TEE 搭載デバイスにおける閲覧画面の偽装検出技術の提案

武藤 健一郎[†1]　山越 公洋[†1]

概要：通常の実行環境から分離されたセキュアなアプリケーション実行環境を提供する技術として Trusted Execution Environment (TEE) がある。TEE 搭載デバイスでは、通常の実行環境を介さずに入力操作や出力結果の画面表示を実行できる Trusted User Interface (TUI) と呼ばれる機能があり、TUI の実装として、専用のインジケータや、TEE 内の秘密情報を画面表示する方法が知られている。本稿では、専用のインジケータを持たないデバイスで課題となる画面の盗み見・推測への対策として、外部の信頼されたハードウェアを用いて閲覧画面の出力元を検証し、閲覧画面の偽装を検出できるようにする方式を提案する。

キーワード：Trusted Execution Environment, Trusted User Interface, 信頼の連鎖

Detection Method of Viewing Screen Forgery in a Device Supporting TEE

Kenichiro Muto[†1]　Kimihiro Yamakoshi[†1]

Abstract: Trusted Execution Environment (TEE) isolate Application Execution Environment from Rich Execution Environment (REE) for Secure Execution. In a device supporting TEE, Trusted User Interface provide I/O interface between TEE and User without I/O interface of REE. In this paper, We propose a detection method of viewing screen forgery in a device without hardware indicator by using secure elements. Device user, that verify viewing screen using the secure elements, can detect screen forgery.

Keywords: Trusted Execution Environment, Trusted User Interface, Chain of Trust

1. はじめに

インターネットにおける取引が社会に浸透し、我々の生活に欠かせないものとなっている。近年では、モバイル機器を利用した取引も広く行われており、IoT の拡大とともに、その機会と重要性は今後ますます拡大することが予想される。一方で、汚染されたアプリケーションや、取引画面を偽装するアプリケーション等、安全な取引を脅かす新たな脅威が日々発見されており、アンチウィルスソフト等による従来のセキュリティ対策では、これらの脅威を見逃してしまう可能性が懸念される。

このような背景の中、モバイル機器や組み込み機器内の OS 上のアプリケーション実行環境を分離することで、通常の実行環境とは異なる信頼された OS 上で、安全なアプリケーション実行環境を実現する技術が開発されている。近年では、ARM 社の TrustZone [1] をはじめとして、このような機構がハードウェアに搭載される製品も増加しており、近年急速に市場が拡大している IoT 機器のセキュリティ対策として期待が高まっている。

1.1 Trusted Execution Environment

安全なアプリケーション実行環境を提供する機構について、Global Platform では Trusted Execution Environment (TEE) とよばれるアーキテクチャを規定し、標準化を行っている[2]。ハードウェア上に、通常の環境（OS／アプリケーション）から分離されたセキュアな環境を構成し、マルウェア等の汚染されたアプリケーションが混入したとしても、分離された環境は保護される。

TEE では、通常の環境とセキュアな環境で、OS やアプリケーションの呼び方を区別している。通常の環境のことを Rich Execution Environment (REE) と呼び、セキュアな環境のことを Trusted Execution Environment (TEE) と呼ぶ。TEE 側では、REE 側の OS とは独立した Trusted OS と呼ばれる OS が動作し、REE 側の OS(Rich OS)とは独立した安全な環境で、アプリケーションを実行する。通常のアプリケーションは REE 側で実行させ、特に高いセキュリティが求められるアプリケーションを Trusted OS 上で実行させ、必要な機能を REE 側から呼び出して利用することによって、当該アプリケーションの安全性を向上させることができる。

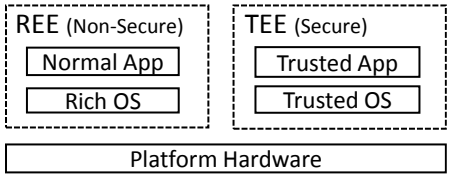

図 1　TEE によるセキュアなアプリケーション実行環境
Figure 1　Trusted Execution Environment

†1 NTT セキュアプラットフォーム研究所
NTT Secure Platform Laboratories, NTT Corporation.

1.2 Trusted User Interface

TEE を搭載したデバイスにおいて、TEE 側で動作するアプリケーションがユーザへの入出力を伴う場合、REE を介して入出力を行ってしまうと、入出力情報の改ざん・盗聴リスクが懸念される。例えば、銀行取引のアプリケーションでは、入力した振込先口座の改ざんや、暗証番号の窃取が懸念される。また、別の例としては、TEE 側にセキュリティログを保管し、そのログを閲覧しようとした場合に、「異常」という内容が含まれていたログレコードを閲覧した場合でも、画面出力上では「正常」という情報に改ざんされてしまうような事象が懸念される。

このようなリスクへの対策として、Trusted User Interface (TUI) と呼ばれるユーザインターフェースが考案されている[3]。TUI では、デバイスに搭載された TEE が入出力画面を占有し、REE を介さずに、情報の入力操作や出力処理を行う。これにより、TEE 側のアプリケーション（TA:Trusted Application）が表示した入力操作や閲覧情報を、TEE とユーザの間で、REE を介さずに直接やりとりができる。

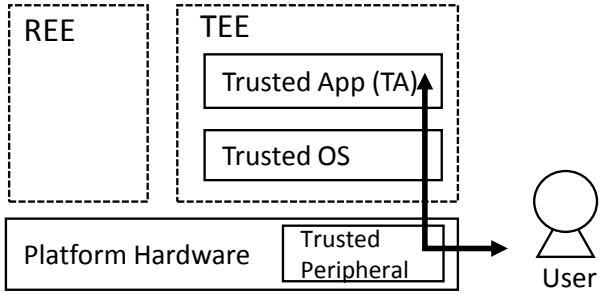

図 2　TUI による入出力の保護

Figure 2　Protection of I/O by Trusted User Interface

本稿では、上記 TUI に関するセキュリティ上の課題を述べ、信頼可能な外部デバイスを活用することにより、安全性を向上させることができる技術を提案する。以降、TUI を利用する際に発生が懸念されるセキュリティ事象と、その事象に対応するための従来技術について説明し、従来技術の課題について述べる。

2. 従来技術

ユーザによる閲覧画面が、もし TUI を介して出力された画面であれば、その画面を閲覧したユーザは、安全に TA との間でやり取りができる。しかし、以下のような事象が発生した場合、閲覧している入出力画面の安全性は確保されない。本稿では、以降、以下の事象を総称して「閲覧画面の偽装」と呼ぶことにする。

- REE に混入されたマルウェアが、TUI を模擬した偽の画面を作成・表示する
- TEE 搭載デバイスとは全く別のデバイス（TUI を模擬した偽の画面を作成・表示するデバイス）へのすり替えが発生し、すりかえられたデバイスが TUI を模擬した偽の画面を作成・表示する

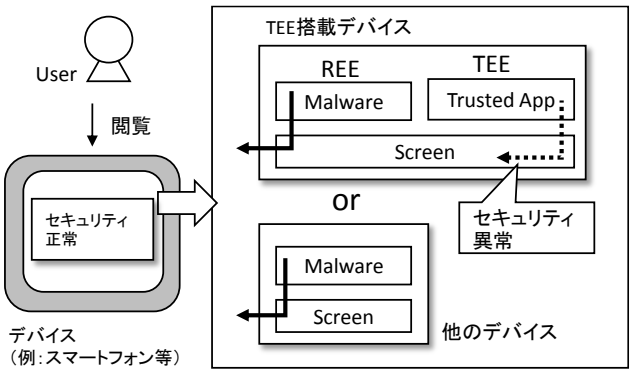

図 3　TEE 搭載デバイスにおける画面の偽装

Figure 3　Viewing Screen Forgery in a Device Supporting TEE

2.1 インジケータによる画面出力元の証明

閲覧画面の偽装への対策としては、画面を閲覧したユーザに対して、以下の２つの条件が成立することを証明する必要がある。

- 閲覧画面の出力元デバイスが、TEE 搭載デバイスであること
- 閲覧画面の出力元アプリケーションが、TEE にインストールされた正規の TA であること

これらを証明する従来技術として、TEE が搭載されたデバイス上で、インジケータを導入する方法が知られている。インジケータは、TA から TUI を介して画面出力を行っていることを示す機能を持ち、例えば、以下のような実装方法が知られている[3]。

(1) **専用の HW 部品の組み込み**

TEE が占有することができる専用の HW 部品（例：LED ランプ）を実装デバイスに組み込み、その部品をインジケータとする。TA が画面を占有している際に、その占有状態を、HW 部品の操作（例：点灯／消灯）によって示すことで、TA が画面出力を行っていることをユーザに証明する。画面出力とは別に、REE が操作できない（REE が占有不可能な）I/O を TA に操作させる方法であるため、閲覧画面を偽装しても、このインジケータ操作までは偽装できない。

(2) **秘密情報の表示**

ユーザのみが知りうる秘密情報（例：テキスト・画像）を TEE 側に予め登録しておき、TA が画面出力を行う際に、その秘密情報を画面に埋め込んで表示し、ユーザに閲覧させる。秘密情報は、TA からのみアクセスが可能な（REE からのアクセスが不可能な）データ保存領域に保存しておくことにより、閲覧画面を偽装しようとしても、秘密情報を埋め込んだ画面を作成し、ユーザに閲覧させることが難しいという前提に基づいた方法である。

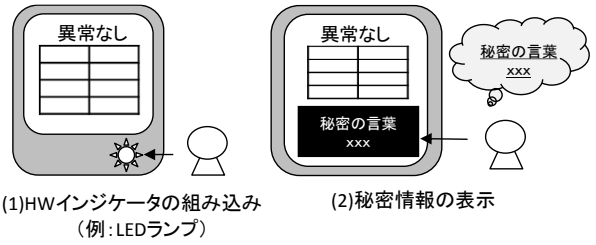

図 4　TUI におけるインジケータの実装例
Figure 4　Implementation of Security Indicator of TUI

2.2 取引認証による入力内容の証明

入出力画面の改ざんに関わる脅威として、MITB(Man-in-the-Browser)攻撃が知られている。MITB 攻撃では、サービス提供者とユーザ間のインターフェースとして利用される端末内のブラウザが、マルウェア等により汚染され、取引内容（例：送金先の口座情報）が改ざんされる攻撃である。代表的な対策として、サービス提供者が正当な取引である事を確認する仕組み（取引認証）があり、この仕組みを、TUI を介して実現する方式も検討されている[4]。

3. 従来技術の課題

(1) 専用の HW 部品を搭載しないデバイス

デバイスに組み込まれた専用の HW 部品を TEE に占有させる方法では、デバイスの製造段階でインジケータとする HW 部品を組み込む必要がある。出荷の段階でこのような HW 部品が組み込まれているデバイスでは有効な方法だが、必ずしも全てのデバイスに、TEE のための専用 HW 部品が組み込まれているとは限らない。また、インジケータを模倣した偽の HW 部品を組み込んだ全く別のデバイスを製造し、そのデバイスにすりかえる手口で閲覧画面の偽装が行われてしまうと、閲覧画面の偽装が検出できないまま、閲覧画面の表示内容をユーザが信用してしまう。そのため、インジケータの役割を担う HW 部品が組み込まれていないデバイスでは、閲覧画面の偽装を検出することができない。

(2) 秘密情報の覗き見・推測

ユーザのみが知りうる秘密情報を、TA だけがアクセス可能な領域に保管し、その情報を画面に埋め込んで表示する方法では、画面を閲覧したユーザへの可読性を確保する必要が生じる。しかし、可読性のある情報を画面に表示することになるため、画面に表示された秘密情報の盗み見に対する安全性を確保することができない。また、可読性のある情報である必要があるため、現実的には、簡単に推測可能な程度の情報しか運用できないケースも考えられる。このように、TEE 側に登録した秘密情報を攻撃者が容易に入手できる方法では、TA だけが出力できるはずの秘密情報を埋め込んだ画面を容易に偽造できてしまい、簡単に閲覧画面を偽装することができてしまう。例えば、REE 上のマルウェアによって秘密情報が埋め込まれた画面が表示された場合、ユーザは閲覧した画面上に秘密情報が表示されているため「TEE 搭載デバイスにインストールされている TA が表示した画面である」と誤認識してしまう。また偽造された画面を表示した別のデバイスに、ユーザに知られることなくすりかえられた場合、同様に偽造された画面を、ユーザに誤認識させることができてしまう。

(3) 画面閲覧によって終端されるサービス

取引認証は、サービス提供者が受信した取引内容を確認するための仕組みである。取引内容が改ざんされたとしても、サービス提供者でサービスが終端される場合には、例えばサービス提供者側からの別経路（電話など）での取引内容確認や、取引内容に対する電子署名を検証する等の手段により、入力した取引内容の改ざんを検出することができる。しかし、画面閲覧によって終端されるサービスの場合、取引認証では対策ができない。例えば、TEE 側に蓄積したセキュリティログを閲覧するようなサービスでは、閲覧画面が偽装されていたとしても、偽装を検出することができず、閲覧した内容を誤ってユーザが信用してしまう。

4. 提案技術

4.1 提案技術の要件

先に述べた３つの課題に対応するためには、TEE が占有可能な専用の HW 部品を搭載しないデバイスにおいて、秘密情報の盗み見や推測への安全性が確保できる方法で、閲覧画面の偽装（REE からの画面出力、及びすり替えたデバイスからの画面出力）を検出できるようにする必要がある。そこで、本稿では、提案技術が達成すべき要件として、以下の３つの要件を設定することとした。

- **要件１**
  閲覧画面の偽装を検出できること
- **要件２**
  専用の HW 部品をデバイスに組み込む必要がないこと
- **要件３**
  盗み見や推測への攻撃耐性を持つこと

4.2 提案技術の概要

本稿では、TEE 搭載デバイスとは分離された信頼されたデバイスを導入し、そのデバイスを利用して、画面の偽装を検出する方法を提案する。以降、TEE 搭載デバイスを「画面表示デバイス」と記載し、TEE 搭載デバイスとは異なる信頼されたデバイスを「画面検証デバイス」と記載する。

画面表示デバイスでは、ユーザに閲覧させる画面を TA が作成する。この際「REE からの出力画面ではないこと」を証明する情報を表示することで、閲覧した画面の出力元環境が REE だった場合に、検出できるようにする。

従来技術（2.1 (2)の方法）では「REE からの出力画面ではないこと」を証明するために、ユーザの知識に基づく秘密情報を利用していたが、前記で述べた盗み見や推測の課

題を解決する必要があるため、秘密情報の代わりに、以下の条件が成立することを証明するワンタイムコードを表示する。

- 画面表示デバイスが TEE 搭載デバイスであり、画面出力機能を持つ TA がインストールされていること (条件 a)
- 画面表示デバイスのスクリーン上に、TA から出力された画面を表示している状態であること (条件 b)

このワンタイムコードは、画面表示デバイスにおける画面表示の都度、画面検証デバイスへ発行要求を行うことにより取得する。発行要求を受けた画面検証デバイスでは、画面表示デバイスが上記の条件を充足するか否かを検証し、検証が成功した場合に限り、検証が成功したことを証明するワンタイムコードを発行し、発行したワンタイムコードを画面検証デバイスの画面上に出力する。その後、画面表示デバイスでは、TA からの出力画面に、画面検証デバイスから発行されたワンタイムコードを埋め込んで表示する。

この方法では、ユーザが閲覧した画面からワンタイムコードを読み取り、画面検証デバイスに表示されたワンタイムコードとの一致性を確認することで、閲覧した画面の出力元が TEE 搭載デバイスであり、かつ TA から出力された画面であることを、ユーザ自身で検証することができる。

もし、REE からの画面出力や、すり替えたデバイスからの画面出力によってユーザに閲覧させる画面が偽装されたとしても、攻撃者は画面検証デバイスに表示されているワンタイムコードと一致する情報を閲覧させることができず、閲覧画面の偽装が検出できる。

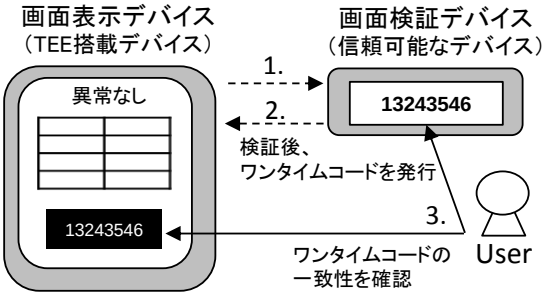

図 5　ワンタイムコードの一致性確認（イメージ）

Figure5　Collation of One-Time Code

4.3 提案技術の詳細

提案方式は、「画面表示デバイス」「画面検証デバイス」の二つのデバイスと「認証局」で構成する。本稿では、画面検証デバイスと認証局は信頼可能という前提を置く。以下では、提案方式の詳細を Step1～Step6 に分けて説明する。

Step1：準備

認証局では、コードサイニング証明書、および公開鍵証明書を発行する。発行準備のために、事前に鍵ペアを生成し、画面表示デバイスにインストールされる TA に、予めコードサイニング証明書を付与する。これにより、認証局は「画面表示デバイスにインストールされる TA は、画面出力機能を持つこと」を証明する。また、検証デバイスに生成した公開鍵を配布し、コードサイニング証明書や公開鍵証明書の検証鍵として利用できるようにしておく。

次に、画面表示デバイスで、コードサイニング証明書が付与された TA をインストールする。インストールされた TA は、インストール後ただちに鍵ペアを生成する。鍵ペアの内、秘密鍵は TEE 側で保持し、公開鍵は認証局へ送信する。公開鍵を受信した認証局は、例えば[5]のように TA のインストールを安全に管理する為のフレームワークなどを活用して「画面表示デバイスに TA がインストールされていること」を確認し、受信した公開鍵に対する公開鍵証明書を、画面表示デバイスに発行する。

上記の準備を実施することで、画面表示デバイスは 4.2 (条件 a) で述べた「画面表示デバイスが TEE 搭載デバイスであり、画面出力機能を持つ TA がインストールされていること」を、Step2 と Step3 により画面検証デバイスに証明できるようになる。

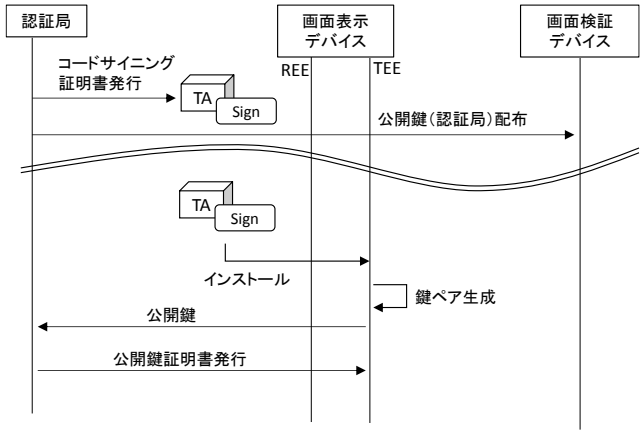

図 6　TA のインストールと鍵生成

Figure6　Install of Trusted Application and Key Generation

Step2：画面出力とワンタイムコードの発行要求

画面表示デバイスで、TA からの画面出力を行う。この際、TA から出力された画面を表示する前に、画面検証デバイスに対し、ワンタイムコードの発行を要求する。ワンタイムコードの発行を要求する際、Step1 で付与されたコードサイニング証明書、及び発行された公開鍵証明書を送信する。

Step3：画面表示デバイスの検証

ワンタイムコードの発行要求を受け付けた画面検証デバイスでは、まず、画面表示デバイスから受信したコードサイニング証明書と公開鍵証明書を検証する。ここで、コードサイニング証明書と公開鍵証明書を検証することによって 4.2 (条件 a) を確認し、検証が成功した場合に限り、

次の処理を進める。

次に、4.2 (条件 b)を確認する。そのために、まず、画面検証デバイスから画面表示デバイスへ認証要求を送信する。画面検証デバイスから認証要求を受け付けた画面表示デバイスは、TA が画面出力をした場合に限り、秘密鍵と認証要求をもとに認証応答を計算し、画面検証デバイスに返却する。なお、認証要求や認証応答の計算方法は限定しないが、例えば、電子署名やチャレンジレスポンス方式などの方法が考えられる。画面検証デバイスでは、返却された認証応答と公開鍵証明書をもとに、画面表示デバイスを認証し、認証が成功した場合には 4.2 (条件 b)が成立していると判定し、次の処理を進める。

Step4：ワンタイムコードの発行

画面検証デバイスで使い捨ての乱数を生成し、画面検証デバイスの画面に表示する。この乱数は、Step3 で実施した 4.2 (条件 a) (条件 b) の確認後に発行されるため、これらの２つの条件が成立したことを証明するワンタイムコードとして機能する。生成したワンタイムコードは、Step3 で検証した公開鍵証明書を用いて暗号化を行い、画面表示デバイスに送信する。

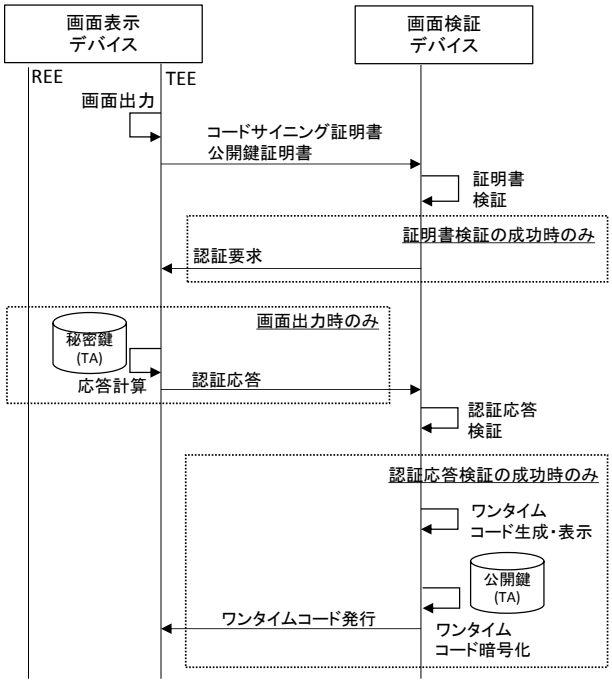

図 7　画面表示デバイスの検証とワンタイムコード発行

Figure7　Verification of Screen Display Device and Issuance of One-Time Code

Step5：ワンタイムコードの画面埋め込みと画面表示

画面表示デバイスで、暗号化されたワンタイムコードを受信し、TEE 側で保持している秘密鍵で復号する。そして、復号されたワンタイムコードを、Step2 で TA が出力した画面に埋め込み、画面表示デバイスに表示する。ここで、暗号化されたワンタイムコードは、要求元デバイスの TA だけが復号できるため、復号したワンタイムコードを TA からの出力画面に埋め込むことによって、4.2 (条件 a) (条件 b) が成立していることを、画面表示デバイスからユーザへ証明することができる。

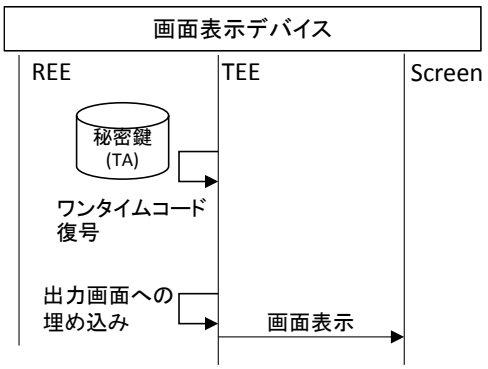

図 8　ワンタイムコードの表示

Figure8　Display of One-Time Code

Step6：ワンタイムコードの検証

ユーザは、画面表示デバイスと画面検証デバイスの両方の画面を閲覧し、画面表示デバイスのワンタイムコードが、画面検証デバイスのワンタイムコードと一致することを確認する。ワンタイムコードは、4.2 (条件 a) (条件 b) の２つの条件が成立したことを示す情報であるため、２つのコードが一致した場合には、ユーザはこの２つの条件が成立したことを確認することになる。

もし、画面検証デバイスにワンタイムコードが表示されていない場合、4.2 (条件 a) (条件 b) のいずれかが示されない。また、画面表示デバイスのワンタイムコードが画面検証デバイスと一致しない場合、画面閲覧を行ったデバイスは 4.2 (条件 a) (条件 b) を示すデバイスではないことになる。この性質を利用して、デバイス利用者が閲覧している画面が、偽装された画面（REE からの画面表示、又は、すり替えられた画面表示デバイスからの画面表示）であった場合に、ユーザがその画面偽装を検出することができる。

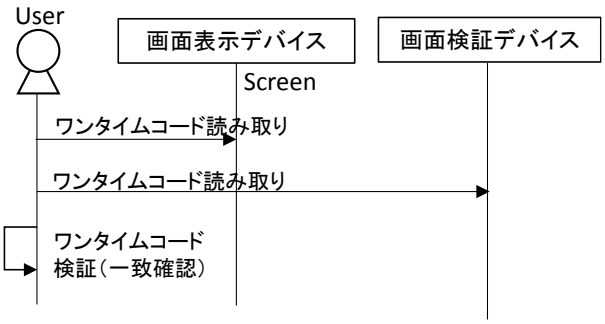

図 9　ワンタイムコードの検証

Figure9　Verification of One-Time Code

5. 考察

閲覧画面を偽装された場合、従来技術では、その偽装をユーザが検出することができない場合があった。この課題に対し､4.1 では提案技術が充足すべき要件を設定したが、ここではその要件充足性を考察する。

5.1 要件１：閲覧画面の偽装の検出可否

閲覧画面の偽装について、すりかえられたデバイスからの画面出力と、REE からの画面出力の２つのケースに分けて考察を行う。詳しい考察は以下に述べるが、結論としては､提案方式は 4.1 の「要件１」を充足する結果となった。

Case1：すりかえられたデバイスからの画面出力

提案方式では、画面検証デバイスにおいて、画面表示デバイス内の秘密鍵を利用した認証を行っており、ワンタイムコードの発行要求を受けても、認証が成功しないデバイスに対してはワンタイムコードの発行を行わない。そのため、画面表示デバイスとは別のデバイスでは、画面検証デバイスに表示されたワンタイムコードと一致する情報を、画面に出力することができない。従って、すりかえられたデバイスからの画面出力において、画面検証デバイスと一致する情報を埋め込んだ画面を出力することが困難である。以上のことから、提案方式は、すりかえられたデバイスで偽装された画面を出力しようとする攻撃に対し、耐性をもつ方式であるといえる。

Case2：REE からの画面出力

提案方式では、画面検証デバイスからワンタイムパスワードを発行する際、画面表示デバイスの TA で作成した秘密鍵と対になる公開鍵証明書を用いて暗号化する。この暗号化されたワンタイムコードは、画面表示デバイスの TEE 内に存在する秘密鍵以外では復号ができない。従って、仮に REE が暗号化されたワンタイムコードを取得したとしても、ワンタイムコードを復号し、画面検証デバイスと一致する情報を埋め込んだ画面を出力することが困難である以上のことから、提案方式は、REE 上のアプリケーションで偽装された画面を出力しようとする攻撃に対し、耐性をもつ方式であるといえる。

5.2 要件２：専用の HW 部品の組み込み要否

提案方式では、画面表示デバイスの他に、画面検証デバイスを必要とするが、画面表示デバイス側では、専用 HW 部品を組み込む必要はなく、単に画面上にワンタイムコードを表示するだけよい。以上のことから、専用の HW 部品が組み込まれていないデバイスにも適用可能な方式であり、提案方式は 4.1 の「要件２」を充足する。

5.3 要件３：盗み見・推測への攻撃耐性

提案方式では、画面表示デバイスに表示する情報は、推測困難なワンタイムコードであり、画面表示の度に更新される情報である。そのため、仮に攻撃者がワンタイムコードを埋め込んだ画面を偽造し、ユーザに閲覧させたとしても、そのワンタイムコードは既に意味をなさない情報となっており、画面検証デバイスを用いて検証することができないため、盗み見に対する攻撃耐性をもつ方式であるといえる。従って、提案方式は 4.1 の「要件３」を充足する。

表 1　要件に対する充足性

Table 1　Fulfillment of Requirements

	要件１	要件２	要件３
提案技術	○	○	○
従来技術			
(1) 専用 HW 部品組込	△*1	×	○
(2) 秘密情報表示	△*2	○	×
(3) 取引認証	△*3	○	○

（凡例：○…充足　△…一部充足　×…不足）

*1 偽造された HW 部品は検出不可

*2 秘密情報の攻撃耐性に依存

*3 画面閲覧で終端するサービスでは検出不可

6. おわりに

TEE 搭載デバイスにおいて、外部の信頼されたハードウェアを用いて閲覧画面の出力元を検証し、閲覧画面の偽装を検出できるようにする方式を提案した。TEE 搭載デバイスにおいて､TA とユーザ間で直接やりとりを行うためのインターフェース(TUI)において、従来の方法では、専用の HW 部品が組み込まれていないデバイスにおいて、TEE 内の秘密情報を画面表示する方法や取引認証を行う方法では、出力画面の偽装が検出できない課題を述べた。

本稿ではこの課題に対し、達成すべき３つの要件を設定し、外部の信頼されたハードウェアにワンタイムコードを発行させ、そのワンタイムコードを閲覧画面に組み込んで表示することで、画面を閲覧したユーザが、閲覧画面の偽装を検出できるようにする方式を提案した。さらに、提案方式に対し、設定した３つの要件の充足性を考察し、全ての要件が充足可能であることを確認した。考察結果より、提案方式は、TEE が占有可能な専用の HW 部品を持たないデバイスにおいて、秘密情報の盗み見や推測への安全性が確保できる方法で、閲覧画面の偽装を検出する効果的な方法であると結論付ける。

参考文献

[1] Arm, “TrustZone for Cortex-A”, https://www.arm.com/why-arm/technologies/trustzone-for-cortex-a (参照 2018-08-20).

[2] Global Platform, “TEE System Architecture Version 1.1”, January 2017.

[3] Global Platform, “Trusted User Interface Version 1.0”, June 2013

[4] 清藤 武暢, “暗号ハードウェア等に対するセキュリティ評価および留意点”, 日本銀行金融研究所 情報セキュリティ・シンポジウム, 2016 年 3 月 2 日, https://www.imes.boj.or.jp/citecs/symp/17/ref3_seito.pdf (参照 2018-08-20).

[5] Global Platform, “TEE Management Framework Version 1.0”, November 2016.