	575 ms
Global Platform	AP 選択		403 ms	534 ms	131 ms
	カード認証	受信	104 ms	191 ms	87 ms
		演算	98 ms	127 ms	29 ms
		送信	31 ms	85 ms	54 ms
	ホスト認証	受信	79 ms	152 ms	73 ms
		演算	75 ms	69 ms	−6 ms
		送信	2 ms	10 ms	8 ms
	計		792 ms	1168 ms	376 ms

6.3.2　実行モジュール容量

従来方式と提案方式の実行モジュールの容量を評価した．評価結果を表 10 に示す．

この結果，提案方式は実行モジュールの自動生成に伴い 1.2〜2 倍程度容量が増加することが判明した．しかし提案方式では SPM を複数の AP から利用可能であるため，従来方式の AP を同一の IC カード内に三つ以上搭載する場合には提案方式の方が使用するリソース量が少なくて済み，有用であるということができる．

6.3.3　プロトコル実行時間

従来方式と提案方式のプロトコル実行時間を評価した．評価結果を表 11 に示す．なお，従来方式，提案方式ともに測定は 10 回行い，表にはその平均値を記載している．また，表中の受信，演算，送信とは IC カード内部におけるデータの受信処理，アルゴリズム演算処理，データの送信処理を示す．

この結果，外部カスタマイズ方式を用いてセキュリ

ティプロトコルの動的なカスタマイズを実現し，かつ，アクセス制御方式を用いてプロトコルカスタマイズに伴うサービスのセキュリティレベルの低下を防止した場合でも，処理時間の増加量は従来方式に比べ 350〜600 ms 程度であり，十分実用に耐えるレベルであることが確認できた．

7. むすび

本論文では，IC カードのプロトコルカスタマイズ技術として外部カスタマイズ方式とアクセス制御方式について記述した．提案方式を適用することにより，SPM の動的なカスタマイズとカスタマイズに伴うサービスのセキュリティレベルの低下を防止することが可能となった．また，提案方式を用いた実行モジュールの生成時間，容量，処理時間を評価した結果，提案方式はマルチアプリケーション IC カードにおいて十分実用的であることが確認できた．

今後は，プロトコル自動生成サーバやプロトコル高速検証サーバとの連携方式や IC カード内部でプロトコルカスタマイズを実現する内部カスタマイズ方式に関して検討を行う予定である．

謝辞 本研究は，独立行政法人情報通信研究機構（NICT）の委託研究「ユビキタスネットワークにおける環境に応じたセキュリティプロトコルの自動生成・カスタマイズ技術に関する研究」の一環として行われた．

文　献

[1] 森山明子，“知っておきたい IC カードの知識 第 2 回 IC カードのソフトウェア，” 月刊バーコード，vol.2002, no.4, pp.47–51, April 2002.
[2] GlobalPlatform Card Specification Version 2.2, GlobalPlatform, March 2006.
[3] Java Card 2.2.2 Runtime Environment Specification, Sun Microsystems, March 2006.
[4] Java Card 2.2.2 Virtual Machine Specification, Sun Microsystems, March 2006.
[5] “MULTOS カード発行ガイド，” マルトス推進協議会，March 2003.
[6] S. Kiyomoto, H. Ota, and T. Tanaka, “On-the-fly automatic generation of security protocols,” Proc. ICEIS 2008, INSTICC, June 2008.
[7] 太田陽基，清本晋作，田中俊昭，“セキュリティプロトコルの自動生成・カスタマイズ技術に関する研究開発 IV—認証・鍵交換プロトコルの安全性検証，” コンピュータセキュリティシンポジウム 2008 予稿集，pp.575–580, Oct. 2008.
[8] S. Kiyomoto, H. Ota, and T. Tanaka, “Design of an efficient security protocol analyzer,” International Journal of Computer Science and Network Security, vol.7, no.6, pp.74–87, June 2007.
[9] “ユーザ企業向け 2005 年ソフトウェアメトリックス調査活動概要，” 日本情報システム・ユーザ協会，April 2005.

（平成 20 年 10 月 31 日受付，21 年 2 月 23 日再受付）

内山　宏樹　（正員）

2003 京都大学大学院情報学研究科通信情報システム専攻修士課程了．同年（株）日立製作所システム開発研究所入所．現在，同社日立研究所勤務．情報セキュリティ技術，保全技術などの研究・開発に従事．情報処理学会会員．

梅澤　克之

1996 早稲田大学大学院理工学研究科機械工学専攻修士課程了．同年（株）日立製作所システム開発研究所入所．以来，分散オブジェクトシステム，モバイルセキュリティ技術，スマートカードセキュリティ技術などの研究・開発に従事．情報処理学会，電気学会各会員．博士（工学）．

洲崎　誠一

1991 横浜国大・電子情報卒．同年（株）日立製作所システム開発研究所入所．以来，情報セキュリティ技術の研究・開発に従事．1996 情報処理学会第 52 回全国大会優秀賞，2000 年度山下記念研究賞受賞．情報処理学会会員．博士（工学）．