RM-009

スマートフォンを用いたリモート接続システムの開発と評価

Development and Evaluation of a Remote Access System with Smart Phone

梅澤 克之† 加藤 崇利† 手塚 悟‡
Katsuyuki Umezawa Takatoshi Kato Satoru Tezuka

1．まえがき

近年，企業における情報漏洩事故が多発しており，情報管理の徹底が求められている．情報漏洩事故の事例としては，自宅に空き巣が入り，個人情報が保存された個人所有 PC が盗難されたり，ノート PC や USB メモリの入った鞄を帰宅中の電車内で盗難されたりする事故が報告されている．これらに共通していることは，情報を社外へ持ち出しているということである．このような情報漏洩対策として，企業においてはシンクライアントソリューションのニーズが高まっている．記憶領域を持たないノート PC を用いて，社内システムに VPN[1] 接続し，情報そのものは社外に出さないという運用方法である．このようなシンクライアントシステムにおいては，社外から社内に接続する際の認証機能が重要となる．従来はノート PC などに IC カードリーダをつなげ，個人用の IC カード（社員証等）やフラッシュメモリ型のセキュリティデバイス（以降 KeyMobile と呼ぶ）を用いて個人認証を行い社内への接続を許可することが行われていた．このようなリモート接続を行う場合，セキュリティデバイスやそのリーダを鞄の中から探し出し，PC に接続し準備をするのに手間がかかっていた．その手間を省くために，セキュリティデバイスを PC のスロットに常時接続しておくことも考えられるが，上述のように，盗難や紛失の際に PC とともにセキュアデバイスも同時に無くなることになり，安全性の観点で推奨されない．

本論文では，新規に開発した microSD™ 型 KeyMobile を装着したスマートフォンを用いて，スマートフォントとノート PC 間で無線（Bluetooth）を用いてコマンド送受信を行い VPN 接続を行う Bluetooth 連携型リモート接続システムの提案を行う．具体的にはスマートフォンを PC と連携させる際に，CSP（Cryptographic Service Provider）と，PC/SC（Personal Computer/Smart Card）という 2 つのインタフェースを Bluetooth で送受信できるデータ列に変換して連携を行う方法を提案し，開発したシステムの性能を計測し，実測値を評価する．

以下では，まず，2 章で従来技術として KeyMobile の概要，および CSP と PC/SC の概要，さらに従来型のリモート接続システムの概要について記述する．3 章でスマートフォンを用いたリモート接続の提案を行う．4 章で性能を評価し，5 章で性能に関する考察を行う．そして最後に 6 章でまとめと今後の課題を示す．

2．従来技術

従来技術として，スマートカード機能とフラッシュメモリ機能を併せ持つ KeyMobile の概要，暗号および IC カード関連の標準的な機能としての CSP と PC/SC，USB リーダを介して KeyMobile を用いてリモート接続を行う従来型リモート接続の概要を示す．

2.1 KeyMobile の概要

KeyMoble は，スマートカード機能とフラッシュメモリ機能を併せ持つセキュリティデバイスである．KeyMobile が持つ主な特徴は下記の通りである．

- IC チップとフラッシュメモリを搭載
- クレジットカードと同等の IC チップを搭載
- IC チップ内に電子証明書を予め格納済み
- VPN アクセス時のクライアント認証が可能

この他に，電子証明書の利用には PIN（暗証番号）入力を必要とし，PIN を複数回誤ると KeyMobile が自動的にロックする機能を有する．また，管理者のポリシーにより，PIN がロックするまでの回数を「5～255 回」あるいは「無制限」へ変更可能である．

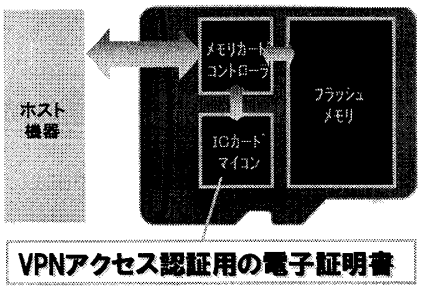

図 1 KeyMobile の内部構造

2.2 CSP と PC/SC

暗号および IC カード関連の標準的な機能として CSP と PC/SC について示す．

2.2.1 CSP（Cryptographic Service Provider）

CSP とは，CryptoAPI というインタフェースで呼び出される暗号エンジンで，アプリケーションやシステムの要求に応じて暗号化やデジタル署名などの機能を提供する．CSP 関数の一覧を下表に示す．

表 1 CSP 関数一覧

CSP 関数	意味
CPAcquireContext	CSP 内の特定の鍵コンテナのハンドルを取得する．
CPGetProvParam	CSP の属性を検索する．
CPReleaseContext	CPAcquireContext で取得したハンドルを開放する．
CPSetProvParam	CSP の指定した属性を設定する．
CPDeriveKey	パスワードから得られる鍵を生成する．

† （株）日立製作所, Hitachi, Ltd
‡ 東京工科大学, Tokyo University of Technology

CSP 関数	意味
CPDestroyKey	鍵を破棄する.
CPDuplicateKey	鍵のコピーを作成する.
CPExportKey	アプリケーションのメモリスペースで鍵を CSP から転送する.
CPGenKey	ランダム鍵を生成する.
CPGenRandom	ランダムデータを生成する.
CPGetKeyParam	鍵のパラメータを検索する.
CPGetUserKey	鍵交換後，あるいは署名鍵のハンドルを取得する.
CPImportKey	秘密鍵をインポートする.
CPSetKeyParam	証明書の書き込みをする.
CPDecrypt	指定した暗号化鍵を利用して暗号文のセクションを復号する.
CPEncrypt	指定した暗号化鍵を利用して平文のセクションを暗号化する.
CPCreateHash	ハッシュオブジェクトを生成して，そのハンドルを返す.
CPDestroyHash	ハッシュオブジェクトを破壊する.
CPDuplicateHash	ハッシュおよびその状態のコピーを作成する.
CPGetHashParam	ハッシュオブジェクトのオペレーションに関するデータを検索する.
CPHashData	指定したハッシュオブジェクトを加えてデータブロックをハッシュする.
CPHashSessionKey	指定したハッシュオブジェクトを加えてセッション鍵をハッシュする.
CPSetHashParam	ハッシュオブジェクトのパラメータをセットする.
CPSignHash	指定したハッシュオブジェクトに署名する.
CPVerifySignature	電子署名を照合する.

2.2.2 PC/SC (Personal Computer/Smart Card)

PC/SC とは，異なるメーカーの IC カードやリーダライタを Windows プラットフォームで相互運用できるように策定された統一仕様である．PC/SC の仕様は以下の 8 つのパートから構成されている．

Part 1: PC/SC の導入とアーキテクチャ概要
Part 2: PC/SC 準拠 IC カードとインタフェースデバイス (IFD) へのインタフェース要件
Part 3: PC に接続される IFD 要件
Part 4: IFD 設計に関する考慮点と参照設計情報
Part 5: IC カードリソースマネージャの定義
Part 6: IC カードサービスプロバイダインタフェースの定義
Part 7: アプリケーション分野および開発者が設計する上での考慮点
Part 8: セキュリティおよびプライバシーのための IC カードデバイス実装に関する推奨

Windows 環境では，Part 5 で規定されるインタフェースが，Windows スマートカード(WinSCard)クライアント API として利用可能となっている．

表 2 PC/SC Part5(WinScard) 関数一覧

PC/SC 関数	意味
SCardBeginTransaction	カードに対して論理トランザクションを開始する.
SCardCancel	終了していないアクション(ブロックされた要求)を中止する.
SCardConnect	リーダに入っているカードとの接続を開く.
SCardControl	リーダデバイスとの直接通信をサポートする.
SCardDisconnect	カードからの接続を断つ.
SCardEndTransaction	論理トランザクションを終了させる.
SCardEstablishContext	IC カードリソースマネージャとの通信で使用される Context を作成する.
SCardForgetCardType	リソースマネージャデータベースから特定のカードタイプの情報を取り除く.
SCardForgetReader	リソースマネージャデータベースからカードリーダを取り除く.
SCardFreeMemory	メモリを解放する.
SCardGetAttrib	リーダの属性を取得する.
SCardGetCardTypeProviderName	与えられたカードタイプに関連するサービスプロバイダの名前を取得する.
SCardGetProviderId	与えられたカードタイプに関連する関連サービスプロバイダ (ICCSP) への参照を取得する.
SCardGetStatusChange	リーダの状態の変更(カードが取り出されたり入れられたり)を取得する.
SCardIntroduceCardType	一致する ATR 参照値などの情報とともに新しいカードタイプをリソースマネージャデータベースに加える.
SCardIntroduceReader	リソースマネージャデータベースに新しいカードリーダを加える.
SCardIsValidContext	カードコンテキストが有効か否かを判定する.
SCardListCards	与えられた ATR 文字列またはインタフェースリストに合うカードタイプのリストを取得する.
SCardListInterfaces	与えられたカードタイプでサポートされたカードインタフェースに関連する GUID のリストを取得する.
SCardListReaders	一つないしそれ以上のグループに割り当てられたリーダのリストを取得する.
SCardLocateCards	リーダの状態に関する情報を取得する.
SCardLocateCardsByATR	与えられた ATR 参照値に基づいてリーダの状態に関する情報を取得する.
SCardReconnect	カードへの再接続を行う.
SCardReleaseContext	SCardEstablishContext で作成された Context を解放する.
SCardSetAttrib	リーダの属性を設定する.
SCardSetCardTypeProviderName	与えられたカードタイプに関連するサービスプロバイダの名前を設定する.
SCardStatus	カードのステータスを取得する.
SCardTransmit	カードにデータを送り，カードから帰ってくるデータを受け取る.

2.3 従来型リモート接続

本節では，従来型のリモート接続の概要を説明する．現状では，シンクライアント PC に RS-MMC(Reduced Size MultiMedia Card)型の KeyMobile 用のリーダを接続し，リーダに KeyMobile を挿すことによって社外から社内へリモート接続を行っている[2]．接続例を図 2 に示す．

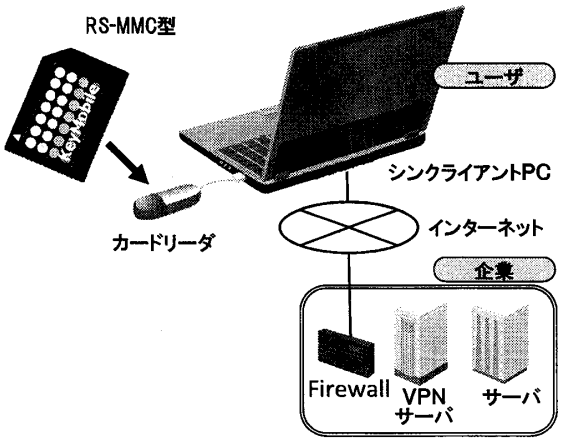

図 2 従来のリモート接続システム

2.4 従来型リモート接続のフロー

従来型のリモート接続の関数呼び出しを含むフローの概略を図3 に示す．シンクライアントPC内の上位アプリケーションがICアクセスライブラリ(CSP)を呼び，CSPが，PC/SCで規定されているSCardBeginTransactionやSCardTransmitなどの関数を呼び出すことで，PC/SCライブラリを経由してIC カードとのコマンドの送受信が行われる．

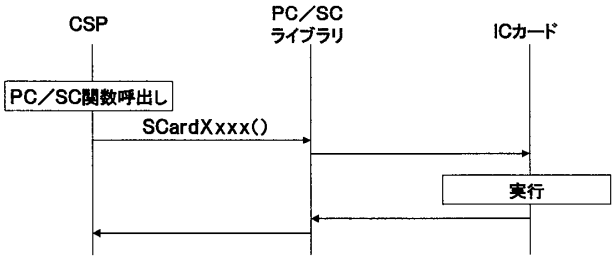

図 3 従来のリモート接続のフロー

3．スマートフォンを用いたリモート接続

前節に示したように，従来は，シンクライアント PC にリーダを接続し，IC カードアクセスを行っていた．このような使い方の場合，リーダや IC カードの準備に手間がかかるため使い勝手が悪かった．これを解決するために，セキュリティデバイスを PC のスロットに常時接続しておくことも考えられるが，盗難や紛失時に，PC と同時にセキュアデバイスも無くなることになり，安全性の観点で推奨されない．そこで，筆者らはスマートフォンや携帯電話機に直接挿せる smartSD™ 型の KeyMobile を開発した．これにより，スマートフォンを胸ポケットに入れておくだけで，リーダ等を準備する必要もなく，リモートアクセスを行うことを可能とした．

3.1 全体アーキテクチャ

図 4 にシンクライアント PC と KeyMobile を装着したスマートフォンを Bluetooth で連携し，インターネットを経由して社内システムに接続する Bluetooth 連携型リモート接続システムの概要を示す．

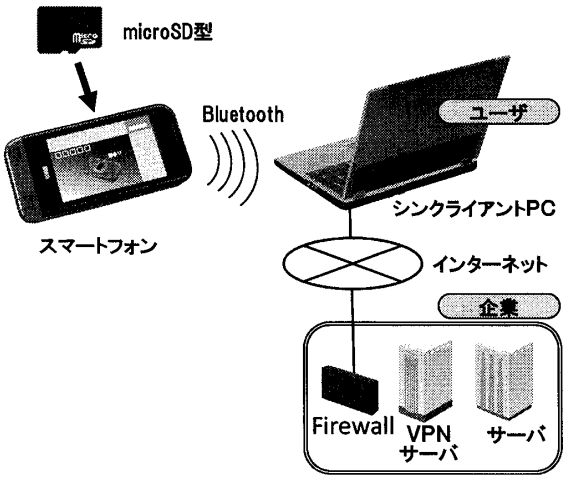

図 4 Bluetooth 連携型リモート接続システム

また，全体構成を図 5 に示す．図 5 に示すように PC とスマートフォンを連携させる方式として CSP の関数を Bluetooth 通信で連携させる方式（CSP 連携方式）と，PC/SC の関数を Bluetooth 通信で連携させる方式（PC/SC 連携方式）の 2 通りの方式を提案する．

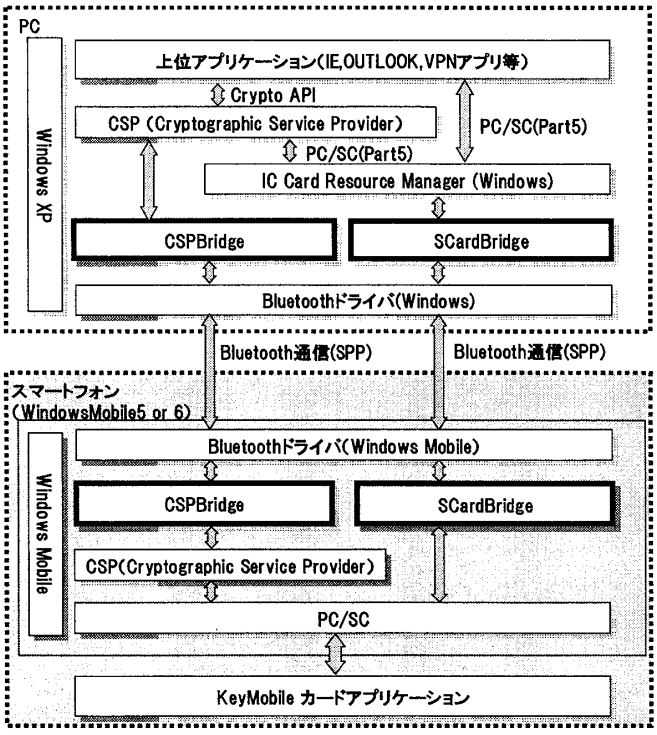

図 5 提案方式の全体アーキテクチャ

3.2 CSP 連携方式

CSP 連携方式は，PC 側の上位アプリケーションが CryptoAPI を呼び出した後，CSP が直接 Bluetooth 連携機能（図 5 の CSPBridge）を呼び出し，CSPBridge は，CSP 関数を Bluetooth で送受信可能なようにシリアル化（バイト列化）して，Windows の Bluetooth ドライバ経由でスマートフォンへコマンドを送信する．Windows Mobile の

Bluetooth ドライバ経由でコマンドを受け取ったスマートフォン側 CSPBridge が，シリアル化されたコマンドを元に戻して，スマートフォン内の CSP インタフェースで KeyMobile にコマンドを伝達する．CSP 連携方式のフローを図 6 に示す．

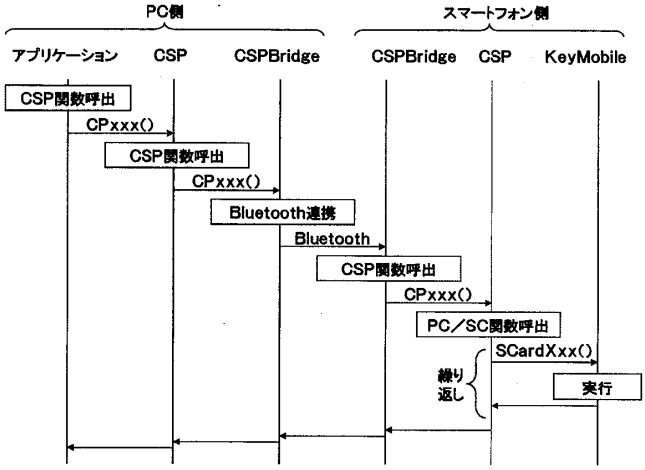

図 6 CSP 連携方式の処理フロー

次に，Bluetooth で送受信するデータの TLV 構造化について示す．図 7 は CPAcquireContext に対応する CryptAcquireContext (WINAPI)の関数定義である．図 8 は図 7 の関数定義を TLV(TagLength Value) 構造としてシリアル化した例である．SendData には，関数の種類を表す情報とともに，入力データとしての引数が TLV 構造化される．ReceiveData には，関数の種類，戻り値とともに出力データとしての引数が TLV 化されて返される．

```
BOOL CRYPTFUNC CryptAcquireContext(
  HCRYPTPROV* phProv,          OUT
  LPCTSTR pszContainer,        IN
  LPCTSTR pszProvider,         IN
  DWORD dwProvType,            IN
  DWORD dwFlags                IN
);
```

図 7 CryptAcquireContext の関数定義

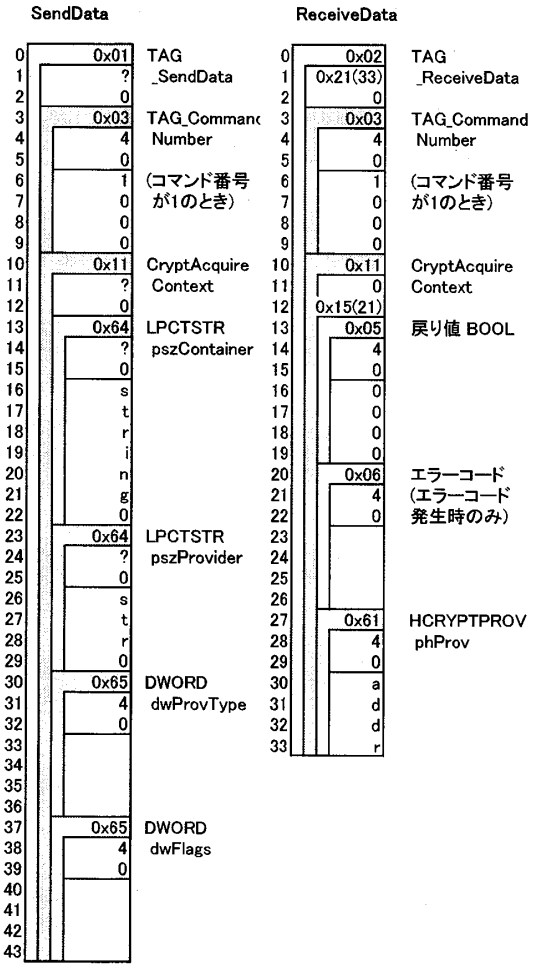

図 8 CPAcquireContext の TLV 構造化

3.3 PC/SC 連携方式

PC/SC 連携方式は，PC 側の上位アプリケーションが CryptoAPI を呼び出した後，CSP が PC/SC Part5 のインタフェースで IC カードリソースマネージャを呼び出し，その後，IC カードリソースマネージャが Bluetooth 連携機能（図 5 の SCardBridge）を呼び出し，SCardBridge は，PC/SC 関数を Bluetooth で送受信可能なようにシリアル化（バイト列化）して，Windows の Bluetooth ドライバ経由でスマートフォンへコマンドを送信する．Windows Mobile の Bluetooth ドライバ経由でコマンドを受け取ったスマートフォン側 SCardBridge が，シリアル化されたコマンドを元に戻して，スマートフォン内の PC/SC インタフェースで KeyMobile にコマンドを伝達する．PC/SC 連携方式の処理フローを図 9 に示す．

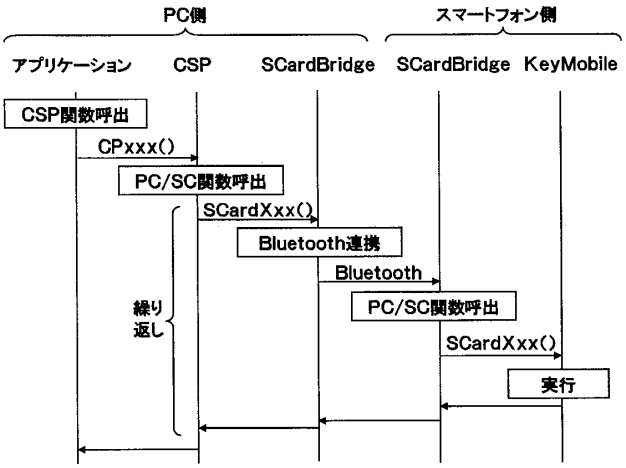

図 9 PC/SC 連携方式の処理フロー

Bluetooth で送受信するデータの TLV 構造化に関しては，CSP 連携方式で述べた方法と同様の方法で，それぞれの PC/SC 関数（SCardXxxx）が TLV 化される．

3.3 ユーザインタフェース

本節では開発した Windows Mobile 側の CSPBridge および SCardBridge について記述する．これらのアプリケーションは，起動時にメモリ上に常駐し，タスクトレイにアイコン化され表示される．タスクトレイのアイコンをタップすることで最大化した状態で最前面に表示され，ログ表示および通信パラメータ設定を行うことができる．図 **10** にログ表示画面を，図 **11** に通信パラメータ設定画面を示す．

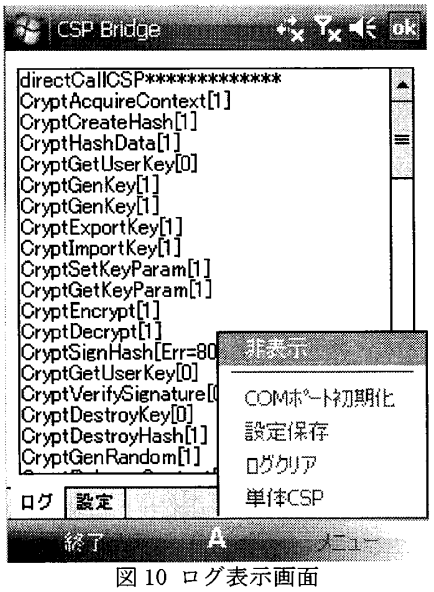

図 10 ログ表示画面

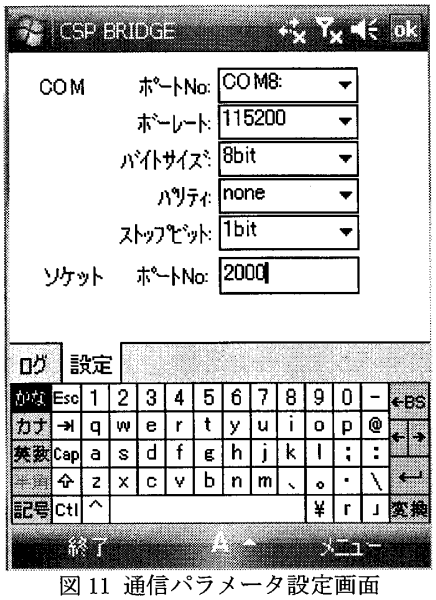

図 11 通信パラメータ設定画面

4．性能評価

本節では，CSP連携方式，PC/SC連携方式，および，従来方式を用いてVPNリモートアクセスを行う時間を計測した．性能測定対象システムの構成は，図4に示した通りである．性能評価で用いたスマートフォンおよびノートPC の機器構成を表3に示す．なお，測定にはBluetooth Ver. 2.1対応のアダプタを用いた．

表 3 性能測定対象システム構成機器

	スマートフォン	ノート PC
CPU	Marvell(R) PXA 270 プロセッサー (520MHz)	Intel(R) Core™ 2 Duo プロセッサー (2.4GHz)
OS	Windows Mobile 6	Windows XP SP3

表 4 に測定結果を示す．表 4 に示した CSP 関数の種類と順序は，PC 上の VPN クライアントアプリケーションが，VPN 接続を完了するまでに呼び出す CSP 関数の典型的な例である[1]．各方式において CSP 関数の処理に要する時間を示す．なお従来とは，他方式と同じ KeyMobile を装着したリーダを直接 PC に接続した構成での測定結果である．どの方式においても同一の KeyMobile カードおよび PC と VPN サーバの間は最大 2.4Mbps の 3G ネットワークを用いた．また，表 3 の結果は 10 回測定の平均値である．

表 4 測定結果

CSP 関数	従来	PC/SC	CSP
CPAcquireContext	1.566	4.747	10.772
CPGetUserKey	0.075	0.667	2.066
CPGetKeyParam	0.020	0.058	0.473
CPExportKey	0.000	0.000	0.308
CPDestroyKey	0.023	0.027	1.003
CPReleaseContext	0.197	1.034	2.611

[1] IC カードに搭載されている電子証明書の数や PIN の要求の有無などの環境の違いにより呼び出される回数や関数の種類に違いが生じる場合がある．

CPAcquireContext	1.714	4.808	10.074
CPGetUserKey	0.089	0.669	2.097
CPGetKeyParam	0.000	0.058	0.466
CPGetKeyParam	0.020	0.000	0.425
CPDestroyKey	0.023	0.026	1.026
CPCreateHash	0.016	0.016	0.445
CPSetHashParam	0.006	0.009	0.567
CPSignHash	0.036	0.024	0.524
CPSignHash	1.745	6.530	3.622
CPDestroyHash	0.062	0.061	0.524
CPReleaseContext	0.308	1.331	2.853
合計時間	5.901	20.065	39.855

（単位は秒）

表 4 に示したように，従来方式としてリーダを直接 PC に接続する方式に比べて PC/SC 連携方式で約 14 秒遅くなるという結果となった．しかし従来方式の場合には，KeyMobile カードおよびリーダを鞄の中などの他の携行品の中から探し出し PC に接続するという煩雑な処理をユーザに強いているのに対して，提案方式の場合には，その手間を省くことができるためユーザの利便性は向上しているということができる．

5．考察

本節では前節の測定結果のうち PC/SC 連携方式と CSP 連携方式の動作速度の違いについて考察する．研究開発当初，PC/SC 連携方式では，1 つの CSP 関数に対して複数の PC/SC 関数が Bluetooth で送受信されるため，CSP 連携方式より効率が悪いであろうと考えた．

実際に予備実験において，SCardBeginTransaction を 1 回，SCardTransmit を 100 回，SCardEndTransaction を 1 回行うような処理の場合には，CSP 連携方式のほうが 32%の高速であるという結論を得ていた[3]．しかし，表 2 より，ほとんどの関数が PC/SC 連携方式のほうが CSP 連携方式より速い結果となった．

本結果を考察するにあたり，実際に VSP 接続を行う際の認証処理において PC 上の VPN クライアントから呼び出される CSP 関数と，その CSP 関数がどんな PC/SC 関数を何回呼び出しているかの調査を行った．表 5 に結果を示す．

表 5 CSP 関数と PC/SC 関数の対応関係

CSP 関数	PC/SC 関数	回数
CPAcquireContext	SCardEstablishContext	1
	SCardConnect	1
	SCardBeginTransaction	5
	SCardTransmit	19
	SCardEndTransaction	5
CPGetUserKey	SCardBeginTransaction	1
	SCardTransmit	2
	SCardEndTransaction	1
CPGetKeyParam	なし	0
CPDestroyKey	なし	0
CPReleaseContext	SCardBeginTransaction	1
	SCardTransmit	1
	SCardEndTransaction	1
	SCardDisconnect	1
	SCardReleaseContext	1
CPAcquireContext	SCardEstablishContext	1
	SCardConnect	1
	SCardBeginTransaction	5
	SCardTransmit	19
	SCardEndTransaction	5
CPGetUserKey	SCardBeginTransaction	1
	SCardTransmit	2
	SCardEndTransaction	1
CPGetKeyParam	なし	0
CPDestroyKey	なし	0
CPCreateHash	なし	0
CPSetHashParam	なし	0
CPSignHash	なし	0
CPSignHash	SCardBeginTransaction	4
	SCardTransmit	15
	SCardEndTransaction	4
CPDestroyHash	なし	0
CPReleaseContext	SCardBeginTransaction	1
	SCardTransmit	1
	SCardEndTransaction	1
	SCardDisconnect	1
	SCardReleaseContext	1

表 4 および表 5 の結果から，CSP 関数の振る舞いについて以下の 3 分類が可能である．

- IC カードでの処理が不要で PC/SC 関数を呼び出さない関数
- 複数の PC/SC 関数から構成され，一括してスマートフォン側で処理したほうが高速な関数
- 複数の PC/SC 関数から構成されるが，一括してスマートフォン側で処理すると遅くなる関数

まず，「IC カードでの処理が不要で PC/SC 関数を呼び出さない関数」（表 5 の対応する PC/SC 関数欄に「なし」と表記した関数）に関しては，図 12 と図 13 に示すように，PC/SC 連携方式では PC 側だけで処理される関数が，CSP 連携方式では，Bluetooth で送受信されているために多くの時間がかかっていることが確認できる．これらの関数に関する解決策は比較的容易であり，PC 側の CSPBridge に当該関数をスマートフォン側に転送しない機能を付加することで解決可能と考えられる．

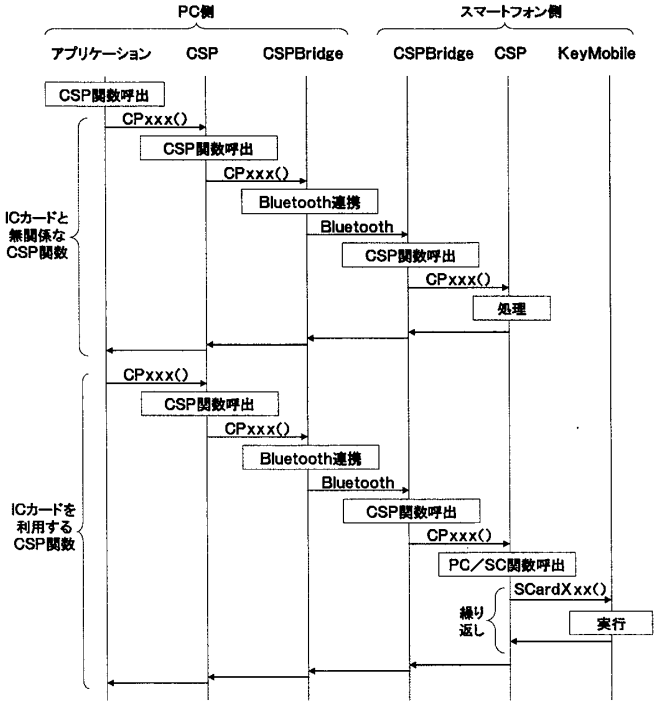

図 12 CSP 連携方式の実際の処理フロー

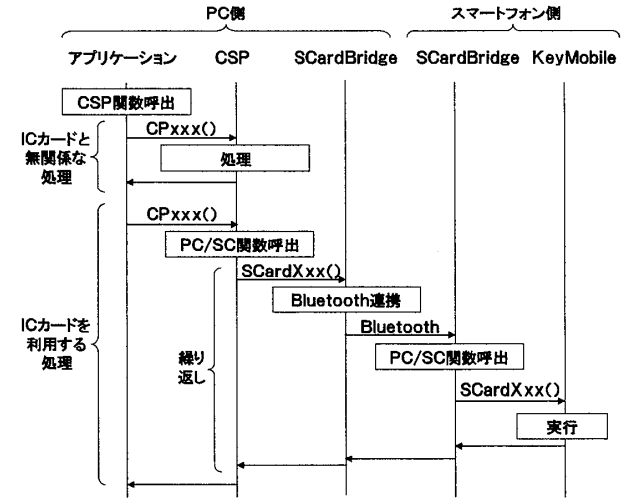

図 13 PC/SC 連携方式の実際の処理フロー

次に，「複数の PC/SC 関数から構成され，一括してスマートフォン側で処理したほうが高速な関数」に関しては，CPSignHash 関数が該当する．本結果は，Bluetooth 通信を複数回繰り返し行うより，その上位関数をまとめて処理したほうが高速になるはずであるという当初の想定および参考文献[3]に示した予備実験の結果に当てはまるものである．

最後に，「複数の PC/SC 関数から構成されるが，一括してスマートフォン側で処理すると遅くなる関数」に関して考察する．これらの関数に関しては，IC カード内での処理時間は CSP 連携方式も PC/SC 連携方式も同一と考えられるため，IC カード内での処理以外の処理を PC 上で実行するよりもスマートフォン上で実行するほうが多くの時間を要している結果であるということができる．

6．まとめと今後の課題

本論文では，microSD™ 型 KeyMobile を装着したスマートフォンを用いて，スマートフォントとノート PC 間で無線（Bluetooth）を用いてコマンド送受信を行い VPN 接続を行う Bluetooth 連携型リモート接続システムの提案を行った．具体的には，スマートフォンを PC と連携させる際に，CSP と PC/SC の 2 つのインタフェースを Bluetooth で送受信できるデータ列に変換して連携を行う方法の提案を行った．さらに開発したシステムを実測し，実測値に対する評価を行った．評価の結果，従来方式としてリーダを直接 PC に接続する方式に比べて PC/SC 連携方式で約 14 秒遅くなるという結果となったが，リーダやセキュリティデバイスを鞄の中などから探し出し PC に接続するという煩雑な処理を省くことができるという点でユーザの利便性を向上させることができた．また，CSP 連携方式と PC/SC 連携方式にはそれぞれ利点と欠点があることが判明し，両者を組み合わせることによって更なる性能向上を図る見通しが立った．

今後は，「CSP 関数の中で複数の PC/SC 関数から構成されるが，一括してスマートフォン側で処理すると遅くなる関数」について，スマートフォン上での処理に関する詳細な解析を行い更なる性能向上を目指す．また自分の PC 以外の，例えば共有スペースに設置された共用 PC を使ってアクセスするための PC とスマートフォンの連携技術に関して研究開発を進める予定である．

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「端末プラットフォーム技術に関する研究開発」の成果の一部である．

商標等に関する表示

参考文献

[1] B. Gleeson, A. Lin, J. Heinanen, G. Armitage and A. Maris: *RFC 2764 - A Framework for IP Based Virtual Private Networks*, IETF, February 2000.
[2] 中西，牧野，小高，杉山，石原：“「セキュアクライアントソリューション」を支える「セキュリティPC」と周辺装置，” 日立評論Vol.88 No.05, p.p. 26- 29, 2006/5.
[3] 梅澤克之，洲崎誠一，“スマートフォンを用いたリモート接続システムの開発，” 第31回情報理論とその応用シンポジウム予稿集，pp.971-974, Oct. 2008.