Title	IoTデバイスのためのノードアイデンティティ管理システムの設計と実装
Sub Title	Design and implementation of node identity management systems for IoT devices
Author	山田, 陽平(Yamada, Yōhei) 砂原, 秀樹(Sunahara, Hideki)
Publisher	慶應義塾大学大学院メディアデザイン研究科
Publication year	2019
Jtitle	
JaLC DOI	
Abstract	
Notes	修士学位論文. 2019年度メディアデザイン学 第780号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40001001-00002019-0780

修士論文 2019 年度

IoT デバイスのためのノードアイデンティティ管理システムの設計と実装

慶應義塾大学
大学院メディアデザイン研究科

山田　陽平

本論文は慶應義塾大学大学院メディアデザイン研究科に
修士 (メディアデザイン学) 授与の要件として提出した修士論文である。

山田　陽平

研究指導コミッティ：

砂原 秀樹 教授　(主指導教員)
加藤 朗 教授　(副指導教員)

論文審査委員会：

砂原　秀樹 教授　(主査)
加藤　朗 教授　(副査)
古川　享 教授　(副査)

修士論文 2019 年度

IoT デバイスのためのノードアイデンティティ管理システムの設計と実装

カテゴリ：サイエンス / エンジニアリング

論文要旨

近年，ユビキタスコンピューティングに続く Internet of Things の概念の登場に伴い，“IoT デバイス” が増加の一途をたどっており，無数の IoT デバイスが日夜動作し続けている．しかし，IoT デバイスが普及が進んだ結果，管理されずに放置される IoT デバイスも増加している．管理不備の結果 IoT デバイスが乗っ取られた後に DDoS 攻撃に利用されるなど，ユーザは悪意のある攻撃者により被害を受けるだけでなく加害者となる事例が存在する．

そこで本研究では，IoT デバイスを管理するシステムの構築を目指し，管理システムへの初期登録の問題を解決するために IoT デバイスに向けたノードアイデンティティ管理システムを提案した．提案システムではワンタイムパスワードを用いた簡易相互認証および公開鍵の交換後にハッシュ値の確認を行う．提案システムにより悪意のある攻撃者によるなりすまし攻撃や中間者攻撃にさらされた場合でもユーザが所望するデバイスを管理システムへ登録し安全に公開鍵の交換が可能となる．設計および実装した提案システムに対して複数のデバイスにて動作確認を行い，Raspberry Pi Zero のような比較的計算資源が制限されたデバイスでの動作が確認された．さらに実装したシステムの脅威分析を行い，なりすまし攻撃や中間者攻撃に対して耐性があると評価した．

キーワード：

セキュリティ, IoT, デバイス管理, 相互認証, ワンタイムキー

慶應義塾大学大学院メディアデザイン研究科

山田　陽平

Abstract of Master's Thesis of Academic Year 2019

Design and Implementation of Node Identity Management Systems for IoT Devices

Category: Science / Engineering

Summary

Recently, with the emergence of the concept of the Internet of Things following ubiquitous computing, the number of "IoT devices" has been increasing and countless IoT devices continue to operate day and night. However, with the spread of IoT devices, the number of IoT devices that are left unmanaged is also increasing. As a result of poor management, this mean in DDoS attacks. That devices can be perpetrators to other.

In this paper, we propose a node identity management system for IoT devices to solve a problem in initial registration where the management system. In the proposed system, simple mutual authentication using a one-time password and confirmation by hash values of public keys.

This process can be proposal a device desired by a user even when exposed to impersonation attack or man-in-the-middle attack by a malicious attacker. The proposal system has been tested on multiple devices and tested on devices with relatively limited computational resources, such as the Raspberry Pi Zero. In addition, threat analysis to the implemented system was carried out, and it was evaluated to be resistant to impersonation attack and man-in-the-middle attack.

Keywords:

Security , Internet of Things, Device Management, Mutual Authentication, One-time Key

Keio University Graduate School of Media Design

Yohei Yamada

目　　次

図　目　次

表　目　次

第 1 章
序　　論

1.1. インターネットとデバイスの普及

インターネットが 1980 年代開発されてから 40 年近くが経過し，現在では人と人のコミュニケーションを超えて人とモノ，モノとモノを繋ぐインフラへと成長した．さらにムーアの法則 [1] に代表されるように集積回路は年々小型化，低価格化が進み，個人でも多数のコンピュータを持つことが一般化している．M. Weiser や坂村健らが提唱したユビキタスコンピューティング [2] [3] の時代が到来し，ありとあらゆるモノにコンピュータが搭載されつつある．ユビキタスコンピューティングに似た概念としてモノのインターネット (IoT: Internet of Things) が現れ，インターネットに繋がるモノ（デバイス）が更に増加している．

1.2. Internet of Things とデバイス

モノのインターネット (IoT: Internet of Things) という言葉が世に出たのは 1999 年 [4] と言われている．当時 IoT という言葉は，RFID(Radio Frequency Identifier) タグとインターネットを組み合わせるアイデアを説明する言葉として利用されていた．しかし現在では IoT の意味が大きく異なり，定義も文献により様々であるが，以下のようなものが挙げられる．

- 物に付けられた識別子やセンサーの情報を、 ネットワークを通じて相互に交換するという技術的な概念 [5]
- 従来インターネットに接続されていなかった様々なモノ（中略）が、ネット

ワークを通じてサーバーやクラウドサービスに接続され、相互に情報交換をする仕組み [6]

- 基本的にスイッチを備えたデバイスをインターネット接続するという概念 [7]

上記の定義に共通した事項は，“多数のデバイスがネットワークに接続されている”ことであり，実際に多くの IoT デバイスがインターネットへ常時接続している．

しかし近年，IoT デバイスの利便性が注目され活用が広がる中，管理の不備などが原因となりセキュリティにおける問題も多発している．特に 2016 年に発見された Mirai と呼ばれるマルウェアは，主に Linux で動作する IoT デバイスに対して出荷時に多く用いられるユーザーネームとパスワードを悪用し感染を広げ，多くの IoT デバイスを遠隔操作が可能なボットとした．世界中の Mirai に感染したデバイスが同時に特定のサーバにアクセスすることで，当時最大 1.1 Tbps に迫る DDoS(Distributed Denial of Service attack) 攻撃が発生することとなった．これらの原因として初期設定状態のまま管理されていない IoT デバイスが無数に存在していたことが挙げられる．IoT デバイスが急速に普及したが，管理されていないまたは存在が忘れられ放置されているデバイスが多数存在すると考えられる．

1.3. 研究目的

本研究は全ての IoT デバイスが管理される状態を目指し，IoT デバイスおよび管理デバイス向けの管理システムを提案する．また，管理システム上で発生する IoT デバイスと管理デバイスにおける初期登録の問題を解決するためにワンタイムキーを用いた簡易相互認証および鍵交換システムを提案を行い，実装および脅威分析を行う．IoT デバイスが同一ネットワーク上に多数存在し管理されていない状態が減ることで，IoT デバイスのセキュリティ強度が上昇し，Mirai のようなマルウェア対策になると考えられる．しかしながら，全ての悪意のある攻撃に対して対応することは不可能であるため，対応する脅威について限定した上で提案システムの実装を行う．

1.3.1 想定しない攻撃手段および攻撃元

本研究においては，高度な技術を持つ知能集団や国家情報機関等による大量の人的・計算資源的リソースを用いた攻撃は想定していない．更に，攻撃者によるデバイスやシステムに対する物理的な接触も想定しない．

1.4. 本論文の構成

本論文は全 6 章で構成される．本章では，研究背景や目的，想定しない攻撃手段および攻撃元について述べた．2 章では，IoT デバイスの概要や，IoT デバイスに関連するセキュリティの現状，業界団体により策定されたセキュリティ要件などについて述べる．3 章では，IoT デバイスにおけるセキュリティ的問題点をまとめた上で，IoT デバイスに向けたアイデンティティ管理システムの提案を示す．4 章では 3 章で述べた提案の実装に向けた要件述べた上で，実際に実装を行ったシステムについて記述する．5 章では，実装したシステムについて動作確認および動作時間，メモリ使用量の計測と評価と脅威分析の結果を記す．6 章では，本論文の結論を述べる．

第 2 章
IoTデバイスおよび認証方法の調査

2.1. IoT製品の増加

2020年1月現在，IoTデバイスと呼ばれるデバイスは多数存在しており，市場に投入されている製品数を把握することは困難である．これはIoTデバイスに対する一定の定義が特に定められていないことも一つの要因である．近年，IoTデバイスは増加の一途をたどっており，コンシューマ向け製品が多数発売されている．スマートスピーカーのようなこれまで製品として存在していなかったデバイスも存在するが，スマートウォッチやスマートロックなどの主にアナログで動いていたものをデジタルおよびネットワーク的に拡張したものも日夜利用されている．

2.1.1 スマートスピーカー

スマートスピーカーはAIアシスタントを搭載したデバイスで音声操作により各種操作が可能である．主要な製品としてGoogleが提供するGoogle Home[1] [8]（図2.1），Amazon.comが提供するAmazon Echo[2] [9]シリーズ（図2.2），Appleが提供するHomePodなどが知られる．機能としてはアラームやスケジュール確認などの基本的な機能のほか，音楽ストリーミングサービスとの連携を利用して音楽の再生やスマートロックなどに代表されるスマート家電の操作も可能である．

1 https://store.google.com/jp/product/google_home

2 https://www.amazon.co.jp/dp/B07PFFMQ64/

Google[1]から引用

図 2.1 Google Home

Amazon.com[2]から引用

図 2.2 Amazon Echo Dot

2.1.2 ネットワークカメラ

ネットワークカメラは主に防犯や子供，ペットなどの見守りを目的として設置されるカメラである（図 2.3[3]）．ネットワークカメラには映像を撮影する機能のほか，Web サーバ，メールクライアントなどの機能を持つものも存在し，メールによるアラーム機能やユーザによる独自のプログラム実行機能なども存在する．

2.1.3 その他の IoT デバイス

この他にも多数の IoT デバイスが存在する．既存の冷蔵庫，エアコン，洗濯機などに IoT 機能を搭載した IoT 家電や，玄関ドアの鍵に IoT の機能を外付けすることのできるスマートロック，温湿度や動きをセンシングするスマートタグなど，“IoT デバイス” と呼ばれるものには枚挙がない．

日本で購入できるデバイスを可視化した “IoT デバイスマップ 2018”[4] [10] がロボットスタート株式会社から公開されている．IoT デバイスマップを図 2.4 に示す．図 2.4 には 330 の IoT デバイスが掲載されており，利用用途および利用場所ごと

3 https://systemk-camera.jp/camera/basic/networkcamera/

4 https://robotstart.info/2018/04/17/iot-device-map-2018.html

SystemKCamera[3]から引用

図 2.3　ネットワークカメラの例

に分類がされている．掲載されている IoT デバイスは，Wi-Fi もしくは Bluetooth によってインターネット接続が可能なデバイス [10] が掲載されている．IoT デバイスの種類は無数に存在している．

2.2. IoT デバイスに関するセキュリティの現状

IoT デバイスの普及が加速度的に進んでいるが，セキュリティ面において問題が存在する．特に，IoT デバイスにおけるデフォルトユーザ名およびパスワードをそのまま利用するケースがある．例としてシングルボードコンピュータの Raspberry Pi では，ユーザが OS を選択することができる．このとき選択する Linux 系 OS である Raspbian では，初期ユーザ名およびパスワードが設定してされている．Windows や MacOS のような発売されている OS とは異なり，セットアップ画面等は基本的には存在しないため，ユーザ名およびパスワードの変更は利用者に委ねられる．ユーザ名およびパスワードが初期設定の状態の場合，telnet や ssh などを用いて端末へ接続が可能な状態の場合は悪意のある攻撃者により端末の制御が奪われる可能性がある．

これはシングルボードコンピュータに限った問題ではなく，IP カメラでも同様に問題となっている．Insecom [11] では，世界中のアクセス可能な IP カメラがデ

IoT デバイスマップ 2018[4]から引用

図 2.4　IoT デバイスマップ 2018

フォルトパスワードを使用している場合，リアルタイムに映像を閲覧することのできるサイトである．図 2.5 に閲覧例を示す．また図 2.6 のように IP アドレスからカメラが設置されている国名，都市名，郵便番号などが確認できる．

2.2.1 Mirai による IoT デバイスに関するセキュリティ被害

一般的に管理者が不在である IoT 機器 [12] は “野良 IoT デバイス” とも呼ばれ，一度設置された機器が使用されなくなり，放置される事 [12] などが問題となっている．特に IoT デバイスの管理不備による攻撃被害の例として，2016 年 8 月に特定された Mirai が形成したボットネット [13] による大規模 DDoS 攻撃が挙げられる．Mirai は最大 40 万のデバイスに感染し DDos 攻撃を行い，最大 1.1 Tbps のトラフィックを観測 [13] し，これらの攻撃により Twitter，Netflix，Github [13] などのサービスを停止させた．Mirai の特徴して IoT デバイスが初期設定で使われる ID・パスワード辞書を持つことが挙げられ，ID・パスワードが変更されていないデバイスに感染することでボットネットを形成した．

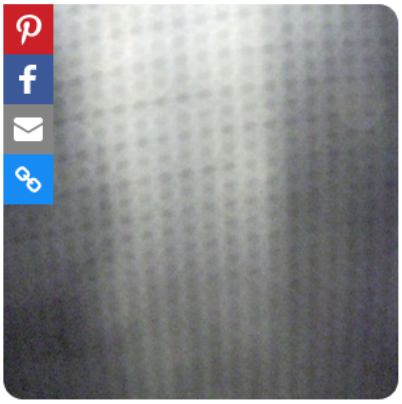
2.2.2 IoT デバイスにおけるセキュリティ要件および対策の取り組み

IoT デバイスに関連するセキュリティ要件を複数の機関が独自に策定，公開を行っている．中でも一般社団法人重要生活機器連携セキュリティ協議会 (CCDS: Connected Consunmer Device Security conuncil) では，IoT 分野共通セキュリティ要件ガイドライン [14] を公開している．ガイドラインではインターネットにつながる機器における最低限守るべき共通要件として表 2.1 に示すように 11 要件を定めている．表 2.1 に示した表は主ベンダおよび開発者向けの共通要件といえる．

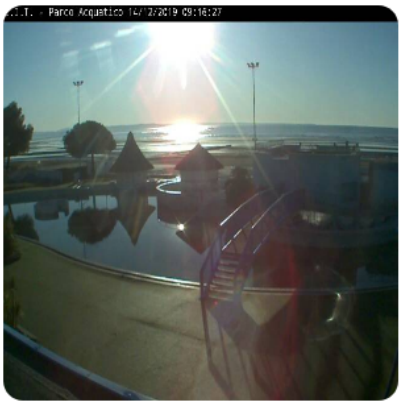
この他にも Open Web Application Security Project(OWASP) Internet of Things Project[5] [15] では，“IoT システムの構築，展開または管理すべき上位 10 項目” を不定期に更新している．OWASP は，ウェブアプリケーションセキュリティに関する課題解決を目的とする国際的オープンコミュニティの 1 つである．2018 年に

5 https://owasp.org/www-project-internet-of-things/

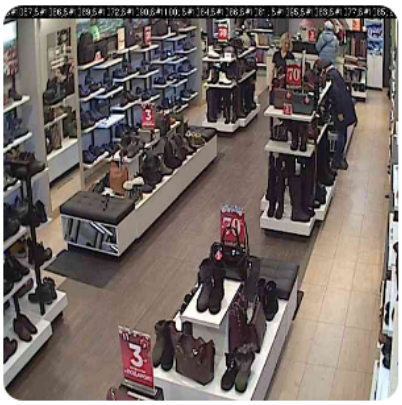
Watch Mobotix camera in Italy,Rome



Watch Panasonic camera in Japan,Fukuoka



Watch Axis camera in Italy,Padova



Watch Axis camera in Russian Federation,Nekrasovka

図 2.5　Insecom で確認できる監視カメラの例

Country:	United States
Country code:	US
Region:	New York
City:	New York City.
Latitude:	40.714270
Longitude:	-74.005970
ZIP:	10116
Timezone:	-05:00
Manufacturer:	Vivotek

図 2.6　Insecom が収集した監視カメラに関する情報の例

表 2.1　CCDS による IoT 分野での最低限守るべき共通要件

No.	要件	脆弱性の種類
1	SQL インジェクション対策	CWE-89
2	クロスサイトリクエストフォージェリ対策	CWE-352
3	パストラバーサル対策	CWE-22
4	未使用の TCP/UDP ポートを外部からの利用対策	CWE-671
5	TCP/UDP ポート適切なアクセス認証	CWE-287
6	認証情報の設定変更が可能 初めて利用する際に設定変更を促す ハードコーディングをしないこと	CWE-259
7	設定及び取得情報の消去	–
8	Wi-Fi アライアンス推奨の最新の認証方式を装備	CWE-326
9	Bluetooth SIG 推奨の最新のペアリング方式を装備	CWE-287
10	システム運用上不要なクラスを認識できない	–
11	ソフトウェア更新が可能かつ更新された状態が電源 OFF 後も維持	–

表 2.2 IoT システムの構築，展開または管理すべき上位 10 項目

順位	概要
1	脆弱，予測可能もしくはハードコードされたパスワード
2	安全でないネットワークサービス
3	安全でないエコシステムインタフェース
4	安全な更新メカニズムの欠如
5	安全でないもしくは古いコンポーネントの使用
6	不十分なプライバシー保護
7	安全でないデータの転送と保存
8	デバイス管理の欠如
9	安全でないデフォルト設定
10	物理的なハードニングの欠如

OWASP Internet of Things Project[5]より引用

示された OWASP Internet of Things Project による “IoT システムの構築，展開または管理すべき上位 10 項目” を表 2.2 に示す．

10 項目ではシステムの設計・構築および展開に関する項目が多く含まれるが，1，5，8 位の脆弱なパスワード，古いコンポーネントの利用，デバイス管理の欠如については，IoT デバイスの利用者に対する警告とも考えられる．

また，日本国内では NOTICE(National Operation Towards IoT Clean Environment) [16] と呼ばれる取り組みが進んでいる．NOTICE は総務省および国立研究開発法人通信研究機構（NICT: National Institute of Information and Communications Technology）およびインターネットプロバイダの協力により，サイバー攻撃に悪用されるおそれのある機器の調査及び注意喚起 [16] を行うなど，国内に存在する “野良 IoT デバイス” の撲滅に向けた取り組みを進めている．

2.3. 利用者認証

表 2.1 の No. 6 に示されているように，IoT 分野での共通要件の 1 つとして認証が挙げられる．IT 分野における認証は，アクセスなどを試みるユーザが正規なユーザであることを確認することで，本人確認とも呼ばれる．認証は各種情報機器やインターネット上に存在するサービスを利用する際に利用され，利用する権限のない第三者による意図しない利用を防ぐために利用される．

2.3.1 パスワード認証

パスワード認証は認証として最も一般的に利用されている認証方法である．一般的にユーザーネーム (または ID) と同時に用いられ，事前に設定したパスワードを知っていることで本人であることを認証を行う．パスワード認証はパスワードを知っていることで本人性を証明するが，ユーザがパスワードを他人に伝えてしまった場合や悪意のある攻撃者により窃取，盗聴，推測などにより本人以外でもアクセスする事が可能である．一般的に無線ルータなどのネットワークデバイスに対するパスワードは工場出荷時に統一またはランダムなパスワードが設定され，デバイスへのシール貼付などの方法でユーザに共有される．特に統一なパスワードが利用されている場合は，デバイスの利用者により初期設定時に変更することが強く求められている [17].

ただし，パスワード認証には総当たり攻撃・辞書攻撃・類推攻撃などの代表的な攻撃方法が知られている．中でも総当たり攻撃は最も単純な攻撃方法であり，考えられる全てのパスワードを試すものである．無限の時間をかけることで総当たり攻撃は必ず成功するが，パスワードの長さや計算資源などにより試行回数が大きく左右されるため，数時間や数日など現実的に可能な時間内で攻撃が成功する環境は限られる．

2.3.2 公開鍵認証

公開鍵認証は公開鍵暗号と呼ばれる事前に作成した公開鍵・秘密鍵の 2 つの鍵を用いた認証方法である．公開鍵・秘密鍵には “公開鍵で暗号化したものは秘密鍵でしか復号できない” という特性があり，これを利用した認証方法である．公開鍵認証では秘密鍵が窃取されなければ認証を突破することは困難である．公開鍵暗号および公開鍵認証で最も一般的に利用される RSA 暗号の安全性評価[6] [18] によれば，鍵長 2048 bit 以上であれば 2035 年までは計算量的安全性が担保されると評価されている（図 2.7 参照）．公開鍵認証方式ではユーザ（クライアント）が公開鍵を事前に交換し利用するサービスやサーバに登録しておく必要がある．

2.3.3 生体認証

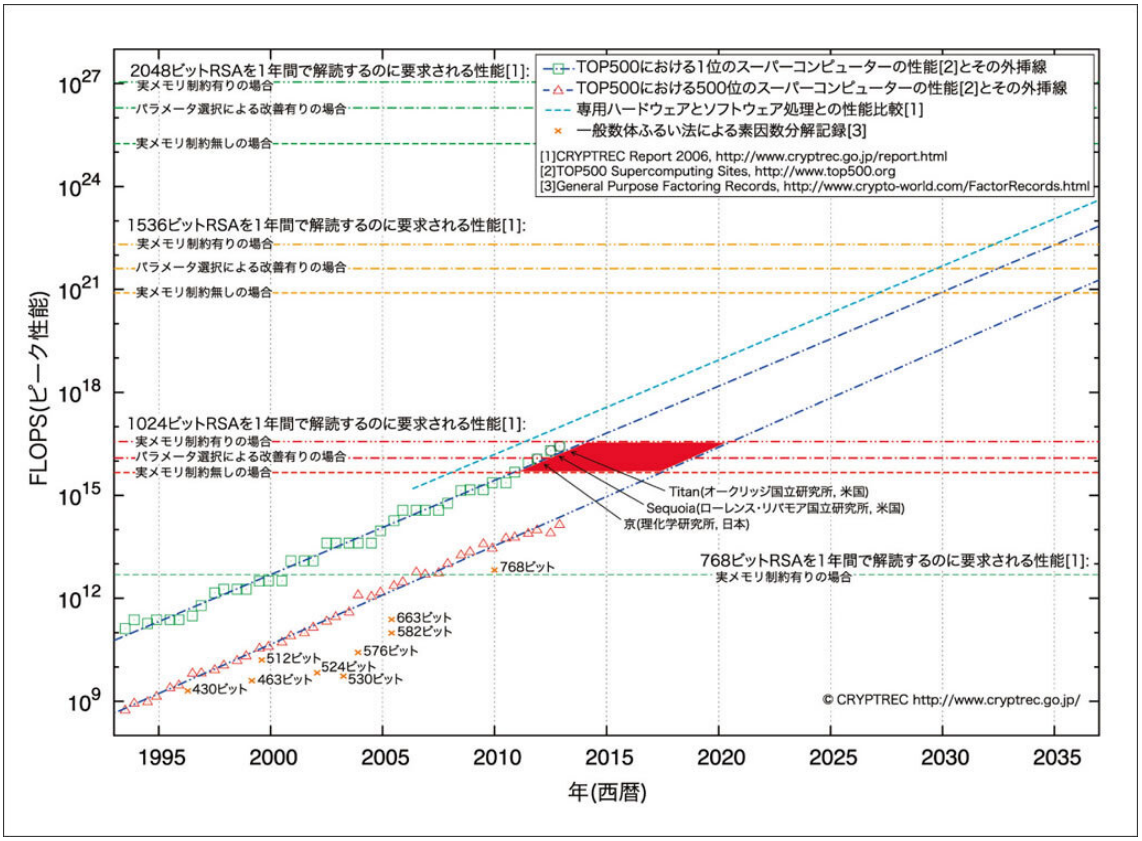
生体認証は人間の指紋・虹彩・顔型などをセンサにより取得し，事前に取得したデータと比較することで認証を行う方式である．近年ではスマートフォンで指紋や顔認証システムが導入されつつある．ただし，必ずしも安全であるとは言い切れず認証に失敗するケースや偽造された指紋により認証を突破されるケース [19] が存在する．さらに残留指紋や写真で撮影された虹彩をもとに認証を突破したという報告 [20] もある．

2.4. 公開鍵暗号

公開鍵暗号とは公開鍵と秘密鍵と呼ばれる対となる鍵を利用する暗号方式である．公開鍵暗号方式特徴とし公開鍵で暗号化した情報は秘密鍵のみでしか復号できないという数学的特性を持つ．

公開鍵暗号を用いたメッセージの交換では予め一対の公開鍵暗号を用意しておき，公開鍵をメッセージを交換したい相手に送信する．A と B における通信を考える．A は予め一対の公開鍵暗号を用意し，公開鍵を B へ送信しておく．通信時

6 https://www.nict.go.jp/publication/NICT-News/1303/01.html



国立研究開発法人情報通信研究機構[6]より引用

図 2.7 RSA 暗号の安全性評価

に B が送信する際は公開鍵を使いメッセージを暗号化し，秘密鍵をもつ A に送信する．B から送信されたメッセージは公開鍵で暗号化されているため，A は自分が持つ秘密鍵によって B からのメッセージを復号することができる．

また，秘密鍵を用いて電子証明を生成しメッセージに付加することで，A からメッセージを送信されたというデジタル署名としても公開鍵暗号を利用することができる．

2.4.1 公開鍵認証基盤

前述の公開鍵暗号では A から B へ公開鍵が事前に送信されているが，悪意のある第三者によりなりすましや，中間者攻撃により不正な公開鍵が送信されている可能性があり，これを直ちに確認することはできない．このなりすましに対する対策として考案されたのが公開鍵認証基盤 (PKI: Public Key Infrastructure) である．PKI には PGP(Pretty Good Privacy) モデルと認証局モデルが存在するが，ここでは認証局モデルについて述べる．図 2.8 に公開鍵認証基盤における認証の例を示す．認証局モデルは公開鍵の所有者を信頼できる第三者機関によって保証するという方式である．この信頼できる第三者機関は認証局 (CA: Certification Authority) とも呼ばれる．CA は何らかの方法で鍵を生成した A が本人であることを確認し，A の公開鍵に対する証明書を発行する．証明書は A の情報，A の公開鍵，CA のデジタル署名がセットになったものである．CA のデジタル署名は CA が発行した CA 秘密鍵にて署名される．この証明書は所有者だけでなく，CA によりリポジトリにも登録され公開される．証明書が何らかの理由で失効した場合もリポジトリへ失効登録がされ失効状態であることが公開される．

図 2.9 に証明書の例を示す．図 2.9 では www.kmd.keio.ac.jp のサーバが持つ公開鍵に対し，DegiCert.inc という CA によって認証がなされている．証明書を確認すると www.kmd.keio.ac.jp の証明書を GeoTrust RSA CA 2018 によりデジタル署名されていることがわかる．更に GeoTrust RSA CA 2018 を確認すると DigiCert Global Root CA によりデジタル署名されていることがわかる．このような署名のツリー構造を階層型モデルと呼び，階層型モデルの頂点に存在する証明書をルート証明書と呼ぶ．階層型モデルは信頼の連鎖により保証されている．デジタル署

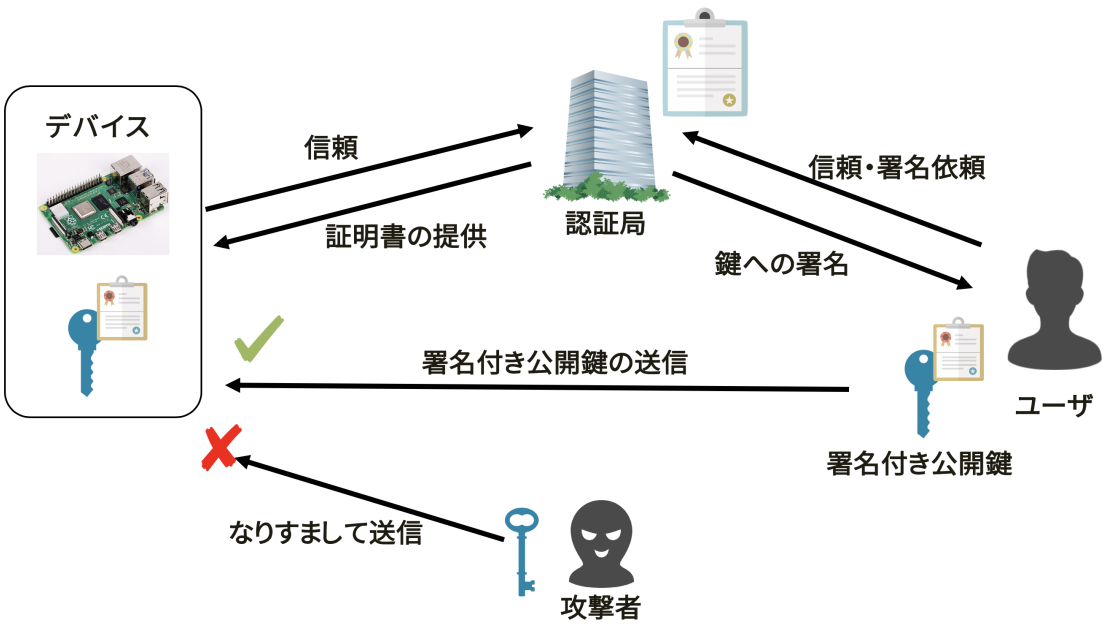


図 2.8 公開鍵認証基盤における認証の例

ウェブサイトの識別情報

ウェブサイト: www.kmd.keio.ac.jp

運営者: 検証され信頼できる運営者情報はありません

認証局: DigiCert Inc　　証明書を表示...

有効期限: 2020年5月8日

プライバシーと履歴

昨日までにこのサイトを表示したことがあるか　はい、66 回

このサイトはコンピューターに情報を保存しているか　はい、Cookie のみ　Cookie とサイトデータを消去

このサイトのパスワードを保存しているか　いいえ　パスワードを表示...

技術情報

接続が暗号化されています (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384、鍵長 256 bit、TLS 1.2)

表示中のページはインターネット上に送信される前に暗号化されています。

暗号化によってコンピューター間の通信の傍受は困難になり、このページをネットワークで転送中に誰かにその内容をのぞき見られる可能性は低くなります。

図 2.9 第三者機関によって保証された証明書の例

名が本当に正しいものであるかの確認のために，利用者は事前に CA 証明書を入手する必要がある．メーラーやブラウザには開発者により信頼された CA 証明書がソフトウェアと共にインストールされ証明書の確認に用いられる．

2.4.2 CA 証明書のプリインストール

前述のように PKI では第三者認証機関である CA により公開鍵に対するデジタル署名が行われている．PKI ではツリー状の信頼の連鎖によって署名の保証や確認を行う．このとき，署名の確認のためにデバイスには CA 証明書がインストールされている必要がある．さらに CA 証明書は常に信頼できるとは限らず，失効や新たに登録される場合も存在する．

デバイスベンダーにより多くのデバイスには工場出荷状態であっても PKI が使えるよう，事前に CA 証明書を製造時にプリインストールされていることが多い．ただし，プリインストールする証明書はベンダーが選択するもので必ずしも信用できるものであるとは限らない．また，ユーザが工場出荷状態にデバイスを戻した場合，失効していた証明書がデバイス内で有効なものとして認識される可能性がある．

第 3 章
提　　案

3.1. IoT デバイスにおけるセキュリティの問題点

前章までで述べたように，IoT デバイスでのセキュリティにおける問題として，デバイス管理の欠如や放置により野放しであることが挙げられる．デバイスの存在や利用者，管理の責任者などが不明であるケースも存在しており，このような IoT デバイスが外部ネットワークからアクセスが可能な場合は，情報の窃取やデバイスの不正アクセス等のリスクが存在する．また一般のパーソナルコンピュータやスマートフォンの持つリソースと比較して，IoT デバイスが持つリソースは限られるものが多い．表 3.1 に主要 IoT デバイスの CPU および RAM 性能の一覧を示す．製品における CPU および RAM 公称値がない Google Home および Amazon Dash Button は分解解析[12]による搭載可能なプロセッサの性能を記載する．

表 3.1　主要 IoT デバイスの CPU および RAM 性能一覧

名称	CPU コア数	CPU 周波数	RAM
Google Home	2	1.2 GHz[1]	512 MB[1]
Amazon Dash Button	1	120 MHz[2]	32MB[2]
Raspberry Pi Zero W [21]	1	1 GHz	512 MB
Raspberry Pi 4B [22]	4	1.5 GHz	1,2,4 GB
ESP32 [23]	1	160–240 MHz	520+12 KiB
Rock Pi 4 [24]	6	2 GHz × 2 + 1.5 GHz × 4	1,2,4 GB

1 https://tomkanok.wordpress.com/2013/07/25/chromecast-device-cpu-detailed-specifications/

2 https://mpetroff.net/2016/07/new-amazon-dash-button-teardown-jk29lp

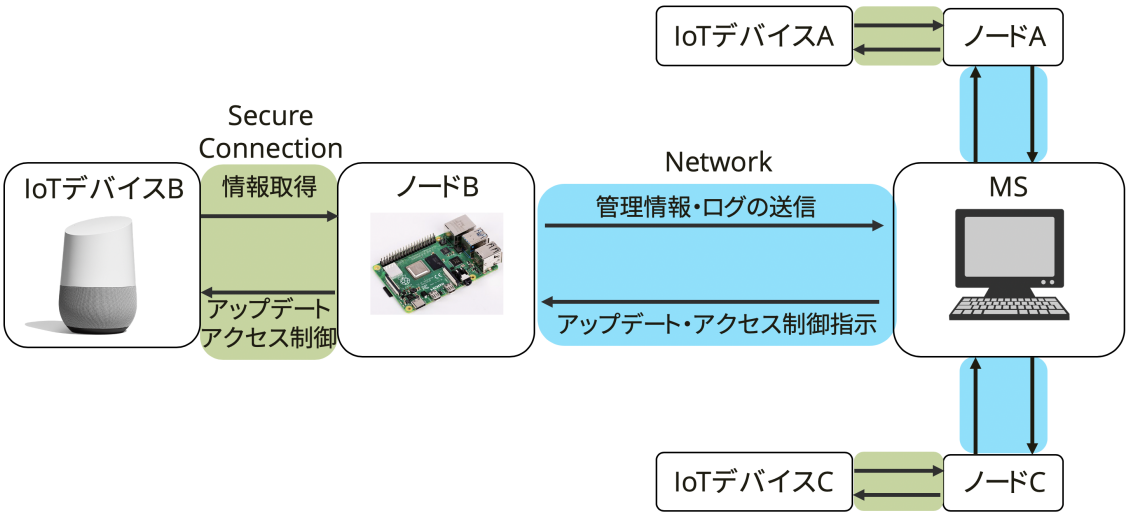


図 3.1 IoT デバイス管理システム

また，IoT デバイスで利用される IoT OS の数は膨大 [25] であり，独自バージョンの OS が利用されていることも多く，利用できるコマンドにもばらつきが存在し統一されていない．

3.2. 提案手法

本研究では，1 章で述べたように IoT デバイス管理システムと管理システムにおける初期登録における鍵交換の問題を解決するために簡易相互認証および鍵交換システムを提案する．本節では，まず IoT デバイス管理システム全体について述べた上で，簡易相互認証および鍵交換システムについて提案を行う．

3.2.1 IoT デバイス管理システム

IoT デバイスのデバイス管理を行うシステムの全体図を図 3.3 に示す．

提案する管理システムでは，管理デバイスであるマネジメントステーション (Management Station) と被管理デバイスである IoT デバイスとの間に IoT ノード（以下，ノード）を設置する．IoT ノードを設置することで，IoT デバイスにおける OS 差の吸収を図る．マネジメントステーションは複数の IoT デバイスおよびノードを管理を行い，各ノードとはネットワークにより接続されている．IoT デバイスとノードは一対一の対応関係であり，IoT デバイスの入出力は IoT ノードに接続される．この 2 つの間における接続は悪意のある攻撃者による攻撃が行われない安全な接続であると仮定する．マネジメントステーションは各ノードに対して IoT デバイスおよびノードに対してアクセス制御やアップデートに関する指示を行う．またノードから発せられたアラートやログ情報をデバイスを管理するユーザに情報提示を行う．ノードは IoT デバイスの入出力のアクセス制御を行い，IoT デバイスのネットワーク入出力や接続に関するログの収集を行い，必要であればマネジメントステーションへアラートを発する．

3.2.2 デバイス管理の定義

デバイス管理とはユーザが利用する全てのデバイスが目視による物理的な確認とネットワーク的存在の確認をすることが確認できることである．さらに，全てのデバイスの温度を始めとする物理的状態や CPU，メモリ等各種計算資源の利用状態などが確認できることとする．

3.2.3 IoT デバイス管理システムに利用する認証

2 章で IT 分野において用いられる主な認証方式としてパスワード認証，公開鍵認証，生体認証について述べた．パスワード認証は最も一般的に用いられる認証方式であるが，ユーザが設定するパスワードが脆弱であったり，推測可能である場合は容易に突破が可能であるという問題点が存在する．次に生体認証は指紋や虹彩などを用いた認証方式であるが，精度については疑問が残るため，現在は一般的に銀行 ATM などで二要素認証の 1 つとして利用されることが多い．

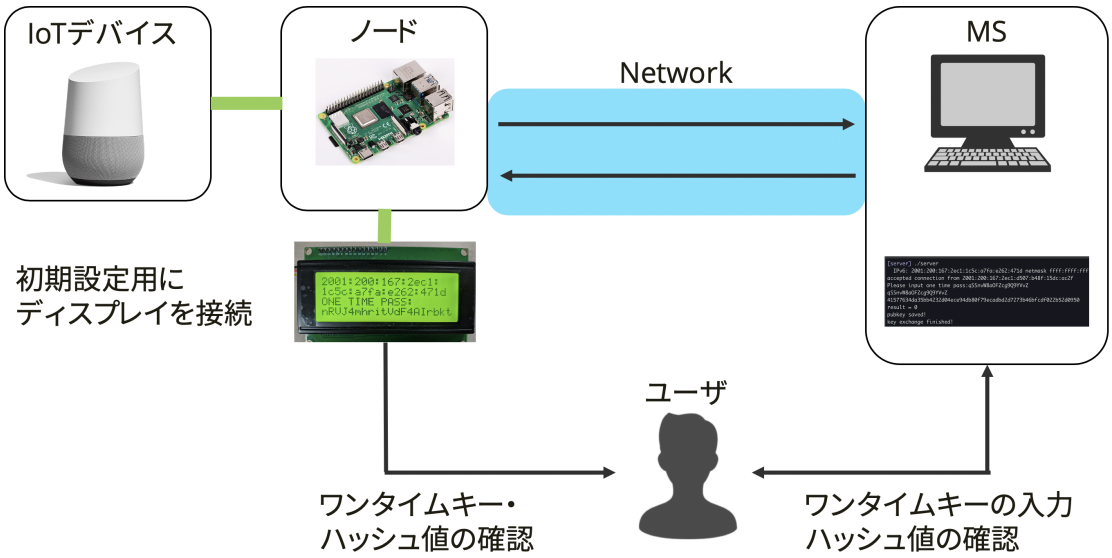


図 3.2 ワンタイムキーを用いた相互認証システム

そこで IoT デバイス管理システムの認証方式として公開鍵認証を用いる．公開鍵認証方式には前述の通り，相互認証方式と第三者認証方式が考えられるが，第三者認証の利用には CA 証明書のプリインストールやユーザが公開鍵を CA に対して署名を依頼する必要があるなど，時間的およびコスト的ハードルが存在する．しかし，相互認証では事前に安全な鍵交換を必要とし，悪意のある攻撃者によるなりすまし攻撃への対策が必要である．そこで本研究では，相互認証における初期登録時の安全な鍵交換を実現するために，第三者認証などを利用しない相互認証を行う手法として，ワンタイムキーを用いた簡易相互認証システムを提案する．

．

3.2.4 IoT デバイス管理システムにおける簡易相互認証システム

ワンタイムキーによる簡易相互認証システム図を図 3.2 に示す．

提案するワンタイムキーによる相互認証システムでは，管理システムへのノードの初期登録時において乱数を元に生成したワンタイムキーとワンタイムキーをハッシュ関数によりハッシュ化を行い，ハッシュ値をネットワーク経由で，ワンタ

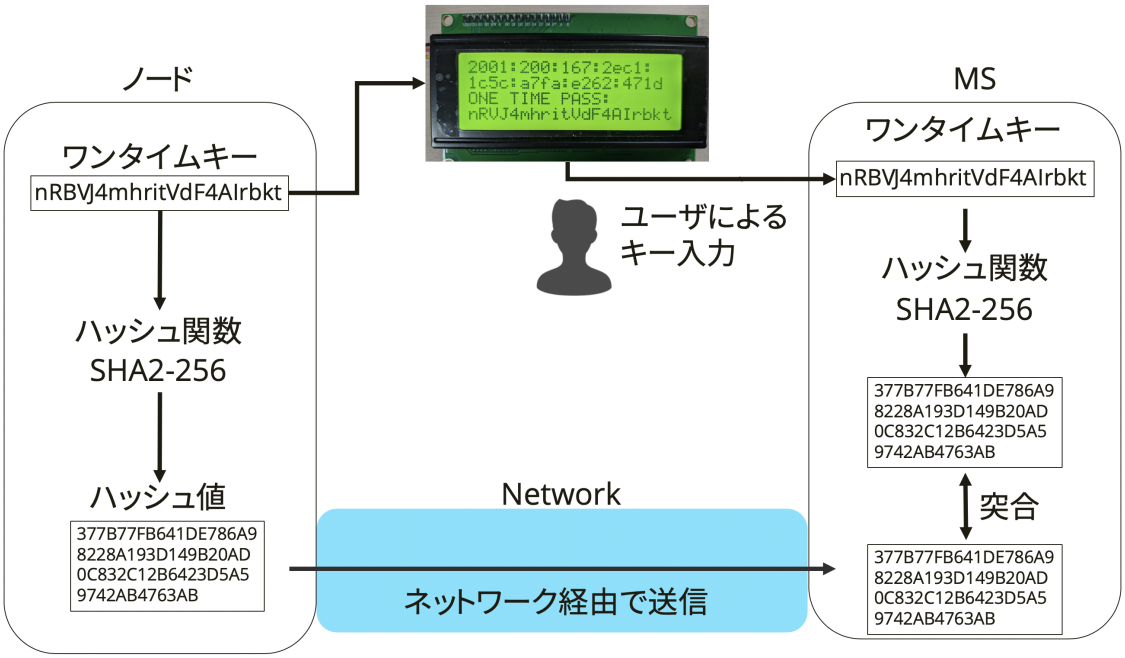


図 3.3 ワンタイムキーとハッシュ値による相互認証

イムキーをディスプレイを使った物理空間で送信を行う．ユーザはノードが表示したワンタイムキーをマネジメントステーションへ入力を行う．マネジメントステーションではユーザが入力したワンタイムキーを同じハッシュ関数によりハッシュ化を行い，ネットワーク経由でノードから受信したハッシュ値と突合を行うことで鍵交換を行う相手がユーザが登録を所望しているノードであるかを確認を行うことで相互認証を実現する．デバイス A および B 間における相互認証を行う際，事前にデバイス A がワンタイムキーを生成し，ハッシュ値 H_A をのみをデバイス B に対して送信する．ユーザの手によりデバイス B にワンタイムキーが入力され，デバイス B では直ちに入力された情報に対するハッシュ値 H_B の計算が行われる．H_A と H_B が一致した場合はワンタイムキーを生成したデバイス A とデバイス B は相互を正しく認証したといえる．

本論文では，IoT デバイスの導入段階における管理デバイスと被管理デバイスである IoT デバイスを，正しく認識した上で紐付けを行い，鍵交換を半自動的に

行うシステムを提案する．

3.3. 仮定環境

3.3.1 利用者条件

本システムを利用するユーザについて定義を行う．IoT ノードを登録するためにシステムを利用するユーザおよび利用するネットワーク管理者は悪意のある攻撃者ではない．また，ユーザは IP アドレスなどのネットワークに接続する知識は理解しているものとし，マネジメントステーションの IP アドレスを事前に把握しているものとする．

3.3.2 ネットワーク条件

提案システムの利用はローカルエリアネットワーク (LAN: Local Area Network) 内での利用を想定する．IP アドレスについては IPv6 がネットワーク管理者により LAN へ提供されており，各端末への割当は Dynamic Host Configuration Protocol(DHCP) や静的割当などにより適切な IP アドレスの割当が事前にされていることを前提とする．

マネジメントステーションの IP アドレスは IoT ノード（またはユーザ）では既知の情報とし，各デバイスを LAN に接続するなどの作業については，事前に完了しているものとする．

3.3.3 物理的条件

提案するシステムにおいて登録を行う際の物理的条件を述べる．

登録を行う IoT デバイスおよび IoT ノードとマネジメントステーションはユーザの手元に存在しており，ノードとマネジメントステーションが遠隔地にある状態での初期設定を行うことは想定しない．また，攻撃者によりマネジメントステーションや IoT ノード，IoT デバイスに対してすり替えや改造，破壊などの物理的

な干渉は行われないこととする．また，ユーザが設定を行なう部屋や環境に対して攻撃者による監視カメラや音声を盗聴されることについても想定しない．

3.3.4 攻撃者

システムに対する攻撃者は様々な攻撃者が想定され，攻撃者の所属する組織によっては攻撃規模や攻撃に用いるシステムの演算能力は異なると考えられる，想定する攻撃内容は，なりすまし (Spoofing) 攻撃や中間者攻撃 (MTM:Man in The Middle) を想定する．

また本論文では，国家によるサイバー攻撃や量子コンピュータを利用する攻撃などは想定しない．また攻撃者による物理的な破壊行為や干渉および管理および非管理デバイスが攻撃者により事前に入手され，回路の改造，プログラムコードの改ざんなどを行う攻撃についても想定しない．

3.4. 設計要件

提案および想定環境から得られたシステムの設計要件について述べる．システムは IoT ノードとマネジメントステーションからなる．マネジメントステーションは IoT ノードを管理する役割を持ち，一台のマネジメントステーションに対して単一ないし複数の IoT ノードと関係を構築する．システムの設計要件を以下に示す．

- マネジメントステーションと IoT ノードが互いに正しい通信相手であることを確認できること
- 鍵交換を自動的に行い送信（受信）した鍵に誤りがないことを確認できること
- 鍵交換などの知識がないユーザであってもシステムが利用できること

以上を踏まえた保護機構としてシステム全体を図 3.3 に示す．

ネットワークはLANを前提とする．またインターネット・プロトコルバージョン4（IPv4: Internet Protocl version 4）は割当が終了 [26] し各種団体がIPv6の利用推進が行われており，今後利用に適さない可能性がある．このため，提案するシステム全体としてIPv6のみに対応することとする．

3.4.1 ハードウェア設計要件

既存のIoTデバイスではその目的，価格帯によりデバイスの演算資源，ディスプレイの有無は異なり，汎用入出力 (GPIO: General-Purpose Input/Output) が存在しない場合もある．そのため，鍵生成や情報表示，入出力制御を行うデバイスが必要である．本論文ではこの鍵生成や情報表示，入出力制御を行うデバイスをIoTノードと呼ぶ．IoTノードにはキャラクタ表示機として小型の液晶ディスプレイ (LCD:Liquid Crystal Display) を接続し必要な情報をユーザに提供する．マネジメントステーションは複数のIoTノードを管理するデバイスである．マネジメントステーションは，登録要求をしてきたIoTノードのIPアドレスを確認できるようユーザが理解できるよう表示し，またワンタイムキーを入力できるようキー入力が行う事ができる必要がある．このようにハードウェアとしてPCなどのマネジメントステーションと鍵生成や情報表示，入出力制御を行うIoTノード2つで構成する設計とする．

第 4 章
設計および実装

4.1. IoT ノードの設計

IoT ノードではユーザがマネジメントステーションと IoT ノード間のセットアップを行う．IoT ノードには以下の機能の実装を行う．

- 公開鍵・秘密鍵の自動生成
- ワンタイムキーの生成および LCD によるユーザへの提示
- ワンタイムキーのハッシュ値の計算およびマネジメントステーションへの送信
- 公開鍵の自動生成およびマネジメントステーションとの鍵交換
- ハッシュを用いた公開鍵の正当性チェック

4.2. マネジメントステーションの設計

マネジメントステーションは IoT ノードから来た登録要求に対して登録を行い鍵交換をするか拒否するかを判定する．登録を行う場合，IoT ノードの IP アドレスと共に公開鍵を保存する．マネジメントステーションは以下の機能を実装する．

- IoT ノードからの登録要求待機
- 入力されたワンタイムキーのハッシュ値を計算し受信したものと突合するかのチェック

- IoT ノードの登録および公開鍵の保存
- ハッシュを用いた公開鍵の正当性チェック

4.3. 実装

システムにおけるマネジメントステーションと IoT ノードの鍵交換プロセスは以下の通りである．

1. ユーザはマネジメントステーションでサーバプログラムを起動する．起動時にマネジメントステーションの IP アドレスが表示され，待機状態になる．
2. ユーザはマネジメントステーションに登録を行いたい IoT ノードでクライアントプログラムを起動する．
3. IoT ノードで自動的に公開鍵の生成が行われた後に，ワンタイムキーの生成およびワンタイムキーの SHA2-256 によるハッシュ値の計算を行う．
4. IoT ノードからマネジメントステーションへ鍵交換要求を送る．
5. IoT ノードからの要求を受けたマネジメントステーションは自身の IP アドレスを送信する．これから先マネジメントステーションは，現在登録を要求している IoT ノードの登録が終了または失敗するまで他の IoT ノードからの登録受付を一時停止する．
6. IoT ノードはマネジメントステーションから受け取った IP アドレス情報およびワンタイムキーを LCD に表示する．マネジメントステーションへワンタイムキーのハッシュ値を送信する．
7. マネジメントステーションはハッシュ値を受け取った後，ワンタイムキーの入力を待機する．
8. ユーザは LCD に表示されたワンタイムキーをマネジメントステーションに入力する．

9. マネジメントステーションは入力されたワンタイムキーのハッシュ値を計算し突合を行う．一致していた場合はIoTノードへ鍵交換の許可し，一致しない場合はその時点で通信を終了する．

10. IoTノードは鍵交換の許可を受け取り，2.で生成した公開鍵のハッシュ値を計算した上でマネジメントステーションへ送信し，ハッシュ値をディスプレイへ表示する．

11. マネジメントステーションは公開鍵の受け取り，ハッシュ値を計算する．このときユーザにハッシュ値が一致するかの確認を求める．ユーザによる確認でハッシュ値が一致した場合，マネジメントステーションはIoTノードの接続情報を用いて送信された鍵を保存する．

12. マネジメントステーションの公開鍵をIoTノードへ送信し，公開鍵のハッシュ値をディスプレイに表示する．

13. 11.と同様にユーザはハッシュ値の確認を要求する．

14. ユーザの確認によりハッシュ値が一致すればIoTノードはマネジメントステーションの鍵を保存し，接続を切る．

以上が認証を行い，鍵交換を完了するまでのプロセスである．ユーザは6.のタイミングでマネジメントステーションに表示されているIPアドレスと，LCDに表示されているIPアドレスが一致していることを確認しマネジメントステーションへワンタイムキーを入力する．鍵交換時にはマネジメントステーションとIoTノードが持つそれぞれのハッシュ値を確認することで，鍵交換および登録は終了する．

IoTノードおよびマネジメントステーションの実装における共通要件を表4.1に示す．

表 4.1 簡易相互認証システムの実装における共通要件

OpenSSL Version	1.1.1d
ハッシュアルゴリズム	SHA 2-256
公開鍵の種類	RSA
生成公開鍵鍵長	4096 bit

表 4.2 Rock Pi 4B 諸元

OS	Ubuntu 18.04.3 LTS
Processor	Rockchip RK3399
CPU	2 GHz × 2 + 1.5 GHz × 4
RAM	LPDDR4 3200 Mb/s 4 GB
Display	HDMI 2.0
Audio	3.5mm jack with mic
Network Interface	Gigabit Ether LAN, Wifi 802.11 ac
I/O	UART SPI I2C PCM/I2S PWM ADC GPIO
Size	85 mm × 54 mm
Others	USB3.0 OTG, Bluetooth 5.0

諸元は Rock Pi 公式ホームページ[1]より引用

4.3.1 IoT ノード

IoT ノードの本体には Radxa 社製シングルボードコンピュータ Rock Pi 4B[1]（図 4.1）を用いた．Rock Pi 4B の性能諸元を表 4.2 に示す．Rock Pi 4B は公式サポート OS として Android7,9，Ubuntu18.04，Debian9 などが選択できるが，IoT ノードの実装において I2C などの利用が確認されている Ubuntu18.04 を用いた．ユーザへの初期登録時に IP アドレスやハッシュ値の情報を提供するモジュールとして LCD キャラクタディスプレイモジュール (LCD2004A) を用いた．また LCD キャラクタディスプレイモジュールと Rock Pi 4B 間の変換にはアイ・スクエアド・シー (I^2C) 変換モジュール (FC-113) を用い結線の簡略化を図る．

IoT ノードでは以下の機能を実装した．

1 https://rockpi.org/

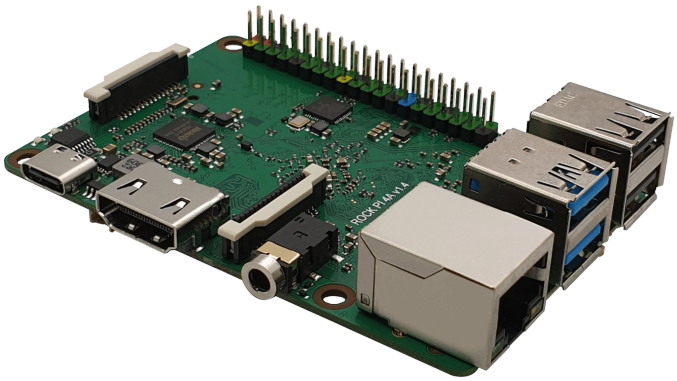


図 4.1 Rock Pi 4B ver1.4

- シェルスクリプトによる公開鍵の自動生成
- ワンタイムキーの自動生成
- ワンタイムキーおよび公開鍵のハッシュ化
- マネジメントステーションへの登録および鍵交換要求
- LCD を用いた IP アドレスおよびハッシュ値の表示

実装は明記がない場合，C 言語を利用し，コンパイラは gcc version 7.4.0 (Ubuntu/Linaro 7.4.0-1ubuntu1 18.04.1) を用いた．

4.3.2 マネジメントステーション

マネジメントステーションでは以下の機能を実装した．

- IoT ノード登録受付および鍵交換
- ワンタイムキーおよび公開鍵のハッシュ化

表 4.3 マネジメントステーション諸元

OS	macOS 10.15.2（19C57）
Processor	6-Core Intel Core i7
CPU	2.6 GHz × 6
RAM	LPDDR4 2400 MHz 16 GB
Network Interface	Wifi 802.11 ac

今回の実装では，マネジメントステーションに Macbook Pro(15-inch,2018) を用いた．マネジメントステーションの諸元を表 4.3 に示す．実装は C 言語にて行い，コンパイラは Apple clang version 11.0.0 (clang-1100.0.33.8) を用いた．

第 5 章
評 価・考 察

5.1. システム動作確認

実装したシステムの動作結果を示す．図 5.1 はマネジメントステーションの実行結果である．図 5.2，図 5.3 および図 5.4 に IoT ノードに接続した LCD の表示を示す． IoT ノードでは，プログラム起動後に公開鍵の生成，ワンタイムキーの生成及びハッシュ値の計算を行い，マネジメントステーションへ登録要求を送信する．マネジメントステーションからの受付確認後，予め計算したワンタイムキーのハッシュ値をマネジメントステーションへ送信する．また同時に IoT ノードは LCD へマネジメントステーションの IP アドレスおよびワンタイムキーを表示する．マネジメントステーションへユーザが正しくワンタイムキーを入力した後は，マネジメントステーションとの鍵交換を実施し，ユーザによるハッシュ値の確認を経て，マネジメントステーションの公開鍵を保存する．

マネジメントステーションではプログラムを起動直後に自身の IP アドレス表示し，IoT ノードの登録要求を待機する．その後 IoT ノードからの接続要求およびワンタイムキーのハッシュ値を受信し，ユーザによるワンタイムキーの入力を待機する．このときのワンタイムキー “DKWbgnGWnow0t5dXiGMP” が正しく入力され，ハッシュ値が一致したため，IoT ノードとの鍵交換を行い，ユーザによるハッシュ値の確認を経て，保存された．

```
[server] ./server
  IPv6: 2001:200:167:2ec1:1c5c:a7fa:e262:471d netmask ffff:ffff:ffff:ffff::
accepted connection from 2001:200:167:2ec1:799a:904b:1e63:a544
Please input one time pass:DkWbgnGWnow0t5dXiGMP
DkWbgnGWnow0t5dXiGMP
5e814b875acb110b9e5a41c232092ecdaec3541f17d56ff247df04e160ca198c
result = 0
hash = 8ec4cf2a776dcf14b09574f8a0eb8a79c8b30ca2fb47365d2c6a1d63e8763b1e
Conform? [y/N]:y
pubkey saved!
public key send
hash = 62ccd70a96c96c7705b6bd41cd8a04ac954c2de4867bd6ddb8c529ab5e936426
Conform? [y/N]:y
key exchange finished!
```

図 5.1 マネジメントステーションにおける server プログラムの実行結果

表 5.1 動作評価諸元

名称	CPU	RAM	OS
Raspberry Pi Zero W [21]	1 GHz	512 MB	Raspbian 4.19
Raspberry Pi 3B [22]	1.5 GHz	1 GB	Raspbian 4.19
Rock Pi 4	2 GHz × 2 + 1.5 GHz × 4	4 GB	Ubuntu18.04

5.2. ソフトウェアの動作評価

実装を行った簡易相互認証システムの IoT ノードを複数のシングルボードコンピュータにて実行し，各シングルボードコンピュータにおける実行時間，メモリ使用量を取得を行う．実験に利用したシングルボードコンピュータを表 5.1 に示す．

実験は各シングルボードコンピュータにおいて 10 回の鍵交換を行い，time コマンドを用いてプロセスの実行時間，メモリ使用量の測定を行なう．プロセス実行時間の計測は，単純にプロセスの開始から終了までを計測した場合，ユーザが行うワンタイムキーの入力や鍵交換プロセスにおけるハッシュ値の確認時間を含めてしまい，プロセス自身の実行時間を計測することができない．このため，今回の計測評価ではプロセスがユーザモードで CPU を使った時間である，user time を用いて計測を行う．

図 5.2 IoT ノードにおける LCD の表示結果 1

図 5.3 IoT ノードにおける LCD の表示結果 2

図 5.4 IoT ノードにおける LCD の表示結果 3

5.2.1 計測結果・考察

各シングルボードコンピュータのメモリ使用量の計測結果を図5.5に，ユーザ時間の計測結果を図5.6に示す．メモリ使用量の計測結果は，どのシングルボードコンピュータにおいても約6 MBであり，実験を行なったシングルボードコンピュータが搭載する最小RAMである512 MBよりも十分小さいメモリ使用量であると考えられる．プロセス実行時間についてはシングルボードコンピュータによって大きな差がみられた．Rock Pi 4では平均3.976秒にて鍵生成から鍵交換までが終了するが，最も性能の低いRaspberry Pi Zeroでは平均45.609秒，Raspberry Pi 3Bは平均22.912秒であった．プロセス実行時間の差については主に公開鍵・秘密鍵の鍵生成プロセスが原因であると考えられる．

5.3. 脅威分析

実装した簡易相互認証システムに対して悪意のある攻撃者によるなりすまし攻撃および中間者攻撃に対して脅威分析を行う．

5.3.1 なりすまし攻撃

なりすまし攻撃(Spoofing)は攻撃者により情報窃取などを目的として他人や他のデバイスになりすます手法である．本システムに対するなりすましは2パターン考えられる．

IoTノードが攻撃者によりなりすまされている場合

この攻撃はは攻撃者がシステムへの情報窃取やシステムを破壊することを目的して，マネジメントステーションへ偽りのIoTノードを登録しようと試みる場合である．正規のユーザがマネジメントステーションが登録を受け付ける際に，正規のIoTノードより早く登録要求を送ることができればマネジメントステーションは偽りのIoTノードに対する認証を行おうとする．このときマネジメントステー

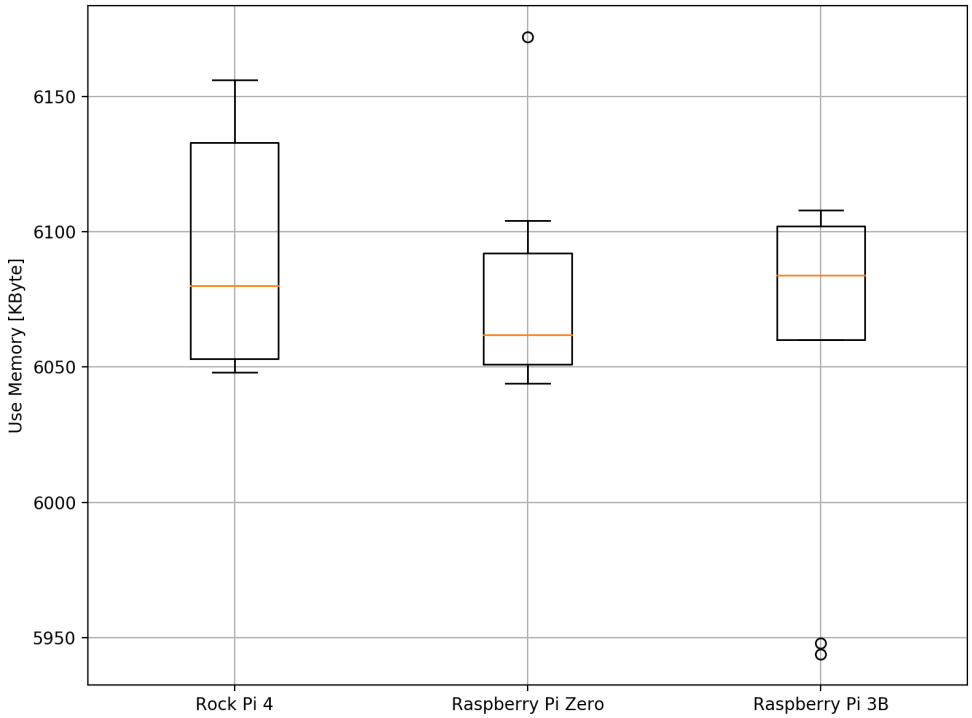


図 5.5 各シングルボードボードコンピュータによるメモリ使用量計測結果

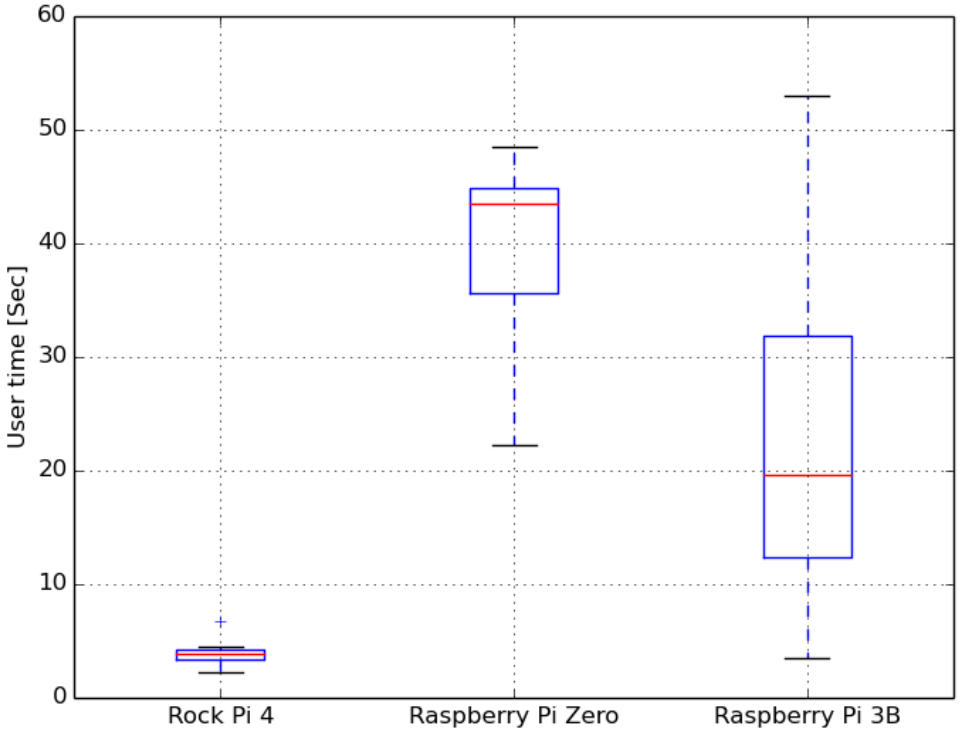


図 5.6 各シングルボードボードコンピュータによる実行時間計測結果

ションは後から来る正規の IoT ノードの登録要求は受け付けない．4.3 で示したプロセス 6 で停止してしまい LCD には何も表示されないため，ユーザはワンタイムキーを知ることができず登録作業を続行することができない．攻撃者がマネジメントステーションへ偽 IoT ノードのワンタイムキーを入力するためには，マネジメントステーションに直接触れることのできる環境または事前にマネジメントステーションを乗っ取り制御できる状態に限定される．このため，なりすましによって偽の IoT ノードを正規マネジメントステーションへ登録することはできない．

マネジメントステーションが攻撃者によりなりすまされている場合

攻撃者がマネジメントステーションに対してスプーフィングを行う場合，ユーザが既知のマネジメントステーション IP アドレスになりすます場合である．この場合，ユーザの手元にある正規マネジメントステーションには IoT ノードから登録要求が来ないため，ユーザは登録を行えず異変に気づくことができる．また，IoT ノードから偽のマネジメントステーションへ登録要求が来た場合，攻撃者がパケット解析などにより得ることのできる情報は，IoT ノードの IP アドレス，MAC アドレスおよびワンタイムキーのハッシュ値のみである．ワンタイムキーは大小英文字および数字の 20 桁から構成されているため，その組み合わせは約 7.04×10^{35} 通り存在する．ここで攻撃者がハッシュ値が一致するワンタイムキーを求めるために総当たり攻撃を行う場合を考える．オープンソースのパスワードクラッカーソフトであり，高速な計算が可能なことで知られる hashcat [27] は，GPU ないし CPU を用いて効率よくハッシュ値を計算することができる．市販の高性能 GPU である GeForce2080Ti との組み合わせでは，約 100 GHash/s の計算 [28] ができることが判明している．この組み合わせにより総当たり攻撃を行った場合，前述の全組み合わせを計算し終えるのに約 2.23×10^{17} 年かかる．このため，ブルートフォースアタックによる突破は現実的ではなく不可能であると考えられる．以上のことから提案システムはなりすまし攻撃に耐性があると考えられる．

5.3.2 中間者攻撃

中間者攻撃はデバイス間の通信に介入し，盗聴または改ざんを行う攻撃である．IoT ノードとマネジメントステーション間に攻撃者が介入し盗聴および改ざんを行う場合，公開鍵の交換時に改ざんを行い任意の鍵にすり替える攻撃が考えられる．提供する簡易相互認証システムでは，公開鍵のハッシュ値の計算しユーザによる確認を行うことで改ざんやすり替えが行われていないかチェックを行うため，悪意のある攻撃者により公開鍵がすり替えられた場合もハッシュ値のチェックによりユーザは異常を検知可能である．ただし，ハッシュ値の確認はユーザによる目視で行う必要があるため，ユーザがハッシュ値の確認を行わない場合やユーザがハッシュ値の確認を誤った場合は公開鍵のすり替えに気づかないため注意を要する．

第 6 章
結　　論

本論文では，IoT デバイスの管理システムにおける初期登録のアイデンティティの問題を解決するために，ワンタイムキーによる簡易な相互認証システムを提案した．提案した簡易相互認証システムは，IoT ノードとマネジメントステーション間における初期登録においてワンタイムキーと公開鍵のハッシュ値の確認を行うことで，ユーザが所望する IoT ノードとマネジメントステーションの相互認証と公開鍵の交換を可能にする．

また提案した簡易相互認証システムの実装を主に C 言語により行い，公開鍵の自動生成およびシステムによる相互認証，公開鍵の交換が正しく動作することを確認した．また，性能の異なるシングルボードコンピュータによる動作確認と計算資源の計測を行った．動作実験により簡易相互認証システムはシングルボードコンピュータの CPU 性能により簡易相互認証全体のプロセス実行時間に大きな差があるが，メモリ使用量についてはほぼ一定の値であることが確認された．

簡易相互認証システムに対する悪意のある攻撃者によるなりすまし攻撃および中間者攻撃に対する脅威分析を行った．なりすまし攻撃に対する脅威分析では，悪意のある攻撃者が IoT ノードおよびマネジメントステーション双方に対するなりすまし攻撃について分析を行い，登録プロセスが続行不可能であることからなりすまし攻撃に耐性があると評価した．また中間者攻撃による脅威分析では，悪意のある攻撃者による公開鍵のすり替えについて分析を行った．提案システムでは簡易相互認証後にネットワーク経由で送信した公開鍵をハッシュ値を用いた確認を行うため，ハッシュ値の確認が正しく行われることで，中間者攻撃に体制があると評価を行った．

6.1. 課題と展望

6.1.1 今後の課題

複数台のデバイスの登録

4 章で実装を行った簡易相互認証・鍵交換システムでは，ユーザにより 20 桁の英数字で構成されるワンタイムキーの確認とマネジメントステーションへの正確な入力を行う必要がある．また鍵交換後も公開鍵のハッシュ値の確認が二度必要で，これは 16 進数で 64 桁が完全に一致するかをユーザの目により確認しなければならない．管理システムへ複数台のデバイスを登録するためには非常に手間と時間がかかることが考えられるため，複数台のデバイス登録のための仕組みが必要である．

6.1.2 今後の展望

将来的な展望として，ユーザによるチェックの簡略化が挙げられる．現在のシステムではワンタイムキーをユーザによる確認および入力を必要としているため，確認や入力ミスが発生する可能性が高い．このため，ワンタイムキーを QR コードなどで表示し，ワンタイムキーのチェックをスマートフォンアプリケーションによりより簡略的に行う手法が考えられる．

また，システムの拡張として IoT ノードによる IoT デバイスの入出力管理およびロギングを行うことが考えられる．実装に利用したシングルボードコンピュータの Rock Pi 4 ではネットワークインターフェイスの他，USB OTG(On-The-Go) によるデバイスモード機能が搭載されており，ネットワークおよび USB デバイスの入出力管理やロギングを行うことで，データの持ち出しや USB によるウィルス持ち込みなどを防ぐ事ができると考えられる．

謝　　辞

本研究の指導教員であり，多くの知見に基づく指導と公私に渡る励ましや私の要領の得ない相談に対し的確に助言をしていただきました慶應義塾大学大学院メディアデザイン研究科の砂原秀樹教授に心から感謝致します．砂原先生は私が KMD を知るきっかけでもあり，Network Media Project だけでなく SecCap などを通じて過ごした 2 年間は人生の貴重な経験でした．また本研究の副指導教員であり，日頃より多くの助言やご指導ときには人生相談に乗っていただいた慶應義塾大学大学院メディアデザイン研究科の加藤朗教授に心から感謝致します．長年 KMD を支えていただき今年度をもって引退される古川享教授には副査として審査を担当していただきました，心より感謝致します．また，研究や Network Media Project の活動で様々なご指導をいただきました慶應義塾大学大学院メディアデザイン研究科の山内正人特任講師に心から感謝いたします．KMD において学外における経験が多数できたことは本当に幸運でした．

研究における助言や研究以外の活動において一緒に仕事をさせていただいた慶應義塾大学大学院ディアデザイン研究科博士課程の加藤大弥さんに心から感謝いたします．たくさんの技術や知識を得ることができました．また，Network Media Project の石井美穂さん，太田智美さん，PLAY の岡田光代さんには NetworkMedia の活動だけではなく，私的な部分でも何度も支えていただきました．本当にありがとうございました．また Network Media の同期である，安藤亮介くん，梶浦瑶子さん，冨安香澄さん，岩本佑太さんに感謝いたします．様々な面でご迷惑をおかけしてしまいましたが，皆様のおかげで卒業することができました．

最後に修士過程までの進学を認めていただき，支援していただいた家族に感謝いたします．

参 考 文 献

[1] Gordon E Moore, et al. Cramming more components onto integrated circuits, 1965.

[2] Mark Weiser. The computer for the 21st century. https://web.archive.org/web/20141022035044/http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html, 1988 [Archive 2014]. [Online; accessed Jan-2020].

[3] 坂村健. ユビキタス・コンピューティング新時代. オペレーションズ・リサーチ: 経営の科学, Vol. 49, No. 4, pp. 203–209, 2004.

[4] Kevin Ashton, et al. That ‘internet of things’ thing. *RFID journal*, Vol. 22, No. 7, pp. 97–114, 2009.

[5] Japan Network Infomation Center. インターネット用語 1 分解説〜iot とは〜 – jpnic. https://www.nic.ad.jp/ja/basics/terms/iot.html, 2014. [Online; accessed Nov-2019].

[6] Amazon Web Services. Iot とは？(internet of things) — aws. https://aws.amazon.com/jp/iot/what-is-the-internet-of-things/, 2019. [Online; accessed Jan-2020].

[7] Jacob Morgan. A simple explanation of’the internet of things’. *Retrieved November*, Vol. 20, p. 2015, 2014.

[8] Google Inc. Google home - スマートスピーカー - google ストア. https://store.google.com/jp/product/google_home. [Online; accessed Jan-2020].

[9] Amazon.com. Amazon — echo dot - コンパクトスマートスピーカー. https://www.amazon.co.jp/dp/B07PFFMQ64/, 2019. [Online; accessed Jan-2020].

[10] ロボットスタート株式会社. 日本で買える iot デバイス 330 個をまとめた「iot デバイスマップ 2018」公開　ダウンロード無料. https://robotstart.info/2018/04/17/iot-device-map-2018.html, 2018. [Online; accessed Jan-2020].

[11] Insecom.org. Insecam – world biggest online cameras directory. https://www.insecam.org, 2019. [Online; accessed Nov-2019].

[12] 中澤祐樹, 佐々木良一, 猪俣敦夫ほか. 野良 iot の地域特性の調査と分析. マルチメディア, 分散協調とモバイルシンポジウム 2017 論文集, Vol. 2017, pp. 1138–1144, 2017.

[13] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, Vol. 50, No. 7, pp. 80–84, 2017.

[14] 一般社団法人重要生活機器連携セキュリティ協議会. Iot 分野共通セキュリティ要件 ガイドライン 2019 年版 ver. 1.0. https://www.ccds.or.jp/certification/document/IoT%E5%88%86%E9%87%8E%E5%85%B1%E9%80%9A%E3%82%BB%E3%82%AD%E3%83%A5%E3%83%AA%E3%83%86%E3%82%A3%E8%A6%81%E4%BB%B6%E3%82%AC%E3%82%A4%E3%83%89%E3%83%A9%E3%82%A4%E3%83%B32019%E5%B9%B4%E7%89%88_ver1.0.pdf, 2019. [Online; accessed Jan-2020].

[15] The OWASP Foundation. Owasp internet of things. https://owasp.org/www-project-internet-of-things/, 2018. [Online; accessed Jan-2020].

[16] NOTICE SUPPORT CENTER. Notice — サイバー攻撃に悪用されるおそれのある iot 機器の調査、注意喚起を行うプロジェクト. https://notice.go.jp/. [Online; accessed Jan-2020].

[17] 独立行政法人情報処理推進機構 技術本部セキュリティセンター. 安心相談窓口だより 第 16-13-359 号. https://www.ipa.go.jp/security/anshin/mgdayori20161125.html, 2016. [Online; accessed Jan-2020].

[18] 盛合志帆. 暗号の安全性評価 nict news. https://www.nict.go.jp/publication/NICT-News/1303/01.html, 2013. [Online; accessed Jan-2020].

[19] 小池英樹, 高田哲司, 増井俊之ほか. 画像を用いた個人認証手法. 情報処理, Vol. 47, No. 5, pp. 479–484, 2006.

[20] 越前功, 大金建夫ほか. 写真からの指紋復元の脅威とその対策技術. 情報処理, Vol. 58, No. 9, pp. 824–829, 2017.

[21] Raspberry Pi Foundation. Buy a raspberry pi zero – raspberry pi. https://www.raspberrypi.org/products/raspberry-pi-zero/, 2017. [Online; accessed Jan-2020].

[22] Raspberry Pi Foundation. Raspberry pi 4 model b specifications – raspberry pi. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/, 2019. [Online; accessed Dec-2019].

[23] Interface 編集部/編. Interface 2018 年 9 月号. CQ 出版, 2018.

[24] RADXA. Rock pi – a series of single board computers. https://rockpi.org/. [Online; accessed Jan-2020].

[25] Hemant Jain. 多様化する iot の os がインターネットの安全性にとっては災いに. https://archives2011-2018.fortinet.co.jp/security_blog/170104-iot-operating-systems-bad-news-for-the-safety-of-the-internet.html, 2017. [Online; accessed Jan-2020].

[26] 一般社団法人日本ネットワークインフォメーションセンター. Ipv4 アドレスの在庫枯渇に関して. https://www.nic.ad.jp/ja/ip/ipv4pool/, 2017. [Online; accessed Jan-2020].

[27] Hashcat. hashcat - advanced password recovery. https://hashcat.net/hashcat/, 2018. [Online; accessed Jan-2020].

[28] hashcat. hashcat(@hashcat) — twitter. https://twitter.com/hashcat/status/1095807014079512579, 2019. [Online; accessed Jan-2020].