論　文

ICカード内部でセキュリティプロトコルの変更を実現する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 that Enable to Change Security Protocol in the Smart Card

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として外部カスタマイズ方式を提案した．その結果，動的なセキュリティプロトコルのカスタマイズが実現可能である見通しを得たものの，カスタマイズには多大な処理時間が必要であることが判明した．本論文では，この課題を解決するために短時間で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．

このような課題に対し，筆者らは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の動的なカスタマイズ方式としてICカード外部でセキュリティプロトコルをカスタマイズする外部カスタマイズ方式を提案した．その結果，セキュリティプロトコルの動的なカスタマイズの実現見通しを得ることができたものの，カスタマイズには多大な処理時間が必要であることが判明した．

本論文では上記課題を解決するために，短時間で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カードの管理仕様及びプロトコルカスタマイズ技術について記述する．**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.** で提案方式の評価を行う．最後に **5.** でむすびとしてまとめと今後の課題を示す．

[†]（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

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

マルチアプリケーション IC カード用の代表的な管理スキームとして GlobalPlatform 仕様が挙げられる．GlobalPlatform 仕様とは，業界標準化団体である GlobalPlatform により規定されたマルチアプリケーション IC カード向けの管理仕様であり，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が定義されている [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．

2.3 IC カードのセキュリティプロトコルカスタマイズ技術

従来，IC カードで利用するセキュリティプロトコル(注1)をカスタマイズする技術として筆者らが提案する外部カスタマイズ方式 [5] があった．外部カスタマイズ方式で利用する IC カード内の構成を図 3 にその概要を図 4 に示す．

外部カスタマイズ方式は，IC カード内にセキュリティプロトコルモジュール（以降，SPM と記述する）

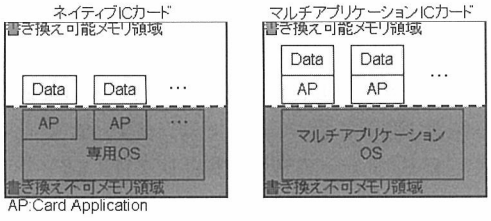

図 1 ネイティブ IC カードとマルチアプリケーション IC カード
Fig. 1 Native smart card and multi-application smart card.

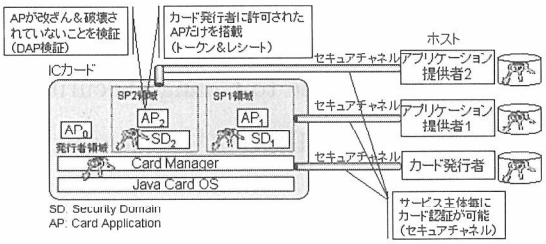

図 2 GlobalPlatform 仕様
Fig. 2 GlobalPlatform specification.

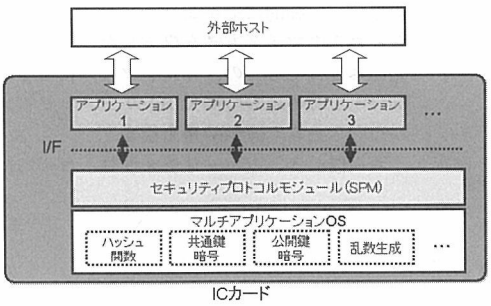

図 3 IC カード内の構成
Fig. 3 Architecture in the smart card.

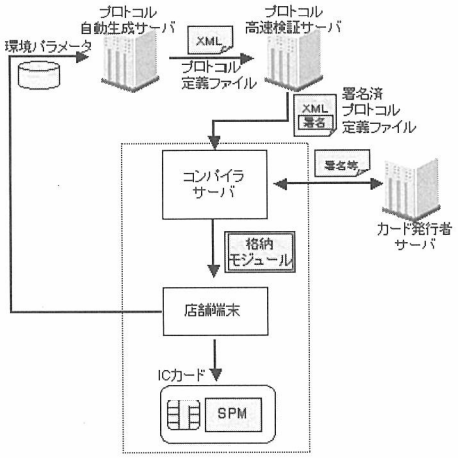

図 4 外部カスタマイズ方式の概要
Fig. 4 Outline of external customization method.

(注1)：セキュリティプロトコルとは，IC カードと外部ホストとの間で実行する認証プロトコルや鍵共有プロトコル等の IC カードの安全性を確保するために利用するプロトコルのことを指す．

と呼ばれるミドルウェアを搭載し，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 4 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，IC カードに搭載されている SPM と置き換える．このような方式により，IC カードの様々な管理スキームに依存することなく IC カードで使用するセキュリティプロトコルを安全にカスタマイズすることが可能となった．しかし，外部カスタマイズ方式では SPM そのものの削除や搭載の処理が必要となるため，IC カードが使用される環境やサービスに応じてセキュリティプロトコルを短時間でカスタマイズすることは困難であった．

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 提案方式

従来の外部カスタマイズ方式の課題を解決するために，IC カード内部で SPM の動作を短時間でカスタマイズす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」と呼ぶ）を提案する．既に提案済みの外部カスタマイズ方式と本論文で提案する内部カスタマイズ方式は表 1 に示すようにセキュリティプロトコルのカスタマイズ処理時間や実行処理時間に関して異なる特徴が存在するため，両方式を組み合わせることにより，両者の利点を生かした様々なサービスへの適用が可能となると考えられる．

以降の節では提案する内部カスタマイズ方式の詳細を示す．

3.2 内部カスタマイズ方式

3.2.1 概　　要

図 5 に内部カスタマイズ方式の概要を示す．

提案する内部カスタマイズ方式は，外部カスタマイ

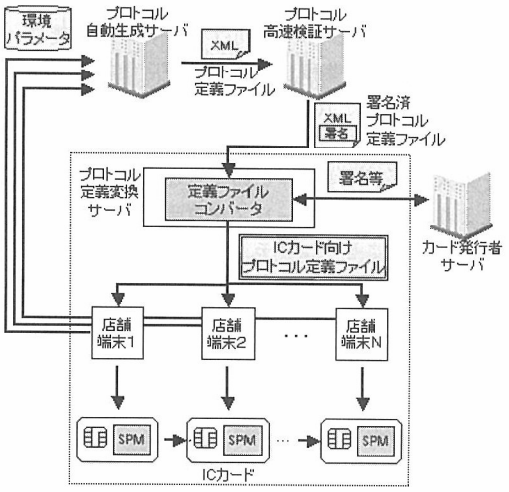

図 5　内部カスタマイズ方式の概要
Fig. 5　Outline of internal customization method.

表 1　セキュリティプロトコルカスタマイズ方式の比較
Table 1　Comparison of security 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
特徴	プロトコル定義ファイルを IC カード外部でコンパイルし，生成した SPM を IC カード内部の SPM と置き換えることによりカスタマイズを実現する	プロトコル定義ファイルを IC カード内部に格納されている SPM で逐次解釈して実行することによりカスタマイズを実現する
利点	セキュリティプロトコルカスタマイズ後の実行処理時間が要求される場合に有効	短時間でのセキュリティプロトコルのカスタマイズが要求される場合に有効
適用例	ETC 環境などカード挿入時と利用時のタイミングに違いがあり，カスタマイズ処理時間よりも実行処理時間が要求される場合	モバイル環境，ネットカフェ環境等の要件が異なる様々な環境でサービスを享受する場合
	内部カスタマイズ方式で利用する SPM そのものをカスタマイズする必要がある場合	電子マネーを用いた店舗決済やクレジット決済など，実行処理時間があまり要求されない場合

ズ方式と同様で清本ら [6] により提案さ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，プロトコル定義変換サーバ，コンビニエンスストアのレジ端末やクレジット決済端末などの店舗端末，IC カードから構成される．プロトコル自動生成サーバは，複数の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それぞれ取得し，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，プロトコル定義変換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，カード発行者サーバと連携して IC カード向けプロトコル定義ファイルに変換する．最後に，変換した IC カード向けプロトコル定義ファイルを各店舗端末に蓄積し，利用者が店舗端末でサービスを享受する際に IC カード向けプロトコル定義ファイルを IC カードに送信し，IC カード内の SPM の動作をカスタマイズする．このような方式を実現することにより，利用者がサービスを享受する環境の変化に応じて短時間にセキュリティプロトコルをカスタマイズすることが可能となる．以下では図 5 に示す「定義ファイルコンバータ」と「SPM」に関して詳細に示す．

3.2.2 定義ファイルコンバータ

(a) 機　　能

定義ファイルコンバータは，プロトコル定義ファイルを IC カード向けプロトコル定義ファイルに変換し，セキュリティプロトコルのカスタマイズに必要となるデータ列を生成するものである．

(b) 処理フロー

定義ファイルコンバータの処理フローを図 6 に示す．

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，プロトコル定義ファイル中のプロトコル定義群（データ定義，関数インスタンス定義，関数定義，フロー定義）を抽出する．次に，抽出したプロトコル定義群を表 2 に示す TLV 形式(注2)のデータ要素，関数インスタンス要素，関数要素，コマンド要素(注3)に変換してスクリプトファイルを生成する．最後に生成したスクリプトファイルの改ざんを防止するためにカード発行者サーバにスクリプトファイルを送信し，取得した署名と組み合わせて IC カード向けプロトコル定義ファイルとして出力する．このような手順を実現す

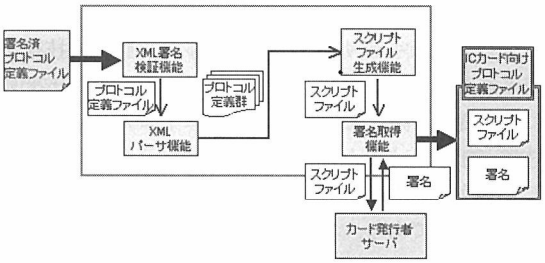

図 6　定義ファイルコンバータの処理フロー
Fig. 6　Processing flow of definition file converter.

表 2　スクリプトファイルの構成
Table 2　Contents of script file.

Tag	Name	Length	Value
04	データ要素	L_1L_2	データ情報群
41	データ情報	L_1L_2	データ ID，データサイズ，データ種類，グループ識別子
42	初期値	L_1L_2	初期値
43	データ取得元	L_1L_2	データ取得元
44	データ格納先	L_1L_2	データ格納先
03	関数インスタンス要素	L_1L_2	関数インスタンス情報群
31	関数インスタンス情報	L_1L_2	関数インスタンス ID，出力サイズ，関数 ID
02	関数要素	L_1L_2	関数情報群
21	関数情報	L_1L_2	関数 ID，鍵サイズ，引数サイズ，関数種類
22	鍵情報	L_1L_2	データ種類，データ ID
23	引数情報	L_1L_2	データ種類，データ ID
01	コマンド要素	L_1L_2	コマンド情報群
11	コマンド情報	L_1L_2	コマンド ID，受信コマンド ID，コマンド順序，受信サイズ，送信サイズ
12	受信情報	L_1L_2	データ引数群，データ処理群
15	データ引数	L_1L_2	データ種類，データ ID
16	データ処理	L_1L_2	データ種類，データ ID
13	送信情報	L_1L_2	データ引数群，データ処理群
15	データ引数	L_1L_2	データ種類，データ ID
16	データ処理	L_1L_2	データ種類，データ ID
14	検証情報	L_1L_2	データ種類，データ ID

(注2)：TLV (Tag Length Value) 形式とはタグと長さと値の三つのフィールドから構成される形式のことを示す．
(注3)：コマンド要素はホストから IC カードへのフロー定義と IC カードからホストへのフロー定義を組み合わせて生成する．

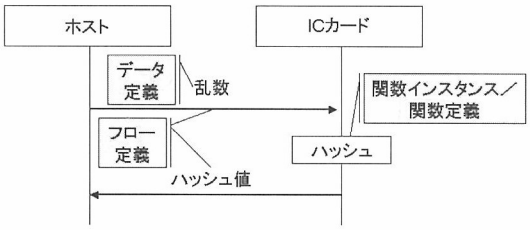

図 7　サンプルプロトコルフロー
Fig. 7　Flow of sample protocol.

ることにより，プロトコル定義ファイルやスクリプトファイルの改ざんを防止することが可能となる．

（c）　サンプルプロトコルを用いたスクリプトファイルの生成例

サンプルプロトコルを用いて，プロトコル定義ファイルをスクリプトファイルに変換する手順を示す．図 7 に使用するサンプルプロトコルを示す．

サンプルプロトコルはホストから IC カードに乱数を送信し，IC カード内で演算した乱数のハッシュ値を受信するという単純なプロトコルである．図 7 では「乱数」がデータ定義で定義され，「ハッシュ」が関数インスタンス及び関数定義で定義され，「ホストと IC カード間のフロー」がフロー定義で定義される(注4)．図 7 のプロトコルを XML 形式のプロトコル定義ファイルで記述した結果を図 8 に示す．dataList タグ，functionInstanceList タグ，functionList タグ，flowList タグにおいてデータ定義，関数インスタンス定義，関数定義，フロー定義がそれぞれ記述されている．なお，図中の数字はプロトコル定義ファイル内の重要な情報を示す箇所を識別するために付与している．

次に，図 8 に示すプロトコル定義ファイルを定義ファイルコンバータに入力し，TLV 形式のスクリプトファイルに変換した結果を図 9 に示す．図中には表 2 に対応する項目を文字で，図 8 に対応する項目を数字でそれぞれ記載している．例えばデータ要素の 7 バイト目の “0x01” は表 2 のタグ 41 のデータ ID を示し，図 8 の (1) を変換したものである．このようにスクリプトファイル内にはスクリプトファイルの構造を示すタグ情報とプロトコル定義ファイルに記載されているすべての情報が含まれている．

以上のような変換を行うことにより，プロトコル定義ファイルを IC カードで受信可能なデータ形式に変換することが可能となる．

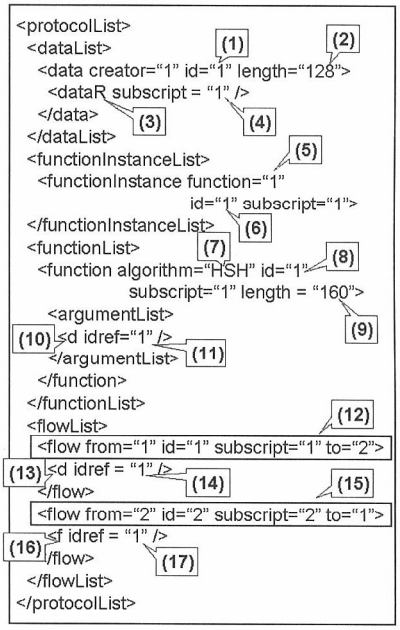

図 8　サンプルプロトコルのプロトコル定義ファイル
Fig. 8　Protocol definition file of sample protocol.

図 9　サンプルプロトコルのスクリプトファイル
Fig. 9　Script file of sample protocol.

(注4)：ハッシュ値はハッシュ関数の出力データという扱いであるためデータ定義には含まれない．

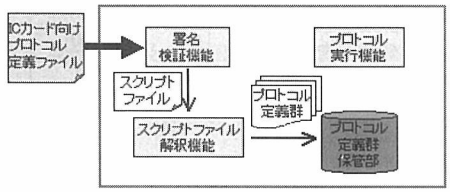

図 10　カスタマイズ時の SPM 内部の処理フロー
Fig. 10　Internal flow of SPM for customization.

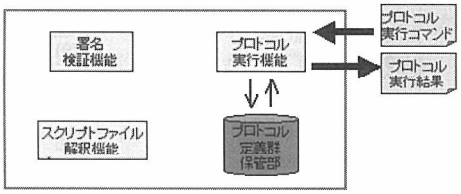

図 11　実行時の SPM 内部の処理フロー
Fig. 11　Internal flow of SPM for execution.

3.2.3 SPM

(a) 機　　能

SPM は，IC カード内にあらかじめ実装され，プロトコルカスタマイズ時に外部から受信した IC カード向けプロトコル定義ファイルを内部に格納し，プロトコル実行時に格納されている IC カード向けプロトコル定義ファイルに従ってプロトコルを実行するものである．

(b) プロトコルカスタマイズ処理フロー

プロトコルカスタマイズ時の SPM の内部処理フローを図 10 に示す．

まずはじめに，SPM は IC カード向けプロトコル定義ファイルの署名を検証し，スクリプトファイルを抽出する．次に，スクリプトファイルからデータ定義や関数定義等のプロトコル定義群を抽出し，プロトコル定義群保管部に格納する．

(c) プロトコル実行処理フロー

プロトコル実行時の SPM の内部処理フローを図 11 に示す．

まずはじめに，SPM はプロトコル実行コマンドを読み取り，プロトコル定義群保管部から対応するフロー定義，関数定義等を抽出する．次に，抽出した定義群に従ってプロトコル処理を行い，プロトコル実行結果を出力する．

4. 評　　価

本章では，内部カスタマイズ方式を実装したシステムの性能評価を行う．

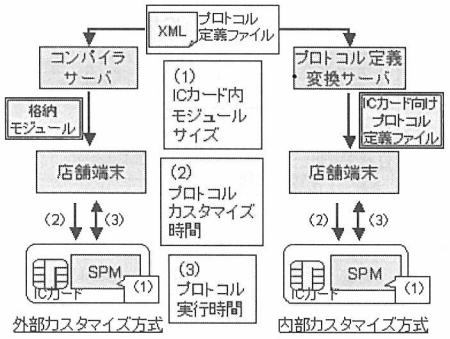

図 12　評価システム概要
Fig. 12　Outline of evaluation system.

表 3　評 価 環 境
Table 3　Evaluation environment.

項目	ホスト	IC カード
OS	Windows XP SP2	JavaCard 2.1.1 GlobalPlatform 2.1
CPU	PentiumM 1.73 GHz	AE46C1
HDD	40 GB	-
EEPROM	-	60 KB
RAM	480 MB	2.7 KB

4.1 評価システム概要

評価に使用したシステムの概要を図 12 に示す．内部カスタマイズ方式の有効性を評価するために，外部カスタマイズ方式を実現したコンパイラサーバ，定義ファイルコンバータを組み込んだプロトコル定義変換サーバ，ホスト機能を組み込んだ店舗端末を構築し，(1) IC カード内モジュールのサイズ，(2) プロトコルカスタマイズ時間，(3) プロトコル実行時間の 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やプロトコル定義変換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，図 12 に示したホスト（コンパイラサーバ，プロトコル定義変換サーバ，店舗端末）と IC カードの仕様を表 3 に示す．

4.2 評価フロー

性能評価には，図 13 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，大きく二つの部分から構成される．一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もとにホスト内で IC カード認証を行う．二つ目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もとに IC カード内でホスト認証を行う．

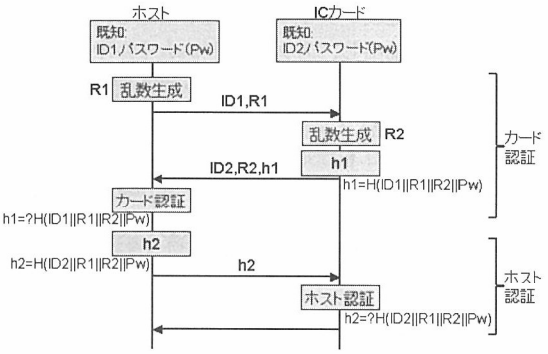

図 13　評価フロー
Fig. 13　Evaluation flow.

表 4　IC カード内モジュールサイズの評価結果
Table 4　Evaluation result for capacity in smart card.

外部方式	内部方式	増加率
4025 byte	13799 bytes	343%

表 5　カスタマイズ処理時間評価結果
Table 5　Evaluation result for protocol customizing time.

外部方式	内部方式	増減量
6615 ms	1891 ms	−4724 ms

表 6　プロトコル実行時間評価結果
Table 6　Evaluation result for protocol processing time.

処理項目		外部方式	内部方式	増減量
カード認証	受信	133 ms	166 ms	33 ms
	演算	78 ms	157 ms	79 ms
	送信	56 ms	52 ms	−4 ms
ホスト認証	受信	106 ms	125 ms	19 ms
	演算	38 ms	101 ms	63 ms
	送信	1 ms	25 ms	24 ms
計		412 ms	626 ms	214 ms

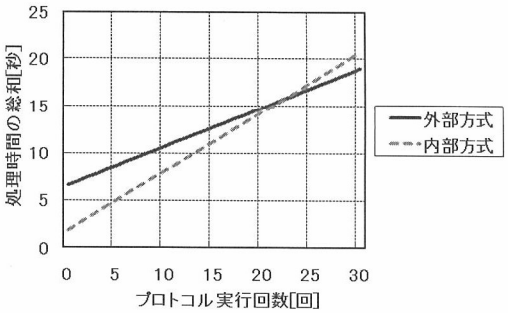

図 14　外部方式と内部方式の比較
Fig. 14　Comparison of proposal method.

4.3　評価結果

4.3.1　IC カード内モジュールサイズ

図 13 のフローを実現する SPM を外部カスタマイズ方式により実装した場合（以降，外部方式と記述する）と，内部カスタマイズ方式により実装した場合（以降，内部方式と記述する）の IC カード内モジュールサイズを評価した．評価結果を表 4 に示す．

この結果，内部方式は外部方式に比べ IC カード内モジュールサイズが大幅に増加することが判明した．よって，内部方式を適用するためには IC カード内のリソースが豊富に必要であることが判明した．

4.3.2　プロトコルカスタマイズ時間

外部方式と内部方式のプロトコルカスタマイズ時間を評価した．評価結果を表 5 に示す．なお，外部方式，内部方式ともに測定は 10 回行い，表にはその平均値を記載している．

この結果，内部方式は外部方式のプロトコルカスタマイズ時間よりも約 4.7 秒短縮できていることが判明した．よって内部方式は環境の変化等に応じて短時間でプロトコルのカスタマイズを実現可能であることが判明した．

4.3.3　プロトコル実行時間

外部方式と内部方式のプロトコル実行時間を評価した．評価結果を表 6 に示す．なお，外部方式，内部方式ともに測定は 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それぞれ示す．

この結果，内部方式は外部方式に比べ 200 ms 程度処理時間が増加することが判明した．

4.4　プロトコルカスタマイズ方式の選択

4.3 の評価結果を用いて，プロトコルカスタマイズ時に外部カスタマイズ方式と内部カスタマイズ方式のどちらの方式を選択すべきか検討を実施した．表 5 に示す結果より，プロトコルを頻繁にカスタマイズするサービスにおいては内部カスタマイズ方式が有効であると考えられる．また，プロトコルを頻繁にカスタマイズしないサービスであっても，図 14 に示すようにカスタマイズ後のプロトコル実行回数が 20 回以内であれば，内部カスタマイズ方式の方が外部カスタマイズ方式よりもカスタマイズ時間とプロトコル実行時間を合わせた処理時間の総和が小さくなることが判明した．よって，プロトコルカスタマイズの頻度やカスタマイズ後のプロトコル実行回数に応じてプロトコルカ

スタマイズ方式を選択する必要があると考えられる．

4.5 カスタマイズ処理時間の確保が困難なサービスへの適用

4.3.2 の評価結果より，内部カスタマイズ方式を用いた場合でも約 2 秒程度のカスタマイズ処理時間が必要となるため，交通系サービスのようにプロトコル実行を数百 ms 以内に完了させる必要があるサービスには適用が困難であることが判明した．そこで，プロトコル実行時に少しずつプロトコル定義ファイルを送信し，徐々にプロトコルカスタマイズを実施する方式を検討した．提案する方式を図 15 に示す．ここでは，IC カードに既に格納されているプロトコルをプロトコル A，新たにカスタマイズするプロトコルをプロトコル B とする．

まずはじめに，店舗端末においてプロトコル A の実行コマンドとプロトコル B を記載した IC カード向けプロトコル定義ファイルの一部を組み合わせたプロトコルコマンドを作成し，IC カード内の SPM に送付する．SPM はプロトコルコマンドを IC カード向けプロトコル定義ファイルの一部とプロトコル実行コマンドに分離する．次に，IC カード向けプロトコル定義ファイルの一部をプロトコル定義群一時保管部に格納する．一方，分離したプロトコル実行コマンドに対応するプロトコル A をプロトコル定義群保管部から取得し，プロトコル処理を実行する．これらの処理フローをプロトコル A を実行する際に複数回繰り返し，プロトコル定義群一時保管部にプロトコル B の定義ファイルを徐々に蓄積する．そして，プロトコル B のプロトコル定義ファイルがすべて蓄積された段階で，IC カード向けプロトコル定義ファイルの署名を検証し，スクリプトファイルを抽出し，スクリプトファイル中のプロトコル B のプロトコル定義群をプロトコル定義群保管部に格納されているプロトコル A のプロトコル定義群と置き換え，プロトコルカスタマイズを実現する．このようなフローを実施することにより，カスタマイズ処理時間の確保が困難なサービスにおいてもセキュリティプロトコルをカスタマイズすることが可能となる．

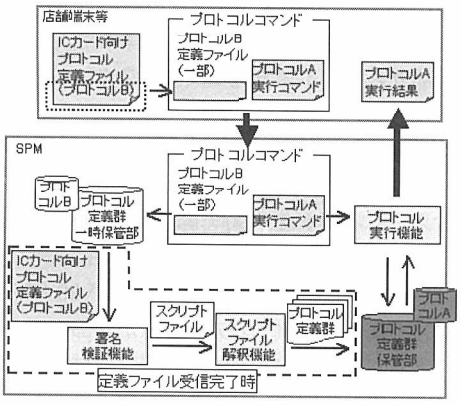

図 15 提案するプロトコルカスタマイズ手順
Fig. 15 Proposed scheme for customization of security protocol.

5. 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 IC カード内部でセキュリティプロトコルをカスタマイズする内部カスタマイズ方式について記述した．内部カスタマイズ方式のプロトコルカスタマイズ時間を評価した結果，外部カスタマイズ方式に比べ大幅な処理時間の短縮を実現できていることが判明した．これにより，外部カスタマイズ方式では実現が困難であった環境やサービスの変化に応じてセキュリティプロトコルを短時間でカスタマイズすることが可能となった．

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外部カスタマイズ方式と内部カスタマイズ方式を動的に変更するシステム等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] 内山宏樹, 梅澤克之, 洲崎誠一, "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," 信学論 (B), vol.J92-B, no.7, pp.1030–1038, July 2009.
[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.

(平成 21 年 3 月 30 日受付, 7 月 21 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).