論理デバイスプロキシを利用した IoT セキュリティプラットフォームの提案

Proposal of the IoT Security Platform using Logical Device Proxy

今野 裕太 †　佐藤 健哉 †
Yuta Konno　Kenya Sato

1. はじめに

近年，IoT(Internet of things) の発展により，今までネットワークに接続されていなかったモノが接続され，総務省が公表した平成 27 年度版情報通信白書によると，2020 年には 500 億台以上の IoT デバイスがネットワークに接続される見込みである [1]．あらゆるモノが IoT によりネットワークに繋がることで，情報流通を促進し ビッグデータを収集，解析が行われることに加え，ローカルネットワーク内においても，今後様々な IoT デバイスの登場により，今まで相互接続されていなかった機器同士が接続されると考えられる．以上のような変化により，様々な課題の解決や新たな価値の創出が期待されている．

一方で IoT デバイスは一般的な PC などの既存の機器と比較すると，CPU 等のリソースを多く持たないという特徴がある．特にセンサーなどの小型デバイスになるとその特徴は顕著になり，暗号化等のセキュリティ対策の適用は困難となる．

本研究はセキュリティ対策を施し仮想的に作成した論理デバイスを利用し，実際の IoT デバイスの通信を中継することで，セキュアな通信環境を提供するプラットフォームを提案する．

2. IoT デバイスのセキュリティ

2.1 IoT デバイスの特徴と問題点

従来の PC 等の機器と比較した場合，IoT デバイスは一般に小型で安価であるという特徴を持つ．一方でその特徴から CPU などのリソースを十分に保持していないため，適用できる機能が限られるという問題がある．特にセンサー機能等に特化した IoT デバイスでは，その問題が特に顕著に現れ，暗号化等のセキュリティ対策の適用は難しい．現在も世界中でセンサーの小型化・低消費電力化・低価格化が進展している．現状では必要な機能を詰め込んで実装するのが現状である [4]．

また今後は多種多様な IoT デバイスが大量に開発製造されるものと予想されるため，各 IoT デバイスに求められるセキュリティ要件は様々である．加えてあらゆるデバイスが相互接続することにより，開発者が想定していない通信が発生し，その際にはセキュリティ要件の変化が想定されることからも各デバイスに対し柔軟なセキュリティ対策を適用することが必要となる [2]．特に私達の生活に IoT デバイスがより身近になり，これらのデバイスで個人情報等のプライバシーに関わる情報がやり取りされるようになると考えると，各デバイスへのセキュリティ対策の適用は重要な課題である [3]．

2.2 関連システム

IoT デバイスのセキュリティを提供する方法として IoT クラウドプラットフォームの利用が考えられる．その一種である SORACOM が開発した SORACOM beam は，IoT デバイスにかかる暗号化等の高負荷処理をクラウドにオフロードできるサービスである [5]．実際の利用手順としては，IoT デバイスからアップロードされた通信を閉域網を経由してクラウドへ転送し，処理を適用後目的地のサーバへ転送する．

このようにすることで，CPU や電力制限によって暗号化処理ができない IoT デバイスの通信を暗号化することができ，また証明書の管理等の煩雑な処理をクラウドにオフロードすることで，暗号化された通信を簡単に利用することが可能になる．

2.3 関連システムの問題点

関連システムの問題点として，クラウドへオフロードした処理の遅延が考えられる．通信速度はクラウドが存在するリージョンに依存することに加え，今後は IoT デバイスの普及に伴いデータの爆発的増加がもたらされ，すべてのデータをクラウド上で処理することは困難となる [6][7]．

またセキュリティの観点からも，データが一箇所に集中することにより攻撃のリスクが高まることに加え，今後 IoT デバイス間で個人情報のやり取りが行われるようになった場合，プライバシーの保護に関する懸念も発生する．遅延や通信量の問題に対してはエッジコンピューティングのような，デバイスに近い場所でデータ処理を行い処理を分散する手法があるが，セキュリティ面からも前述の問題点を考慮した手法として有用であると考えられる．

† 同志社大学大学院理工学研究科情報工学専攻，
Graduate School of Information and Computer Science,
Doshisha University

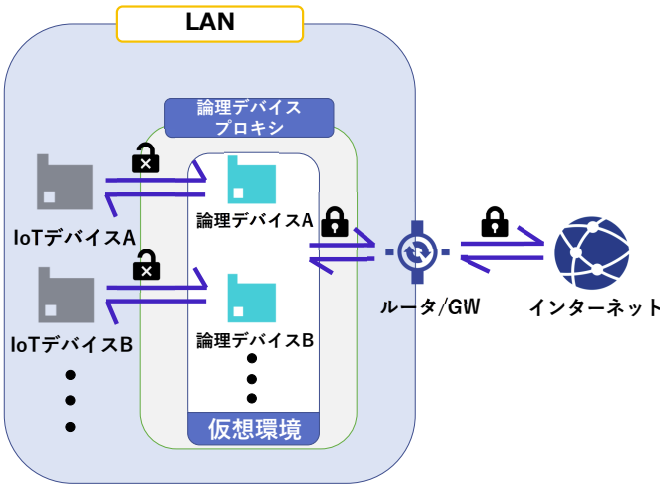

図 1　提案システムの構成

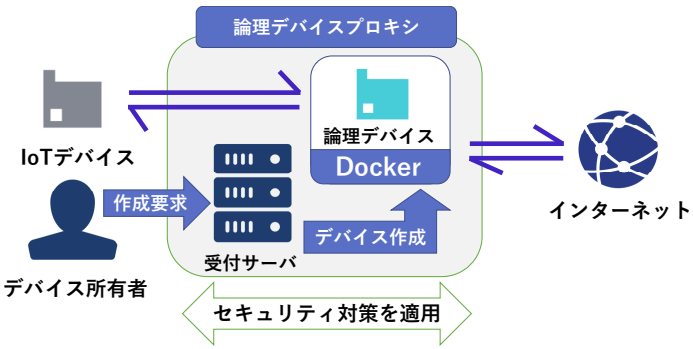

図 2　実装環境の構成

3.　提案システム

3.1　概要

提案システムでは論理デバイスというセキュリティ対策を適用可能なデバイスを，サーバ上に仮想的に作成する．ここに IoT デバイスがリソースの都合上適用できないセキュリティ対策をオフロードし，この論理デバイスが IoT デバイスの通信を中継することで，本来 IoT デバイスに適用したいセキュリティ対策を実現する．

論理デバイスにはセキュリティ対策を行うために十分なリソースを与えることにより，適用する対策は元の IoT デバイスのリソース量に依存しないことに加え，IoT デバイスごとに論理デバイスを作成することで，各デバイスに応じた柔軟なセキュリティ対策が実現可能となる．また論理デバイスプロキシは IoT デバイスと同一 LAN に配置することで，クラウド上で処理を行う場合と比べ，個人情報などを扱う際のプライバジーに対する懸念も考慮することが可能となる．

3.2　システム構成

提案システムの構成を図 1 に示す．本提案手法の構成要素としては IoT デバイス，論理デバイス，論理デバイスプロキシ，仮想環境から構成される．

- IoT デバイス

  本稿で扱う IoT デバイスはセンサーを始めとした，CPU 等のリソースを十分に持たずデバイスに直接セキュリティ対策を適用出来ないデバイスと定義する．

- 論理デバイス

  IoT デバイスに要求されるセキュリティ対策を，仮想的に実現したものである．IoT デバイスからの通信をプロキシのように中継しセキュリティ対策を適用する．セキュリティ対策ごとに作成し IoT デバイスと紐つけることで，当該デバイスに応じた必要な対策を実現することが可能である．

- 論理デバイスプロキシ

  作成した論理デバイスを稼働しているサーバを論理デバイスプロキシと定義する．IoT デバイスがある同じ LAN 上に設置し，ユーザからの作成要求を元に仮想的に論理デバイスを作成する．

- 仮想環境

  論理デバイスプロキシで提供される論理デバイスの実行環境である．論理デバイスが作成される際に要求されるリソースを十分に提供することが可能である．

4.　実装

4.1　実装環境

本研究の実装環境，実装環境の構成をそれぞれ表 1，図 2 に示す．IoT デバイスには RasberryPi 3 を利用し，温度，湿度，気圧の三項目を計測するセンサデバイスを作成した．これらの情報は図 3 のようにラズベリーパイのアドオンボードである Sense HAT により取得している．論理デバイスの作成方法としては軽量なアプリケーション実行環境である Docker を利用した．論理デバイスを Docker で作成されるコンテナ上で稼働させることで複数の論理デバイスをリソース, オーバーヘッドを抑えて作成できる事に加え，Docker Hub より配布される Docker Image を用いることで容易に作成可能となる．また今回扱う通信プロトコルとしては http(REST) を想定する．

4.2　動作手順

図 4 に併せて IoT デバイスの所有者であるユーザが本提案システムを利用する場合の動作手順を示す．

1. IoT デバイスの所有者はデバイスを LAN 内に接続した後，論理デバイスプロキシの受付サーバへアク

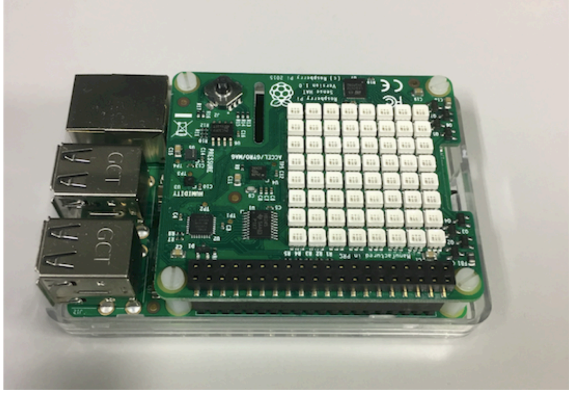

図 3　Sense Hat による実装

セスする．

2. 論理デバイスプロキシは LAN 内に IoT デバイスが接続されたことを確認した後，デバイス所有者へデバイス情報入力画面を送信．
3. デバイス所有者はデバイス情報である ID,Password,Image 情報を入力画面に入力．
4. 論理デバイスプロキシは入力された Docker Image 情報を基に Docker Hub から Docker image を入手．
5. 入手した Docker Image を基に論理デバイスプロキシ内に論理デバイスを作成．
6. デバイス情報を用いて IoT デバイスに接続し，通信を論理デバイスを経由するように設定．
7. 設定が完了し，IoT デバイス・論理デバイス間の通信が確立された後，デバイス所有者へ論理デバイス作成結果を送信．

表 1　実装環境

種類	項目	説明
IoT デバイス	計算機	RasberryPi 3
	CPU	1.2GHz quad-core
	OS	Ubuntu mate
	センサー	Sense HAT
	プロトコル	http(REST)
論理デバイスプロキシ	使用ソフト	Docker,Apache,mysql
	CPU	3.30GHz quad-core
	メモリ	4GB
	OS	Ubuntu 14.04
	プロトコル	http/https(REST)

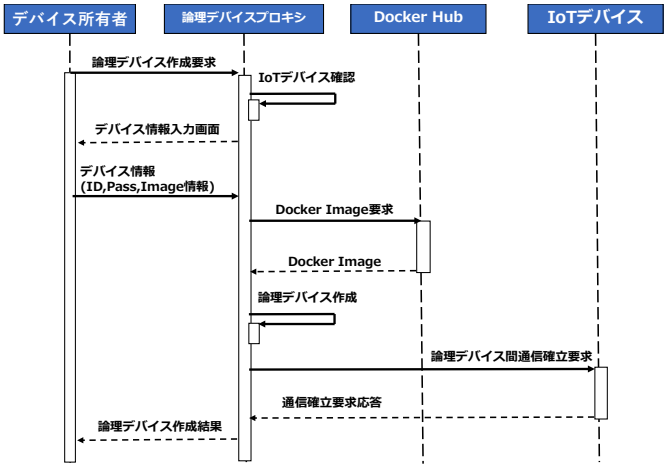

図 4　提案システムの動作手順

4.3　想定ユースケース

本提案システムを用いた想定ユースケースを以下に示す．各 Docker イメージは Docker Hub というユーザーが作成したコンテナをアップロードして公開・共有できるサービスを利用することを想定する．

- リソースを使うセキュリティ対策を予め提供する場合

  SSL による通信の暗号化など，IoT デバイスのリソースを多く利用するために適用できないセキュリティ対策を予め Docker イメージとして提供し，論理デバイス上で実現する．

- インシデント発生時などに対策を提供する場合

  事前に提供していたセキュリティ対策では想定していなかったインシデント等が発生した場合等に，当該デバイスの持つリソース量に依存せず，追加のセキュリティ対策を提供することが可能となる．

5.　評価

5.1　評価内容

本研究の評価として提案システムを適用した上で,IoT デバイスからセンサー情報を JSON 形式で HTTP POST で宛先へ送信し，通信結果の応答が返ってくるまでの時間（ラウンドトリップタイム）を計測した．宛先はクラウド上のサーバ，ローカルネットワーク内のサーバの 2 パターンを用意した．

比較対象としてセキュリティ対策を適用しない場合，関連システムであるセキュリティ対策をクラウドへオフロードした場合（以下，関連システム）についても計測した．

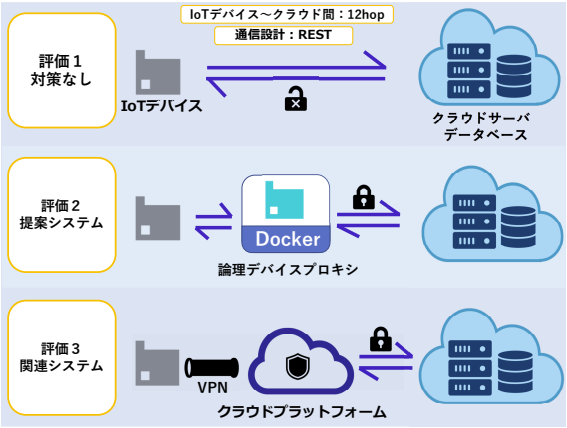

図 5　評価環境構成 (宛先：クラウド)

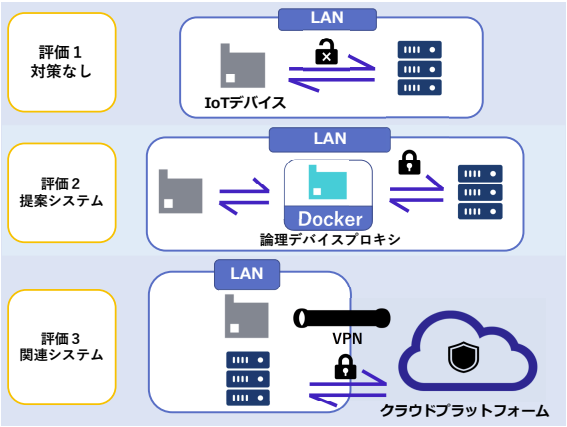

図 6　評価環境構成 (宛先：ローカル)

5.2　評価環境

今回の評価適用するセキュリティ対策としては，鍵長は 1024bit の SSL による暗号化を行い，GO 言語により実装した．また関連システムでは実際のサービスを模して，セキュリティ対策を適用するクラウドまでは VPN による専用線を利用し，クラウド上で SSL による暗号化を行った後，目的地であるサーバへ接続を行っている．

今回行う 2 パターンの評価環境構成を図 5，6 に，評価環境を表 2 に示す．なお IoT デバイスと，論理デバイスプロキシの環境は実装環境と同様である．

5.3　評価結果

SSL による暗号化を行わなかった場合，提案システムを用いた場合，関連システムであるクラウド上で暗号化処理をおこなった場合の比較結果を図 7，8 に示す．ラウンドトリップタイムは 10 秒間通信を行った際の平均値を算出した．

6.　考察

6.1　性能

図 7 の評価結果より，本提案システムである Docker を用いた論理デバイスによるセキュリティ対策は，対策なしの場合と比較し約 40 %増の遅延は発生しているが，関連システムの遅延以下であることから実用性があると考えられる．

今回の関連システムの遅延は，クラウドへセキュリティ対策をオフロードしていることが大きく影響していると考えられる．特に図 8 の結果から分かるように，今後増加していくと考えられるローカルネットワーク内での IoT デバイスの通信に対してセキュリティ対策を適用を試みた場合，提案システムとのラウンドトリップタイムの差が大きくみられる．前述したようにクラウドからの通信遅延に加え，外部ネットワークからローカルサーバ宛の通信の際には，ルータにポートフォワーディング等の処理が必要となり，その部分においても追加のオーバヘッドが発生していると考えられる．

さらに今後 IoT デバイスが普及した際には，クラウドへの通信量が爆発的に増加するため，帯域の輻輳といった問題も発生すると考えられる．そのため通信遅延は更に増加すると思われることから，通信遅延が許容が難しいアプリケーションを利用する際にセキュリティ対策が必要な場合には本提案システムが有効となる．

6.2　信頼性

IoT デバイスを用いたシステムの安心安全を確保するための機能として，IPA により IoT 高信頼化機能が定義されており，IoT 高信頼化要件として開始，予防，検知，

表 2　評価環境

種類	項目	説明
宛先クラウド	使用ソフト	Apache,mysql
	OS	Ubuntu 14.04
	メモリ	4GB
	CPU	2.4 GHz single-core
	クラウド	IDCF Cloud
	リージョン	東日本
宛先ローカルサーバ	使用ソフト	Apache，mysql
	OS	Ubuntu 14.04
	メモリ	2GB
	CPU	3.2 GHz quad-core
クラウドプラットフォーム	使用ソフト	Apache
	使用言語	GO 言語
	OS	Ubuntu 14.04
	メモリ	4GB
	CPU	2.5 GHz single-core
	クラウド	AWS
	リージョン	東京

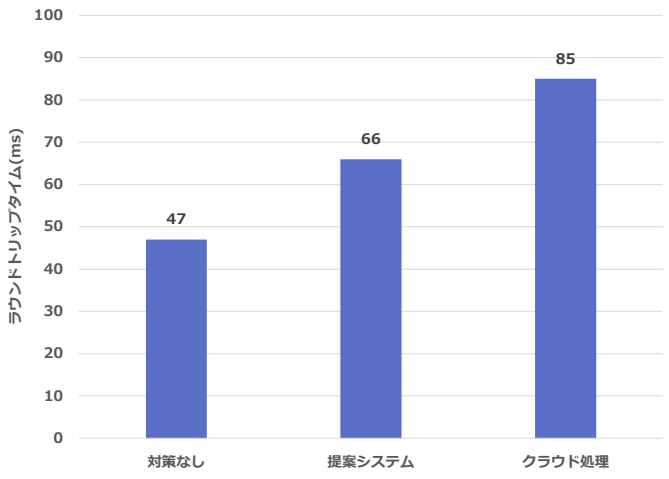

図 7　ラウンドトリップタイム（宛先：クラウド）

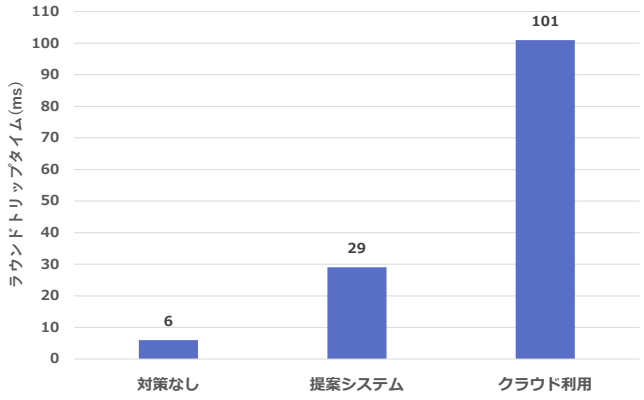

図 8　ラウンドトリップタイム（宛先：ローカル）

回復，終了の５つの局面に分けてそれぞれセキュリティ要件が定義されている [8]．今回は前述の５つより，システム稼働中の局面である予防，検知，回復の 3 つにおける高信頼化要件に対し，提案システムの有効性について考察する．

- 予防の局面における考察

  予防の局面での高信頼化要件は，稼働中の異常発生を未然に防止できることである．これに対応する IoT 高信頼化機能としてはログ収集機能，暗号化機能等があり，異常の予兆の把握，資産の保護を実現する．提案システムを用いることで，リソース量の関係で通常の IoT デバイスに適用できない機能であっても適用可能となる．

- 検知の局面における考察

  検知の局面での高信頼化要件は，稼働中の異常発生を早期に検知できることである．これに対応する IoT 高信頼化機能としては状態監視機能，ログ収集機能があり，異常発生の検知や発生原因の特定を実現する．提案システムを用いることで予防の局面同様，デバイスのリソース量に依存せず，求められる機能を実現できることに加え，論理デバイスは各 IoT デバイスごとに作成するため，個々の機器に応じた細かな検知ルール等を適用可能となる．

- 回復の局面における考察

  回復の局面での高信頼化要件は，異常が発生した場合に稼働の復旧ができることである．特に IoT では様々なデバイスが相互通信を行うため，事前に予測していなかった異常が発生することが考えられる．今回の環境では各セキュリティ対策は Docker Hub を通して Docker イメージとして提供することで，事前に作成したセキュリティ対策だけでなく，追加のセキュリティ対策も配布・適用が容易である．

7.　まとめ

本研究では今後も増加すると考えられる IoT デバイスの特徴である, 十分なリソースを持たないことによるセキュリティ対策の適用の困難さを問題点として挙げた．提案システムのように IoT デバイスの通信を中継する論理デバイスを仮想的に作成し，その上にセキュリティ対策を適用することで，IoT デバイスのリソース量に依存しないセキュリティ対策の実現が可能となる．論理デバイスの作成には Docker を用いて実装し，関連システムである IoT クラウドプラットフォームについても環境を作成し，ラウンドトリップタイム比較による性能評価をクラウドサーバ宛て，ローカルネットワーク上のサーバ宛の 2 パターンで行った．

また IPA により定義された IoT 高信頼化要件について予防，検知，回復の観点から考察することで，提案システムの持つ信頼性について示した．

謝辞

本研究の一部は JSPS 科研費 16H02814 の助成を受けたものである．

参考文献

1) 総務省,"平成 27 年度情報通信白書"；
http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h27/pdf/27honpen.pdf,2017

2) IPA,"IoT におけるセキュリティの脅威と対策"；
https://www.ipa.go.jp/files/000049819.pdf,2017

3) The Quest for Privacy in the Internet of Things；
Porambage, Pawani, et al. IEEE Cloud Computing 3.2 pp.36-45，2016.

4) 大野 浩之，鈴木 裕信，北口 善明，IoT 時代に資するセキュリティゲートウェイの構築，マルチメディア，分散，協調とモバイル（DICOMO2015）シンポジウム論文集，pp.903-910，2015.

5) SORACOM, "SORACOM Junction", ; https://soracom.jp/services/junction/,2017

6) 経済産業省, "分散戦略 WG 中間取りまとめ", ; http://www.meti.go.jp/report/whitepaper/data/pdf/20161129001_01.pdf,2017

7) 総務省, "IoT 共通基盤技術の確立・実証 基本計画書", ; http://www.soumu.go.jp/main_content/000426656.pdf,2017

8) IPA,"「繋がる世界の開発指針」の実践に向けた手引き", http://www.ipa.go.jp/files/000059278.pdf,2017