Vol.2020-DPS-182 No.26
Vol.2020-CSEC-88 No.26
2020/3/12

SIM によるプライベートなクラウドストレージを用いたパスワード管理システムの実装と評価

成田彩織[1]　磯原隆将[2]　窪田歩[2]　山﨑徳和[1]

概要：PC やスマートフォンで，多量の ID やパスワード等の認証情報を簡便に管理するツールとして，パスワードマネージャー（PM）が利用されている．PM の中には，管理する認証情報の更新やバックアップ，復元等の処理の利便性を高めることを目的として，パブリックなクラウドストレージを利用するものがある．しかし，このような PM では，クラウドサービスを提供する第三者に認証情報を預ける性質上，インサイダー攻撃に対する脆弱性が指摘されている．そこで本研究では，宅内等の LAN 環境に設置するホームゲートウェイが備える SIM をプライベートなクラウドストレージとして利用するパスワード管理システムを提案する．提案システムは，通信経路の判別に基づいて，SIM に保存された認証情報を PM が取得する処理をローカルな通信経路に限定するアクセス制御を行う．本システムにより，クラウドストレージを利用する PM における認証情報の管理の利便性と安全性の向上を実現する．

キーワード：パスワードマネージャー，認証情報，インサイダー攻撃，SIM，アクセス制御

Implementation and evaluation of password management system using private storage by SIM

SAORI NARITA[†1]　TAKAMASA ISOHARA[†2]　AYUMU KUBOTA[†2]
NORIKAZU YAMASAKI[†1]

1. はじめに

PC やスマートフォンを通じてオンラインサービスを利用する際，サービスの利用者を認証する場面で，ID とパスワードの組み合わせを認証のための情報（以降，認証情報）として用いる，知識ベースの認証方式が広く用いられている．このとき，複数のサービスで同一の認証情報を使い回した場合，あるサービスがサイバー攻撃を受けて認証情報が漏洩することによって，同じ情報を使う他のサービスについても，不正アクセス等の影響を受ける恐れがある．そのため，認証情報を使い回さないことが重要となる．しかし，認証を伴うオンラインのサービスの数が増加の一途をたどる現状において，記憶力を頼りに，利用するすべてのサービスに対して個別の認証情報を用いることは困難となっている．

そこで，多量の認証情報を利用する際の利便性を向上させるツールとして，パスワードマネージャー（以降，PM）が用いられる．PM は，利用するサービス毎に対応する認証情報等を管理するアプリケーションソフトウェアである．PM の中には，管理する認証情報の更新やバックアップ，復元等の処理の利便性を高めることを目的として，パブリックなクラウドストレージを利用する，クラウドベース PM がある．クラウドベース PM は，アプリ利用者の認証情報を，クラウドサービスを提供する第三者に預けるという特徴を持つ．そのため，クラウドベース PM には，悪意あるクラウドサービス提供事業者による認証情報の窃盗や流出といった，インサイダー攻撃に対する脆弱性が指摘されている[1]．なお，パブリッククラウドサービス提供事業者のような第三者の介在を理由とするインサイダー攻撃の脅威は，同様のサービス利用形態において共通の課題であるが，多量の認証情報が取り扱われるクラウドベース PM においては，情報を悪用された際の被害が甚大であることから，特に重大な課題であると考える．

そこで本研究では，クラウドベース PM における認証情報の管理の利便性と安全性の向上を目的として，宅内等の LAN 環境に設置するホームゲートウェイが備える SIM をプライベートなクラウドストレージとして利用するパスワード管理システムを提案する．提案システムは，通信経路の判定に基づいて，SIM に保存された認証情報を PM が取得する処理の実行をローカルな通信経路に限定するアクセス制御を行う．

以下，2 章で研究背景，3 章で関連技術について述べ，4 章で提案手法の設計について説明する．そして，5 章で実装，6 章で評価について述べ，7 章で考察を行う．最後に，8 章でまとめる．

1 玉川大学
Tamagawa University
2 KDDI 総合研究所
KDDI Research, Inc.

Vol.2020-DPS-182 No.26
Vol.2020-CSEC-88 No.26
2020/3/12

2. 研究背景

2.1 クラウドを利用する PM

複数の認証情報を管理する目的で，PM が利用されている．PM は，認証情報の管理形態に基づいて，ローカルストレージを用いるローカルベース PM と，クラウドストレージを用いるクラウドベース PM に分類される．

ローカルベース PM は，PM を実装したデバイス本体に内蔵された記憶装置を用いて認証情報を管理する．一方，クラウドベース PM は，インターネットを介して複数のユーザー間で共有するストレージを用いて認証情報を管理する．

2.2 クラウドの利用におけるインサイダー攻撃の脅威

クラウドを利用したデータ管理サービスには，インサイダー攻撃の脅威が指摘されている．この課題の原因として，第三者がデータの管理を行うとの管理形態が挙げられ，管理者が悪意ある攻撃者であった場合に情報流出する恐れがある．この管理形態およびインサイダー攻撃の懸念は，クラウドを利用したサービスに限らず，多くのデータ保管サービスに共通した特徴である．ここで，クラウドベース PM は，認証情報を扱っているために情報を悪用された際の被害が甚大になる恐れがあり，課題の重要性が増す．

2.3 クラウドベース PM を安全に利用するための手法に対する課題

クラウドベース PM の特徴と，その利用にあたって想定される脅威に基づき，クラウドベース PM を安全に利用するための手法に対する課題を次のように定義する．

課題

パブリッククラウドストレージを利用する PM において懸念されるインサイダー攻撃の可能性を回避すること

3. 関連技術

安全なクラウドベース PM の実現に必要な関連技術として，Smart Card，Java Card，APDU，DTCP-IP について述べる．また，SIM の技術的な特徴についても整理する．

3.1 Smart Card

Smart Card は，IC（集積回路）カードとも呼ばれ，以下の 2 つの特徴を有するハードウェアである．

（1） 情報を記録するメモリのみならず，演算やデータの処理等を行う CPU も備える

（2） 記録する情報を物理的および論理的な攻撃から保護する耐タンパー性を有する [2]

図 1 に，Smart Card の内部構造を示す[3]．Smart Card は，データの読み書きに用いるインタフェースの違いにより，接触型と非接触型に分類される．接触型はカード端末機と接続するための外部端子を有し，非接触型は内部にアンテナコイルを有する[4]．接触型の Smart Card の実用例として SIM（Subscriber Identity Module）カードやクレジットカードがあり，非接触型の Smart Card の実用例として交通系 IC カードや電子マネーカードがある．

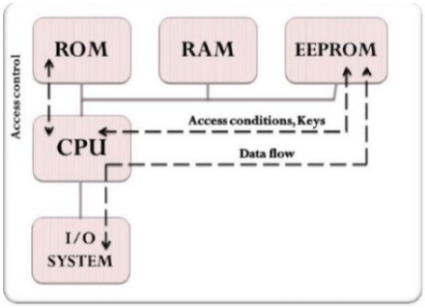

図 1 Smart Card の内部構造

Figure 1 Internal structure of Smart Card.

3.2 Java Card

Java Card は，Sun Microsystems 社が 1990 年代後半に開発した Smart Card 向けのプラットフォーム技術である．Smart Card における，メモリの積載容量や CPU の処理能力の制限を考慮した Java の実行環境[5]であり，Java の最小限のサブセットとして位置づけられる[2]．Java Card のアプリケーションをアプレットと呼び，Java Card API を用いて実装する[6].

3.3 APDU (Application Protocol Data Unit)

Smart Card 上で動作するアプレットとのデータ交換の仕様は，APDU プロトコルを用いる．本プロトコルは， ISO 7816-4 として，国際標準規格が制定されている[4，7].

APDU におけるデータ交換の手順は，カード上のアプレットに対して「コマンド」を送信し，これに対する「レスポンス」をアプレットから受信する仕様となっている[8].

APDU の「コマンド」と「レスポンス」のデータフォーマットを図 2 および図 3 に示す[9].「コマンド」の情報は，カードに対する命令と，命令に付随するデータから構成される．コマンドのデータフォーマットはヘッダーとオプションとしてボディーが定義されている．ヘッダーには，クラスバイトを示す CLA，命令を示す INS，INS の値に依存するパラメータの P1 および P2 等が含まれる．また，ボディーには，カードへ送るデータの長さを示す Lc，カードから送られるデータの長さを示す Le，データ本体をオプション指定できる．一方，「レスポンス」の情報は，アプレットに対するコマンドの実行結果の成否を示すデータから構成される．レスポンスのデータフォーマットはコマンドの成功有無を示すステータスワード SW1 および SW2 と，オプションとしてボディーが定義されている．ステータスワードの例として，コマンドが成功した際に出力される SW1 が 0x90，SW2 が 0x00 の組み合わせが挙げられる．レスポンスにおけるボディーのサイズは，コマンドにおける Le に指定された長さ以下のデータとなる．

コマンド						
ヘッダー				ボディ（オプション）		
CLA	INS	P1	P2	Lc	Data Field	Le

図 2 コマンドのデータフォーマット

Figure 2 Command data format.

レスポンス		
ボディ（オプション）	ステータスワード	
Data Field	SW1	SW2

図 3 レスポンスのデータフォーマット

Figure 3 Response data format.

3.4 SIM の技術的特徴

接触型 Smart Card に分類される SIM は，移動体通信サービスにおいて，契約者の認証やモバイル決済サービスの提供基盤として用いられている[10]．また，通信の機能を活用して，カードに記録された情報を OTA（Over-The-Air）の手段によって遠隔で管理する機能を有する．OTA の手順は GlobalPlatform による仕様の共通化が図られている[11]．

ここで，Smart Card の特徴と上記の説明を総合して，SIM の技術的な特徴を以下に整理する．

（1） 記録する情報を物理的および論理的攻撃から保護する耐タンパー性を有するハードウェアである[2]

（2） Java Card API で実装するアプレットを実行する環境を有する

（3） アプリ本体やそのデータを OTA(Over The Air)によって遠隔から書き換えられる[12]

3.5 DTCP-IP

DTCP-IP（Digital Transmission Content Protection-Internet Protocol）は，家庭内ネットワーク内で著作権保護技術（DRM）によって保護されたコンテンツを伝送するための技術規格である[13]．外部の第三者に傍受されることのないセキュアな状態で，家庭内ネットワークにある機器間のデータ転送を可能とする[13]目的で，機器間の往復遅延時間およびホップ数について確認し，閾値に基づく制御を行う．ここで，往復遅延時間は 7ms，ホップ数は 2 件との閾値が採用されている[14]．

4. インサイダー攻撃の脅威を低減する PM 内認証情報管理手法の提案

クラウドベースの PM において懸念されるインサイダー攻撃の脅威の回避を目的として，認証情報の管理において，クラウドストレージに代わって，SIM を利用するシステムを提案する．提案システムでは，認証情報の管理場所を，宅内等の LAN 環境に設置する H-GW に装着した SIM とする．そして，SIM で管理される認証情報を取得する処理の実行に関して，処理の依頼元が LAN 接続である場合に制限することで，外部の第三者による利用者の意図しない認証情報の取得を防止する．

4.1 システム構成と構成要素の概略

提案システムの構成を図 4 に示す． 本システムは，スマートフォン（以降，スマホ）と，SIM を備える H-GW から構成される．スマホには PM，SIM には認証情報を管理するアプレット（以降，認証情報管理 Applet），H-GW には実行権限の付与と認証情報の中継等を行うアプリ（以降，H-GW アプリ）をそれぞれ実装する．

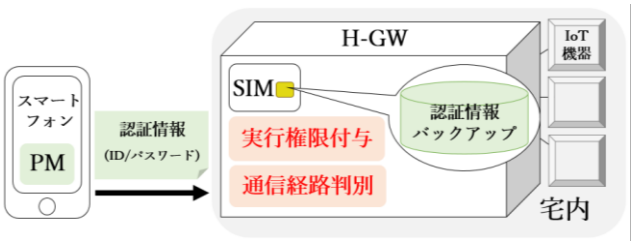

図 4 システム構成

Figure 4 System configuration.

以下，システムを構成する各要素の概略を整理する．

(1) **PM**

PM は，以下の 5 つの機能を備える．

- 認証情報による PM へのログイン認証
- 任意の認証情報の登録，削除，変更
- PM 本体に格納済みの認証情報の表示
- 同期処理の実行
- リカバリ処理の実行

(2) **H-GW アプリ**

H-GW アプリは，PM からの処理要求および認証情報の送受，通信経路判別，認証情報管理 Applet への認証情報の送受および型変換，PM への処理結果報告の機能を有する．

(3) **認証情報管理 Applet**

認証情報管理 Applet は，H-GW アプリからの要求をトリガーとして，認証情報の同期とリカバリに関する処理を行う．

同期処理は，PM から H-GW を通じて，PM を実装した端末本体に格納されている認証情報を SIM 内の認証情報管理 Applet に保管する処理である．

リカバリ処理は，H-GW が SIM 内の認証情報管理 Applet に保管された認証情報を取得し，これを PM へ送信する処理である．

4.2 PM および H-GW 間の通信経路の判別

本システムでは，PM および H-GW 間の通信経路として自宅内（以降，LAN）と自宅外（以降，WAN）の 2 種を想定する．LAN 接続と WAN 接続では，PM が SIM に接続する際の Round Trip Time（RTT）および Time To Live（TTL）が異なることに注目し，H-GW における通信の往復時間およびホップ数の観測結果に対して，閾値に基づく判別を行

う．図 5 に通信経路を判別するフローチャートを示す．

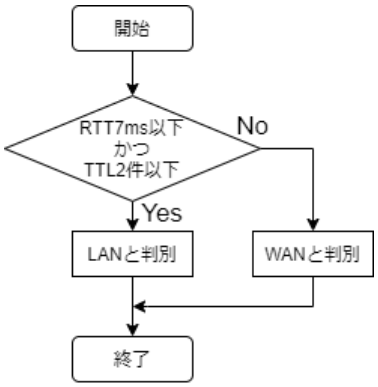

図 5　通信経路判別のフローチャート
Figure 5　Communication path determination flowchart.

4.3 PM と H-GW 間の通信経路の判別に基づいた処理の実行権限の制御ポリシー

本研究では，外部の第三者による意図しないリカバリ処理の実行を防ぐことを安全な認証情報の管理と定義した．これを実現するため，LAN 接続と WAN 接続の場合で，実行可能な処理を区別する制御を実現する．

具体的には，LAN 接続の処理要求は，正規の利用者によるものであると判断して同期処理とリカバリ処理を許可し，WAN 接続の処理要求は，外部の第三者による意図しない接続の可能性があると判断して同期処理のみを許可する．

4.4 SIM 内で管理する認証情報に対する処理

本節では，同期処理とリカバリ処理の詳細な手順を述べる．

(1) 同期処理

同期処理のシーケンスを図 6 に示す．

本処理は，PM における同期処理の実行をトリガーに，H-GW を通じて，SIM 内の認証情報管理 Applet に保管された認証情報を書き換える．このとき，PM は，処理要求とともに PM が保持している認証情報を H-GW に送信する．これを受信した H-GW は，受信した情報を SIM に格納可能なデータ型に変換するとともに，変換したデータを 1 件ずつ認証情報管理 Applet に保管する．そして，保管作業が完了し次第，その旨を PM に報告し，PM はその内容を画面に表示して，一連の処理を終了する．

なお，本処理では，SIM に保管済みの認証情報と，PM を実装した端末本体に格納の認証情報を比較することによる差分の同期ではなく，処理の実行時点でスマホの PM に存在する全ての情報を，SIM 内の認証情報管理 Applet に上書きする同期を実施する．

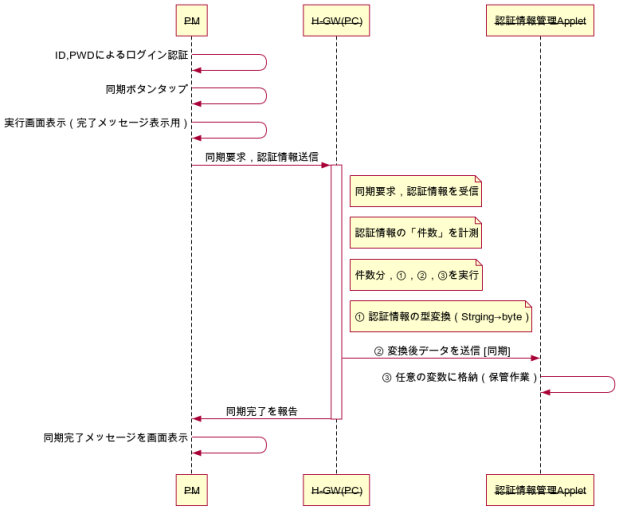

図 6　同期処理のシーケンス図
Figure 6　Sequence diagram of synchronous processing.

(2) リカバリ処理

リカバリ処理のシーケンスを図 7 および図 8 に示す．権限付与制御により LAN 接続時は許可される一方，WAN 接続時は拒否されることから，通信経路別に示す．

本処理は，PM におけるリカバリ処理の実行をトリガーに H-GW を通じて，SIM 内の認証情報管理 Applet に保管された認証情報を取得する．このとき，H-GW では処理要求を受信後，後述の通信経路判別機能を実行する．判別結果が LAN の場合のみ SIM への接続，認証情報の取得，および認証情報の型変換を行う．そして，取得作業が完了し次第，その旨を PM に報告し，PM はその内容を画面に表示して，一連の処理を終了する．

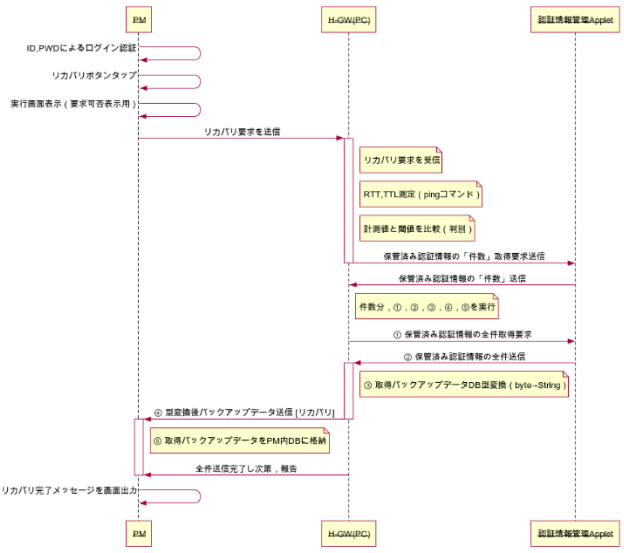

図 7　LAN 接続時のリカバリ処理シーケンス図
Figure 7　Sequence diagram of recovery processing when connecting to LAN.

Vol.2020-DPS-182 No.26
Vol.2020-CSEC-88 No.26
2020/3/12

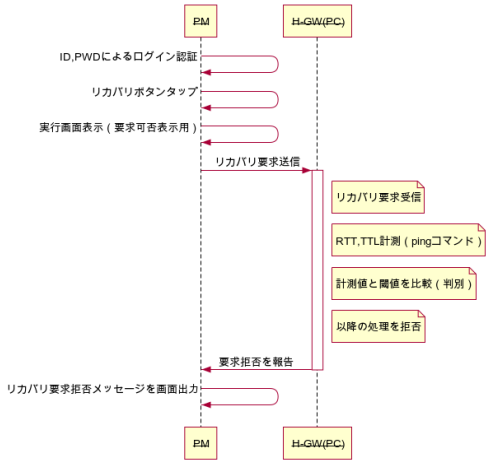

図 8　WAN 接続時のリカバリ処理シーケンス図
Figure 8　Sequence diagram of recovery processing when connecting to WAN.

5. 実装

図 9 に開発するプロトタイプシステムの構成を示すとともに，以下でハードウェアとソフトウェアの仕様について述べる．

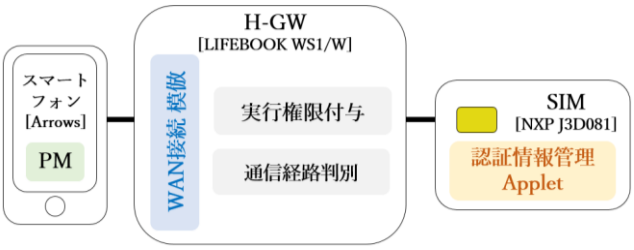

図 9　プロトタイプシステムの構成
Figure 9　Configuration of prototype system.

5.1 ハードウェア仕様

表 1 に使用した機器の仕様を示す．認証情報管理 Applet を実装する Smart Card は，図 10 に示す NXP J3D081 を使用する．また，図 11 に使用したカードリーダ／ライタを示す．なお，実験の都合上，本システムでは PC を H-GW 装置と見立ててシステムを構成した．

表 1　プロトタイプを構成する機器の仕様
Table 1　Specifications of the devices that make up the prototype.

	ハードウェア情報
スマホ	arrows Be F-04K
SIM	NXP J3D081
PC	LIFEBOOK WS1/W
PC 接続用カードリーダ／ライタ	IDBridge CT30 リーダー（PC USB-TR）

図 10　プロトタイプシステムにて使用した Smart Card
Figure 10　Smart Card used in the prototype system.

図 11　プロトタイプシステムにて使用したカードリーダ／ライタ
Figure 11　Reader / Writer used in the prototype system.

5.2 ソフトウェア仕様

(1) **PM**

- 開発環境：Android Studio バージョン 3.5（アプリ開発用の統合開発環境）
- 使用言語：Java 言語

図 12 にログイン認証画面，図 13 にその他の処理実行画面を示す．

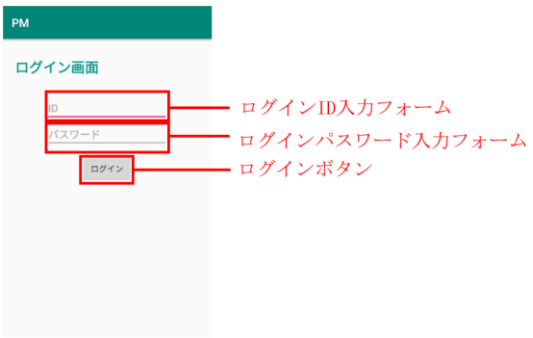

図 12　PM のログイン認証画面
Figure 12　login authentication screen of PM.

Vol.2020-DPS-182 No.26
Vol.2020-CSEC-88 No.26
2020/3/12

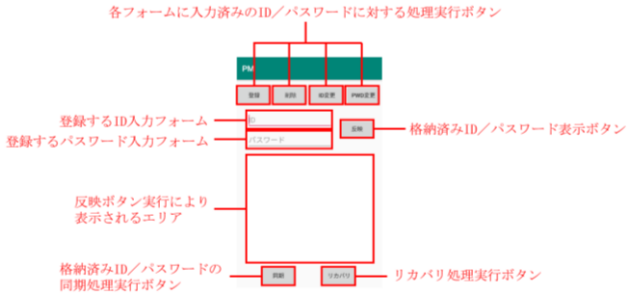

図 13　PM の各機能実行画面

Figure 13　Function execution screen of PM.

(2) **認証情報管理 Applet**

プロトタイプシステムとして簡易な構造を目指し，認証情報送受信処理において，H-GW アプリから受信する INS を，自然数の数値すなわち 10 進数に制限する．これに伴い，INS に割り当て可能な値は 00～99 までの 100 通りとなる．表 2 に INS の設定値および対応する実行動作を示す．表 2 に示す処理構造に伴い，本システムにて同期またはリカバリを実行可能な認証情報の件数は 20 件までとなる．

表 2　認証情報管理 Applet に実装した INS の設定値

Table 2　INS setting value implemented in authentication information management Applet.

INS の設定値	実行動作
01～20	ID の保管［同期］
21～40	パスワードの保管［同期］
41～60	ID の送信［リカバリ］
61～80	パスワードの送信［リカバリ］
90	データ件数の保管［同期］
91	データ件数の送信［リカバリ］

- 開発環境：JCIDE バージョン 2.0.1.70（Java Card 用に設計された統合開発環境）
- 使用言語：Java 言語

(3) **H-GW アプリ**

H-GW アプリとして Windows OS 上で動作可能なアプリを開発した．

- 開発環境：JDK バージョン 8
- 使用言語：Java 言語

(4) **WAN 接続を模倣する遅延の挿入**

プロトタイプシステムにおいては，WAN 接続を模倣するため，ネットワーク遅延を疑似的に発生させるソフトウェアである clumsy を用いた[15].

6. 評価

6.1 実装したアプリの諸元

表 3 に実装したアプリのコードサイズと LOC（Lines Of Code）を示す．

表 3　実装したアプリの諸元

Table 3　Specifications of the implemented application.

	コード部分合計サイズ	合計 LOC（Lines Of Code）
PM	105KB	720
認証情報管理 Applet	24.8KB	758
H-GW アプリ	20KB	424

6.2 プロトタイプシステムにおける認証情報管理動作の検証

擬似的な NW 環境を用いて LAN および WAN を再現し，PM からの H-GW アプリを通じた認証情報管理 Applet に対する各処理が，いずれの経路を経由した場合にも期待通りに制御されていることを確認した．

6.3 データ管理の処理時間

各処理前後に実行したタイムスタンプ取得による SIM で管理する認証情報へのアクセス処理時間の測定実験を行った．計 50 件の実験を行い，処理完了に 32.7 秒を要することを確認した．よって，1 件あたり約 0.7 秒を要し，処理時間は処理件数に比例することを確認した．

6.4 通信経路を判別する RTT および TTL の閾値

6.4.1 閾値の設定

(1) **RTT**

実験に基づく実測値および DTCP-IP の仕様[14]により，RTT の閾値を 7ms に決定した．実測は，プライベート IP アドレスとグローバル IP アドレスを持つサーバに対して，LAN および WAN による接続で ping コマンドを各 25 回実行し，各 100 回の RTT を測定した．実験の結果を表 4 に示す．

表 4　RTT 計測実験の結果

Table 4　RTT measurement results.

	最小値	最大値
プライベート IP アドレス	1 ms	13 ms
グローバル IP アドレス	26 ms	49 ms

(2) **TTL**

DTCP-IP の仕様[14]に基づき，TTL の閾値を 2 件に決定した．

6.4.2 閾値の妥当性

(1) **RTT**

表 4 に示した通り，WAN 接続と想定した実験において，実測値の最小値が閾値を上回っているため，閾値は妥当であると考えられる．また，表 5（後述）に示す通り，95%の確率で LAN 接続であると判断するため，利便性の観点からも，閾値は妥当であると考えられる．なお，プロトタイ

プシステムにおける WAN 接続には，後述の 6.6 節に示す通り，最低 13ms の RTT を要するため，理論上，閾値の 7ms を下回ることは不可能である．

(2) **TTL**

実験の都合上， TTL に関する閾値の妥当性評価については，LAN 接続と想定した実験のみ行う．実験の結果，TTL が 1 件であったことから，閾値の 2 件は妥当であると考えられる．なお，WAN 接続と想定した検証として，一般的なウェブサーバーを宛先とした ping コマンド実行実験も行った．ここで，TTL が 2 件を下回ることはなかった．

6.5 システム利用時の利便性と安全性を考慮した ping コマンドの適切な実行回数

ping コマンドの実行回数別にそれぞれ 100 組の実験を行った．その結果を表 5 に示す．ここで，開発環境である Windows OS における ping コマンドの仕様により，実行回数が 1 回の場合は 1 組あたり 4 件の RTT を取得し，実行回数が 2 回の場合は 1 組あたり 8 件の RTT を取得する．なお，閾値との比較に使用する RTT は， 1 組の ping コマンド実行における最小値とする．従って，WAN 接続と判別した件数は，取得した 1 組の RTT 群の最小値が閾値の 7ms を超えた件数を示す．ping コマンドの実行回数を 1 回とする手法は，22%の確率で WAN 接続による接続と判別されることから，利便性の観点より不採用とした．ここで，1 回の ping コマンド実行には約 3.6 秒を要するため，ping コマンド実行回数を 3 回以上とする場合も，利便性の観点から不採用とした．これらの結果より，ping コマンド実行回数は 2 回が適切であるとの判断に至った．

表 5　ping コマンド実行回数の違いによる LAN 識別の誤り

Table 5　LAN identification error due to difference in the number of times the ping command is executed.

ping コマンド実行回数	WAN 接続と識別した件数
1 回	22 件
2 回	5 件

6.6 実験用 NW 環境の模倣

WAN 接続を模倣のために挿入する遅延は，表 4 に示す RTT 計測実験の結果より，グローバル IP アドレス宛の RTT の最小値とプライベート IP アドレス宛の RTT の最大値の差分である 13ms に決定した．

7. 考察

7.1 プロトタイプシステムにおける課題

(1) **RTT，TTL の閾値調節機能の実装**

プロトタイプシステムでは，通信経路を判別するための RTT および TTL の閾値が変更不可であり，宅内平均 RTT が 7ms を超過する，宅内にルータが 3 つ以上設置されている等の多種多様な通信環境に対応できない仕様となっている．これにより，LAN 接続であるにもかかわらずリカバリ処理が許可されない等の問題が想定される．

この解決策として，閾値の調節機能の実装等が挙げられる．なお，閾値調節については，該当環境における RTT および TTL の計測結果を基に行うことにより，多種多様な通信環境に対応することを目指す．ここで，上記の手法を実装した場合，LAN 接続と判別する範囲が拡大する可能性があり，外部の第三者に対してリカバリ処理を許可する恐れが高まる．このセキュリティ課題を解決するため，調節後の閾値が，プロトタイプシステムにて採用した閾値（RTT は 7ms，TTL は 2 件）を超過する場合に，他の認証方法と組み合わせる等の対策が必要となる．

(2) **PM 内 認証情報の削除動作を考慮した実装**

プロトタイプシステムにおける同期処理は，SIM で管理するバックアップデータの上書き動作を行うことから，PM 内が保持する認証情報を空の状態で同期した場合にバックアップデータを失う恐れがある．そのため，バックアップデータの消失防止を目的に，削除動作後の同期処理実行に関する権限制御の実装が望ましい．

7.2 インサイダー攻撃以外の脅威に関するセキュリティ課題

プロトタイプシステムでは，PM および SIM 間の通信における外部の第三者の侵入防御を目的とした機能を実装した．一方，H-GW（SIM を装着する機器）本体へのハッキング対策は講じられておらず，機器の乗っ取り等の従来の通信機器に対するセキュリティ課題には，別の対策を講じる必要がある．

8. おわりに

本研究では，ID やパスワード等の認証情報をパブリックなクラウドストレージを用いて管理する，クラウドベース PM におけるインサイダー攻撃の脅威に着目した．そして，宅内等の LAN 環境に設置する H-GW に装着した，耐タンパー性を有する SIM を，プライベートでセキュアなクラウドストレージとして利用するパスワード管理システムを提案した．プロトタイプの設計と実装を行い，RTT と TTL の計測結果に着目した通信経路の判別によって，外部の第三者に認証情報が取得されない等の要件を満たすことで，認証情報の管理の利便性と安全性の向上を実現することを確認した．今後は，7 章に整理した各種課題を解決可能なシステムの実現を目指す．

謝辞　本論文を執筆するにあたり，丁寧なご指導を賜り，ご協力いただいた KDDI 総合研究所の磯原隆将様をはじめとするサイバーセキュリティグループの皆様に謹んで感謝の意を表する．

Vol.2020-DPS-182 No.26
Vol.2020-CSEC-88 No.26
2020/3/12

参考文献

[1] Rui Zhao, Chuan Yue, Kun Sun. Vulnerability and Risk Analysis of Two Commercial Browser and Cloud Based Password Managers, 2013
[2] Oracle，Java Card の概要, https://www.oracle.com/webfolder/technetwork/jp/javamagazine/Java-ND17-JavaCard.pdf, (参照 2020-02-13).
[3] Smart Card based Robust Security System，K. Eswar Kumar, Ashok Kumar Yadav, Dr.T.Srinivasulu, https://translate.google.co.jp/?hl=ja#view=home&op=translate&sl=en&tl=ja&text=K.%20Eswar%20Kumar%2C%20Ashok%20Kumar%20Yadav%2C%20Dr.%20T.%20Srinivasulu
[4] IPA，スマートカードの安全性に関する調査 調査報告書, https://www.ipa.go.jp/security/enc/smartcard/sc.html, (参照 2020-02-13).
[5] NTT DOCOMO テクニカル・ジャーナル，Vol.22，No.4, https://www.nttdocomo.co.jp/binary/pdf/corporate/technology/rd/technical_journal/bn/vol22_4/vol22_4_005jp.pdf, (参照 2020-02-13).
[6] Sun Microsystems，Java Card Applet Developer’s Guide，1998
[7] IPA, IC・ID カードの相互運用可能性の向上に係る基礎調査, https://www.ipa.go.jp/files/000024611.pdf, (参照 2020-02-13).
[8] EternalWindows，スマートカード, https://www.google.com/search?client=firefox-b-d&q=APDU, (参照 2020-02-13).
[9] Oracle, An Introduction to Java Card Technology, https://www.oracle.com/technetwork/java/javacard/javacard1-139251.html, (参照 2020-02-13).
[10] MCPC，モバイルシステム技術テキスト,リックテレコム, 2018, p.175
[11] GlobalPlatform ，Requirements for NFC Mobile: Management of Multiple Secure Elements v1.0，2010
[12] KDDI 株式会社, 株式会社 KDDI 総合研究所, SIM を活用した IoT セキュリティ技術を開発, https://news.kddi.com/kddi/corporate/newsrelease/2016/10/20/besshi2106.html, (参照 2020-02-13).
[13] アリオン株式会社，DLNA 認証, https://www.allion.co.jp/certification/dlna-2/
[14] ソニー株式会社，嶋久登，DTCP-IP, https://www.ite.or.jp/contents/keywords/FILE-20111231152950.pdf, (参照 2020-02-13).
[15] MIT, clumsy 0.2, https://jagt.github.io/clumsy/download(参照 2020-02-18).