Vol.2017-MBL-85 No.24
Vol.2017-ITS-71 No.24
2017/11/16

物理的接触を根拠とした IoT デバイスのためのアクセス制御手法

遠藤徹[†1]　武田敦志[†1]

概要：通信機能付きセンサデバイスなどの IoT (Internet of Things) デバイスが様々な場所で利用されている．通常，スマートフォンなどの操作端末からインターネットなどの通信環境を経由して IoT デバイスの制御を行うため，従来の IoT デバイスではパスワードの設定や操作端末の登録などのセキュリティに関する設定が必要となる．本稿では，セキュリティに関する設定を不要とする IoT デバイスを実現するため，操作端末と IoT デバイスの物理的接触に基づいて操作端末からのアクセスの可否を判断する IoT デバイスのためのアクセス制御手法を提案する．提案手法では，QR コードや電子タグなどの物理的に近くにいる利用者のみが確認できるデータをアクセス権限の根拠とすることにより，IoT デバイスに接触可能な利用者の操作端末からのアクセスのみを許可するアクセス制御を実現する．本稿では，提案手法の詳細を説明し，その安全性について考察する．また，物理的接触を確認する方法として QR コードを用い，通信ネットワークとして無線 LAN を用いたプロトタイプ実装について述べる．

キーワード：Internet of Things，アクセス制御

Access Control Method based on Physical Contact for IoT Devices

TOHRU ENDO[†1]　ATSUSHI TAKEDA[†1]

Abstract: IoT (Internet of Things) devices such as sensor devices with communication capabilities are used for various purposes. Generally, data for access control such as passwords or control device IDs must be input to IoT devices because IoT devices are controlled by control devices such as smartphones, tablets or laptops via sensor networks, local area networks or the Internet. In this paper, we propose a new access control method based on physical contacts for IoT devices. The proposed access control method does not need any input data for access control. Instead of the access control data such as password, the proposed method uses short-range communication such as QR codes or electric tags. By using the short-range communication, IoT devices recognize control devices that are near the IoT devices physically, and the proposed access control method permits only the near control devices to access the IoT devices. In this paper, we describe the details of the proposed access control method and discuss its safety. In addition, we show a prototype implementation of the proposed method which uses QR codes for recognizing physical contacts and wireless local area networks for communication.

Keywords: Internet of Things, Access Control

1. はじめに

近年，センサデバイスやモータデバイスに通信機能が搭載され，ユーザがデバイスから情報を取得したり，その情報に基づいてデバイスを操作したりすることができるようになった．このようなセンサデバイスやモータデバイスなどは IoT デバイスと呼ばれ，家庭用セキュリティカメラやリストバンド型健康管理デバイスなどとして，既に商品化されているものもある．多くの IoT デバイスは，製造コストや消費電力量を抑えるため，単純な入力装置や出力装置しか持たない．そのため，これらの IoT デバイスを制御するためには，スマートフォンなどの操作端末から，Bluetooth や無線 LAN などのネットワークを介して制御データを送信する必要がある．IoT デバイスは操作権限を有する端末からの制御データにのみ従う必要があるため，IoT デバイスにはアクセス制御の仕組みが必要不可欠である．

IoT デバイスで用いられるアクセス制御の仕組みとして，従来はパスワードや PIN コードなどの秘密のデータを共有する仕組みが利用されてきた．この従来手法では，IoT デバイス側に予め秘密のデータを登録しておき，この秘密のデータを送信した制御端末にのみ IoT デバイスを制御する権限を与える．しかし，膨大な数の IoT デバイスが利用されるようになったため，これらの IoT デバイスに登録されている秘密のデータのすべてを管理することが困難になってきている．さらに，この従来のアクセス制御の手法では，すべての IoT デバイスに秘密のデータを登録しなくてはならないため，IoT デバイスの数が増えるにしたがって，秘密のデータの登録コストも増加するという問題がある．そのため，近年，初期パスワードを変更せずに IoT デバイスを使用し，その IoT デバイスを悪意のあるユーザに乗っ取られるという問題が発生している．

そこで本稿では，物理的な接触の検出をセキュリティの担保と捉え，スマートフォンと IoT デバイスとの通信において，ID やパスワードではなく，物理的な接触を根拠とした端末認証手法を提案する．物理的に IoT デバイスと接触できるユーザは，ネットワーク経由でのアクセスを拒否さ

†1 東北学院大学教養学部情報科学科
Department of Information Science, Tohoku Gakuin University

れたとしても，その IoT デバイスを物理的に操作可能である．そこで，提案手法では，物理的に接触可能な端末のみに対して IoT デバイスの操作権限を与える．例えば，子供部屋に設置された家庭用セキュリティカメラに物理的に接触できるのは，玄関の鍵を開けることができるその家の住人に限られる．このような環境では，ID やパスワードのような秘密のデータを用いる認証手法よりも，物理的な接触を検出することでアクセス可否を判断する方が安全である．具体的には，NFC や QR コードといった，データの読み取りに物理的な接触を必要とするマーカーに IoT 機器の識別データを記録しておき，これら識別データの読み取れた制御端末を物理的に接触可能な端末と判断する．その後，公開鍵暗号方式と共通鍵暗号方式を用いてスマートフォンと IoT デバイスが相互に識別・認証を行い，安全にスマートフォンと IoT デバイスとの暗号通信路を確立する．また，本稿では，モニタや入力機器を持たない IoT デバイスを用いた提案手法のプロトタイプ実装について述べる．本実装では，QR コードをスマートフォンアプリケーションで読み取り，IoT デバイスに見立てた Raspberry Pi 上の Web サーバから Web ページを取得する．本実装により，本稿の提案手法が実現可能であることを示す．

以下，2 章で認証手法に関する関連研究について，3 章で本提案手法の詳細について，4 章で本提案手法の実装についてそれぞれ述べ，最後に 5 章でまとめとする．

2. 関連研究

認証に関する研究では，シングルサインオンを用いた手法[7][8]に代表される，ネットワークへのアクセス制御を目的とした研究が多い．この手法では，認証用サーバとして設置された Identity Provider（IdP）がユーザ情報を管理するため，ユーザの利用するサービスが増えても，登録する秘密のデータを増やす必要がないという利点がある．しかし，これらの研究におけるシングルサインオン機能では，いくつもの個別のサービスにユーザ登録をして，サービスごとに認証を行う煩雑さは回避できるが，認証用サーバへのユーザ登録は依然として必要であり，ID とパスワードを一切使用しない認証は実現できない．いずれの研究[7][8]でも，IdP へのユーザ登録が必要である．

一方，スマートフォン等のモバイル端末を用いて認証をパスワードレス化する研究も近年多く報告されており，センサ類を使用したもの[1]，ジェスチャやタッチ操作を用いたもの[2][3]，これらの両方を用いたもの[4]などがある．これらの研究は，センサや個人の癖を含む生体情報から個人を特定しようとする試みである．また，通信に使うためのセキュリティを確保する手段として，FIDO 認証を用いた研究[5]がある．FIDO 認証では，認証動作に生体情報などを用いることでパスワードレス化を図ることができるとともに，生体情報などのクレデンシャル情報そのものは通信路に流さないため，セキュリティの向上も実現できる．さらに，FIDO 認証と併せて物理的接触を行い，IoT 機器の操作に成功した研究[6]がある．この研究では，FIDO 認証機能が搭載されたスマートフォンを使い，IoT 機器上の NFC リーダに接触させることで IoT サービスからの認証を受け，電子錠を解除させる．

これらの研究では，いずれも認証時に ID やパスワードを用いないという点では本稿の目的との共通点を見いだせる．しかし，予め認証データや端末証明書をスマートフォンや認証サーバに登録しておく必要があり，例えば，研究[1]ではスマートフォンアプリの練習画面で収集したデータを基に認証を行う．研究[2]ではスマートフォンアプリ上でプロファイル文書の文章を入力し，キーストロークデータを収集している．研究[3]では，ログイン後のタッチやスワイプ操作について，予め収集されたデータとの比較で不正ユーザかどうかを判定する．研究[4]でも，予め登録されたジェスチャと入力されたジェスチャとの比較で認証を行う．研究[5]では，ユーザの認証に指紋認証と PIN 番号を用いており，これらの認証データを Client 端末に登録しておかなくてはならない．研究[6]では，通信路の確立の際に確認する IoT 機器の真正性を，スマートフォンに予め登録されている IoT 証明書で確認している．

このように，シングルサインオンを利用した手法では，ユーザ情報やパスワードを IdP に事前に登録しておかなければならない．また，生体認証や FIDO 認証を利用した手法では，ID やパスワードの登録は不要であるが，生体情報をスマートフォンや認証用のサーバに登録しておかなければならない．これらの手法を用いた場合，利用する IoT デバイスの全てについて，それぞれにユーザ情報や秘密のデータを管理する必要がある．そのため，膨大な数の IoT デバイスを利用する場合，これらの手法を用いることは非現実的である．

本稿で研究対象とするのは物理的接触に基づく端末認証手法である．本稿の提案手法で必要な設定は，通信路となるネットワークの設定のみであり，それ以外の端末情報やユーザ情報を事前に登録しておく必要はない．そして，ユーザが操作端末を用いて IoT デバイスのマーカーと物理的に接触するだけで，その操作端末のみがアクセス可となるアクセス制御を実現する．

3. 提案手法

3.1 提案手法の概要

本稿の提案手法が想定する環境を図 1 に示す．提案手法が想定する環境では，複数の IoT デバイスと複数の操作端末が存在する．それぞれの関係は以下の通りとなる．

- それぞれの IoT デバイスと操作端末には，お互いの情報（IP アドレスやユーザ情報）を登録していない．
- ユーザは複数の操作端末を利用して複数の IoT デバイス

を制御する．

- IoT デバイスや操作端末との間に，何らかの通信路（有線／無線 LAN や Bluetooth，ZigBee など）が存在する．

以上の環境において，操作端末は IoT デバイスと物理的接触を行い，IoT デバイスの識別・接続・認証を行った後，IoT デバイスとの暗号化されたセキュアな通信路を確立する．この通信路を確立するための動作手順を図 2 に示す．

この環境では，IoT デバイスに接続するために必要となる IP アドレスなどの情報を操作端末側には登録していない．そのため，操作端末は，複数の IoT デバイスの中から通信相手の IoT デバイスを識別する必要がある．この機能を実現するため，操作端末側には，マーカーを読み取ることで接続先のデバイス情報を取得し，接続先を決定する仕組みを導入する．提案手法では，QR コードや NFC などのマーカーを物理的接触の根拠となる記録媒体として想定しており，このマーカーの読み取りを①物理的接触と定義する．さらに，通信したい IoT デバイスを識別して接続を確立する仕組みを作り，この通信を②接続要求・③接続許可と呼ぶことにする．接続が確立されたことの確実性については，操作端末側は③接続許可を，IoT デバイス側は後述の④認証要求を受信した時点で確認できる．その後，操作端末は接続許可を取ったある 1 つの IoT デバイスと④認証要求・⑤認証許可をそれぞれ送受信し，操作端末が実際に①物理的接触を経ていることを確認する．

3.2 提案手法の動作手順

提案手法では，物理的な接触を行った操作端末に対してのみ IoT デバイスとの通信を許可する．この機能を実現するため，操作端末と IoT デバイスは以下①～⑤の通信処理を実行する．

① 物理的接触

最初に，操作端末でマーカーを読み取り，接続先の IoT デバイス情報を取得する．この動作を①物理的接触と定義する．マーカーには予め接続先の IoT デバイス情報として，接続先の IoT デバイスを決定するために，②接続要求の通信で使用する識別子と④認証要求の通信で使用する認証 PASS を記録しておく．これらの識別子と認証 PASS は，物理的な接触以外の手段で操作端末に取得されてはならない．

識別子は操作端末が接続先の IoT デバイスを探索するためのものであるため，IoT デバイスごとに固有の値にすべきである．そのため多くの場合，後述する③接続許可の通信で使用する公開鍵から算出したハッシュ値を利用することができる．ハッシュ値を算出する際には一方向ハッシュ関数を用いて，元の公開鍵が復元されないようにする必要がある．IoT デバイスを特定するためのデータとしては，公開鍵のハッシュ値以外にも様々な種類の識別子が考えられるが，これ以降本稿では公開鍵のハッシュ値を利用するものとする．

認証 PASS は IoT デバイス上のサービスにアクセスするためのパスワードであり，サービスごとに固有の値にすべきである．これにより，1 つの IoT デバイス上にある複数のサービスそれぞれが，操作端末に①物理的接触の動作を強制させることができる．そのため，この①物理的接触の動作には，悪意あるボットよる不正ログインやスパムなどを防止する CAPTCHA 認証の役割も与えることができる．

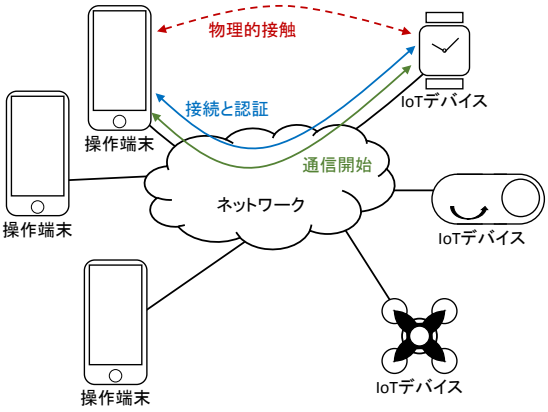

図 1　操作端末と IoT デバイスの関係

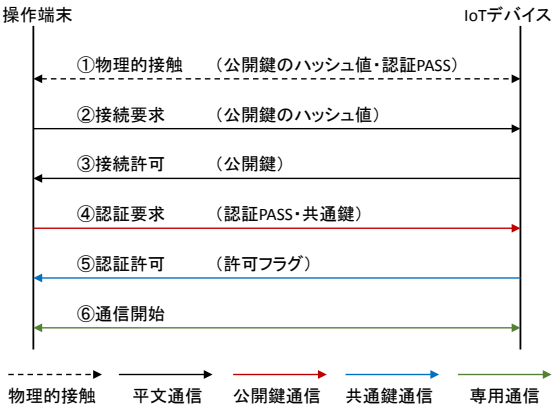

図 2　各通信の種類と内容

② 接続要求

この②接続要求の通信は，複数の IoT デバイスに対して一斉に平文通信で行われる．この通信では，操作端末側から IoT デバイス側に接続を要求する通信を行う．この時，IoT デバイスを識別するために，①物理的接触の動作でマーカーから読み取った情報のうち，公開鍵のハッシュ値を送信して接続先を特定する．IoT デバイス側では，デバイス上の公開鍵からハッシュ値を計算して，受信したハッシュ値と比較する処理が行われる．

③ 接続許可

これ以降の全ての通信は，操作端末と IoT デバイスとの 1 対 1 で行われる．この③接続許可の通信では，IoT デバイ

ス側から操作端末側に接続を許可する平文通信を行う．この時，IoT デバイス側は，接続要求のあった操作端末に対し，公開鍵を送信する．操作端末は公開鍵を受信し，ハッシュ値を計算して QR コード上のハッシュ値と比較する処理を行う．

④ 認証要求

この④認証要求の通信では，公開鍵暗号方式を用いて，操作端末側から IoT デバイス側に認証を要求する通信を行う. 操作端末は, まず共通鍵を発行する. その後, 認証 PASS と共通鍵を公開鍵によって暗号化し，暗号データとして操作端末から IoT デバイスに送信する．IoT デバイスは受信した暗号データを秘密鍵で復号し，認証 PASS をチェックする処理を行う．

⑤ 認証許可

この⑤認証許可の通信では，共通鍵暗号方式を用いて，IoT デバイス側から操作端末側に認証を許可したことを知らせる通信を行う．この通信で用いる許可フラグとなる具体的なデータの種類は，サービスによって異なることが想定される．例えば，Web ページを閲覧するためのサービスなら URL を, 何らかの発券サービスなら整理番号をこの通信で送受信できる．この後，許可フラグを受信した操作端末は⑥通信開始となり，専用通信上で IoT デバイスからのサービスを利用することになる．

3.3 提案手法の検証

この節では，上述した動作ごとに，①物理的接触の動作を経ていない操作端末や不正な IoT デバイスによる中間者攻撃を受けた場合の本提案手法における通信内容の安全性について，盗聴・改変・なりすましの 3 つの観点から検証する．

① 物理的接触

この動作では，ネットワークにアクセスできない操作端末や，物理的にマーカーを読み取ることのできない操作端末が割り込むことはできない．したがって，ネットワークを介した盗聴・改変・なりすましのいずれも実行することはできない.

② 接続要求

この通信では，不正ユーザによる盗聴が発生する可能性がある．しかし，公開鍵のハッシュ値を盗聴されたとしても，①物理的接触の動作を経ていなければ④認証要求の通信で認証 PASS を送信できないため，⑤認証許可の通信まで到達することができない．そのため，この通信で発生する盗聴は，本稿の提案手法における安全性を低下させるものではない. また, この通信でデータが改変された場合は, IoT デバイス上での識別子のチェックに失敗するため，③接続許可の通信まで到達することはできない．さらに，①物理的接触の動作を経ていない操作端末がなりすましをしようとしても，④認証要求の通信で認証 PASS を送信できないため, ⑤認証許可の通信まで到達することはできない. 逆に，悪意ある IoT デバイスがなりすましをしようとした場合には，受信したハッシュ値から元の公開鍵を復元できないため，③接続許可の通信で公開鍵を返信できない．また，予め操作端末になりすまして③接続許可の通信で IoT デバイスから公開鍵を取得した後に，IoT デバイスになりすまして②接続要求の通信で操作端末からの通信を妨害しようとしても，①物理的接触の動作を経ていなければ④認証要求の通信で認証 PASS を送信できないため，⑤認証許可の通信まで到達することはできない.

③ 接続許可

この通信でも，不正ユーザによる盗聴が発生する可能性がある．しかし，前節の②接続要求の通信と同様に，①物理的接触の動作を経ていなければ④認証要求の通信で認証 PASS を送信できないため，⑤認証許可の通信まで到達することができない. そのため, この通信で発生する盗聴は, 本稿の提案手法における安全性を低下させるものではない. また，この通信でデータが改変された場合は，操作端末上で行われる公開鍵のハッシュ値をチェックする処理に失敗するため, ④認証要求の通信まで到達することはできない. さらに，悪意のある IoT デバイスがなりすましをしようとしても，秘密鍵を持っていないため④認証要求の通信で受信した暗号データを復号できず，共通鍵を取り出せないため⑤認証許可の通信で送信する許可フラグを暗号化できない．逆に，悪意ある操作端末がなりすましをしようとした場合には，①物理的接触を経ていなければ④認証要求の通信で認証 PASS を送信できないため，⑤認証許可の通信まで到達することはできない.

④ 認証要求

この通信では公開鍵暗号方式を用いているため，不正ユーザによる盗聴が発生しても内容が把握されることはない. また，この通信でデータが改変された場合は，IoT デバイス上での秘密鍵による暗号データの復号に失敗するため，⑤認証許可の通信まで到達することはできない．さらに，悪意ある操作端末がなりすましをしようとしても，①物理的接触を経ていなければ認証 PASS を送信できないため，⑤認証許可の通信まで到達することはできない．逆に，悪意ある IoT デバイスがなりすましをしようとしても，秘密鍵を持っていないため受信した暗号データを復号できず，共通鍵を取り出せないため⑤認証許可の通信で送信する許可フラグを暗号化できない.

⑤ 認証許可

この通信では共通鍵暗号方式を用いているため，不正ユーザによる盗聴が発生しても内容が把握されることはない. また，この通信でデータが改変された場合は，操作端末上での共通鍵による暗号データの復号に失敗するため，⑥通信開始の動作まで到達することはできない．さらに，悪意ある IoT デバイスがなりすましをしようとしても，④認証要求の通信で取得するはずの共通鍵を持っていないため許

可フラグを暗号化できず，⑥通信開始の動作まで到達することはできない．逆に，悪意ある操作端末がなりすましをしようとしても，共通鍵を持っていないため受信した暗号データを復号できず，許可フラグを取り出せないため⑥通信開始の動作まで到達することはできない．

4. 実装

4.1 システム構成

本提案手法が実現可能であることを示すために，プロトタイプモデルの実装を行った．操作端末として Android 6.0 搭載のスマートフォンを，IoT デバイスとして Raspbian Stretch 搭載の Raspberry Pi 3 を用いた．本実装では，スマートフォンから Raspberry Pi 上にある閲覧制限がかかった Web ページを閲覧することを目的とする．はじめ，Web ページはどのスマートフォンに対しても閲覧不可に設定されており, Raspberry Pi 側でスマートフォンが①物理的接触の動作を経ていることを確認できれば閲覧制限を解除し，そうでなければ閲覧不可の状態を維持する. Raspberry Pi 上の Web サーバには Apache 2.4 を用いた．物理的な接触の根拠となるマーカーには QR コードを採用し，スマートフォンアプリケーションに①物理的接触の動作を行う機能を実装した．また，3.1 節で述べた想定する環境を基に，スマートフォンと Raspberry Pi の間には通信路があるという前提で，それぞれを予め無線 LAN に接続しておく．さらに，複数台の IoT デバイスが関係する環境として Raspberry Pi を 2 つ用意した．この 2 つの Raspberry Pi は，公開鍵・秘密鍵・提供する Web ページ・Web ページに対応する QR コードが異なる．同様に，IoT デバイスが複数のサービスを提供していることを想定し，Web ページをそれぞれの Raspberry Pi に 2 つずつ，合計 4 つ用意した．それぞれの Web ページごとに認証 PASS を設定し，対応する QR コードもそれぞれに発行した．以上で述べた本実装のシステム構成を図 3 に示す．

4.2 実装内容

ここでは，本実装の具体的な内容について，3.2 節で示した通信の構成を基に説明する．

① 物理的接触

本実装における①物理的接触の動作は，スマートフォンで QR コードを読み取ることとした．QR コードには，②接続要求の通信で用いる公開鍵のハッシュ値と，④認証要求の通信で用いる認証 PASS が記録されている. QR コードの仕様は，バージョン 9（53x53 セル）で，誤り訂正レベル H（30%）のものを用いた．この中に，512bit の情報を，Base64 変換を用いて 88 文字の文字列として記録しておく．512bit の情報の内訳は，公開鍵のハッシュ値が 256bit，認証 PASS が 256bit である．また，公開鍵のハッシュ値を導出するアルゴリズムには，SHA-256 を用いた．これは，SHA-256 が一方向ハッシュ関数であり, かつ SHA-1 などの

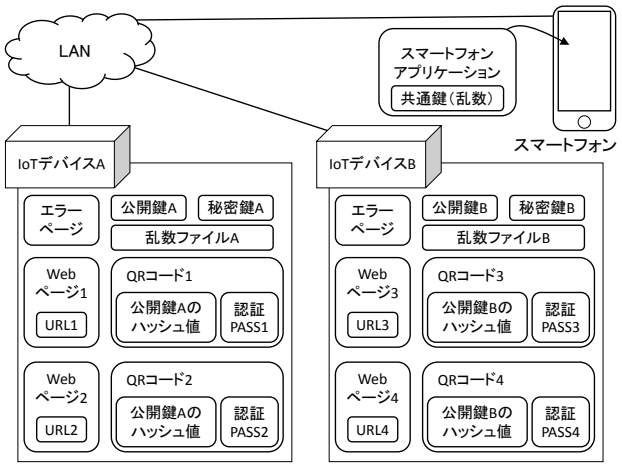

図 3　本実装の構成

意図的にコリジョンを起こすことができるハッシュ関数よりも強固であると考えたからである．さらに，スマートフォンアプリケーションの実装内容として，撮影した画像を加工するのに OpenCV for Android を，画像から QR コードの内容をデコードするのに ZXing をそれぞれ用いた．

② 接続要求

本実装における②接続要求の通信では, Raspberry Pi 側は UDP での受信待ち状態から開始する．そしてスマートフォンから，①物理的接触の動作で取得した公開鍵のハッシュ値が無線 LAN 内にブロードキャストされるのを待つ．公開鍵のハッシュ値を受信したら，取得したハッシュ値とデバイス内の公開鍵から導出したハッシュ値を比較し，チェックに失敗した場合は UDP での受信待ち状態に戻り, 成功した場合は次の③接続許可の通信を行う．スマートフォン側は，UDP で無線 LAN 内に公開鍵のハッシュ値をブロードキャストする．その後，③接続許可の通信を行う．

③ 接続許可

本実装における③接続許可の通信では，スマートフォン側は UDP での受信待ち状態から開始し，Raspberry Pi 側から公開鍵が送信されるのを待つ．公開鍵を受信したら，取得した公開鍵から導出したハッシュ値と①物理的接触の動作で取得したハッシュ値とを比較し，このチェックに失敗した場合は接続を中断する．成功した場合は次の④認証要求の通信を行う．Raspberry Pi 側は，予めペアとなる公開鍵と秘密鍵を持っており, このうち公開鍵を UDP でスマートフォンに送信する. その後, 次の④認証要求の通信を行う．また，公開鍵暗号方式のアルゴリズムには RSA を用いた．

④ 認証要求

本実装における④認証要求の通信で送受信される共通鍵には，スマートフォン側で発行する乱数を使用する．この乱数を Raspberry Pi 上の乱数ファイルに登録することで，スマートフォンは Raspberry Pi 上の Web ページにアクセスできるようになる．具体的には，乱数ファイルに 1 つの乱

数が記録されており，スマートフォンが Web ページを読み込む際の URL パラメータとしてこの乱数が使用される．全ての Web ページはデフォルトでアクセス不可に設定されており，乱数ファイルに登録された乱数が URL パラメータの乱数部分と一致した場合にアクセス可と判断する．

本実装における④認証要求の通信では，Raspberry Pi 側は UDP での受信待ち状態から開始する．そしてスマートフォンからの公開鍵によって暗号化された認証 PASS と乱数を含む暗号データを受信する．暗号データを受信したら秘密鍵で復号し，認証 PASS をチェックする処理を行う．このチェックに失敗した場合は接続を中断する．このチェックに成功した場合はスマートフォンが発行した乱数を乱数ファイルに登録し，次の⑤認証許可の通信を行う．スマートフォン側は，乱数を生成し，認証 PASS→乱数の順に整形された文字列を公開鍵で暗号化し，暗号データとして Raspberry Pi に送信する．その後，⑤認証許可の通信を行う．

⑤ 認証許可

本実装における⑤認証許可の通信で用いる許可フラグには，通信プロトコルとして Web ページの URL 用いた．この通信は，本実装では平文で行われる．本来，⑤認証許可の通信では共通鍵暗号方式を用いた暗号化通信が行われるが，本実装では Web ページがデフォルトでアクセス不可に設定されているため，許可フラグである URL を盗聴されたとしてもシステムの安全性は低下しない．具体的には，④認証要求の通信で登録した乱数を URL パラメータとして付加しなければ Web ページにアクセスすることはできないため，Web ページの URL は秘匿性を必要としない．

本実装における⑤認証許可の通信では，スマートフォン側は UDP での受信待ち状態から開始し，Raspberry Pi から Web ページの URL を受信する．その後，⑥通信開始の動作に移る．Raspberry Pi 側は，Web ページの URL を送信する．ただし，④認証要求の通信で受信した認証 PASS によって，Web ページ 1 と Web ページ 2（または Web ページ 3 と Web ページ 4）のどちらにアクセスしようとしたかを判定し，送信する URL を切り替える．その後，⑥通信開始の動作に移る．

⑥ 通信開始

本実装における⑥通信開始の動作には，OpenSSL による SSL 通信を用いた．また，Web ページはデフォルトではアクセス不可に設定されており，URL パラメータの乱数部分と乱数ファイルに登録された乱数を比較するプログラムを PHP で実装した．このプログラムのチェックに成功した場合は正規の Web ページを表示させる．また，URL パラメータの乱数部分がない場合や，乱数ファイル上の乱数との比較に失敗した場合はエラーページが表示される．

4.3 技術的課題

本実装にはいくつかの技術的課題がある．まず，提案手法では複数の操作端末が関わる環境を想定したものの，本実装では 1 台のスマートフォンのみの実装となった．複数台のスマートフォンが関係する環境では，IoT デバイス側で接続要求の受信から乱数の登録までを排他制御にする必要があり，Raspberry Pi 上のプログラムにこの機能を実装する必要がある．

次に，IoT デバイスには，乱数ファイルを適切に更新する仕組みが必要となる．本実装において，乱数ファイルを更新し古い乱数を消去するタイミングは，IoT デバイスが④認証要求の通信後に認証 PASS をチェックする処理で成功したタイミングとした．このため，例えばスマートフォン A が認証要求を送信してから専用通信を開始する前にスマートフォン B が認証要求を送信すると，スマートフォン A は登録した乱数を削除されてしまい，Web ページを閲覧できなくなるという現象が起きる．一方で，乱数ファイルを更新しなければ乱数の重複が起こり，個々のスマートフォンごとに乱数を設定する意味がなくなってしまう．したがって，複数台のスマートフォンが関係する環境では，乱数ファイルを更新するタイミングを一定時間ごとに区切り，例えば登録されてから 10 分以上が経過した乱数は削除するなどの機能が必要となる．この場合は，乱数だけでなくその乱数を登録した時間も乱数ファイルに記録しておかなければならない．もし IoT デバイスのマーカーとして QR コードではなく NFC リーダが利用できるならば，NFC リーダへの接触時刻を記録することも可能である．また，古い乱数を削除する機能を追加しても確率的に乱数が重複してしまうことが考えられるため，これに備えて再度乱数を生成して④認証要求の通信をやり直す仕組みも必要となる．また，乱数ファイル上において，登録された乱数が重複することを許す場合に，登録端末を識別することで Web ページへの不正アクセスを防ぐという手法を用いることもできる．例えば，乱数ファイルに認証要求を送信してきたスマートフォンの端末情報（IP アドレスや MAC アドレスなど）を記録しておくことで，Web ページにアクセスした際に，①物理的接触の動作を経ているスマートフォンの乱数かどうかを検出できる．

5. まとめ

本稿では，物理的な接触に基づいて操作端末からのアクセスの可否を判断する IoT デバイスのためのアクセス制御手法を提案した．また，本提案手法が盗聴・改変・なりすましに対して安全に通信可能であることも示した．さらに，提案手法のプロトタイプ実装を行うことによって，アクセス制御のあるシステムが実現可能であることを検証した．

4.3 節で述べたとおり，実装にはいくつかの課題が残るため，今後はこれらの課題を解決することが必要となる．また将来的には，端末の認証を超えて，物理的な接触を根拠としたユーザの認証が可能な認証手法についても検討したい．

Vol.2017-MBL-85 No.24
Vol.2017-ITS-71 No.24
2017/11/16

参考文献

[1] 濵野雅史, 新井イスマイル. 加速度センサ・ジャイロセンサを併用したスマートフォンの利用認証手法の提案. 研究報告ユビキタスコンピューティングシステム. 2014, vol. 2014-UBI-41, no. 17, p. 1-8.
[2] 泉将之, 佐村俊治, 西村治彦. フリック入力による日本語非定型文のキーストローク認証. 2013 年度 情報処理学会関西支部 支部大会 講演論文集. 2013, vol. 2013.
[3] 渡邊裕司. Android 端末上のタッチ操作に基づく個人認証に対する操作特徴の比較. コンピュータセキュリティシンポジウム 2014 論文集. 2014, vol. 2014, no. 2, p. 1015-1022.
[4] 見上一憲, 林原尚浩. タッチパネルと加速度センサを用いた携帯端末向けジェスチャ認証とその入力方式の提案. 研究報告コンピュータセキュリティ. 2012, vol. 2012-CSEC-56, no. 8, p. 1-7.
[5] 森井理智, 谷岡広樹, 大平健司, 佐野雅彦, 松浦健二, 関陽介, 上田哲史. パスワードレス認証方式を用いた認証連携に関する研究. 研究報告インターネットと運用技術. 2017, vol. 2017-IOT-37, no. 8, p. 1-6.
[6] 矢崎孝一, 伊藤栄信, 坂本拓也, 二村和明. スマホ認証を用いた IoT 機器サービスの簡易利用方式. 研究報告マルチメディア通信と分散処理. 2017, vol. 2017-DPS-170, no. 5, p. 1-8.
[7] 大谷誠, 江藤博文, 渡辺健次, 只木進一, 渡辺義明. シングルサインオンに対応したネットワーク利用者認証システムの開発. 情報処理学会論文誌. 2010, vol. 51, no. 3, p. 1031-1039.
[8] 末永光弘, 田中久治, 大谷誠, 堀良彰, 岡崎泰久, 渡辺健次. 次世代ネットワークに向けたネットワーク利用者認証システムの設計と実装. 情報処理学会論文誌. 2015, vol. 56, no. 9, p. 1782-1793.