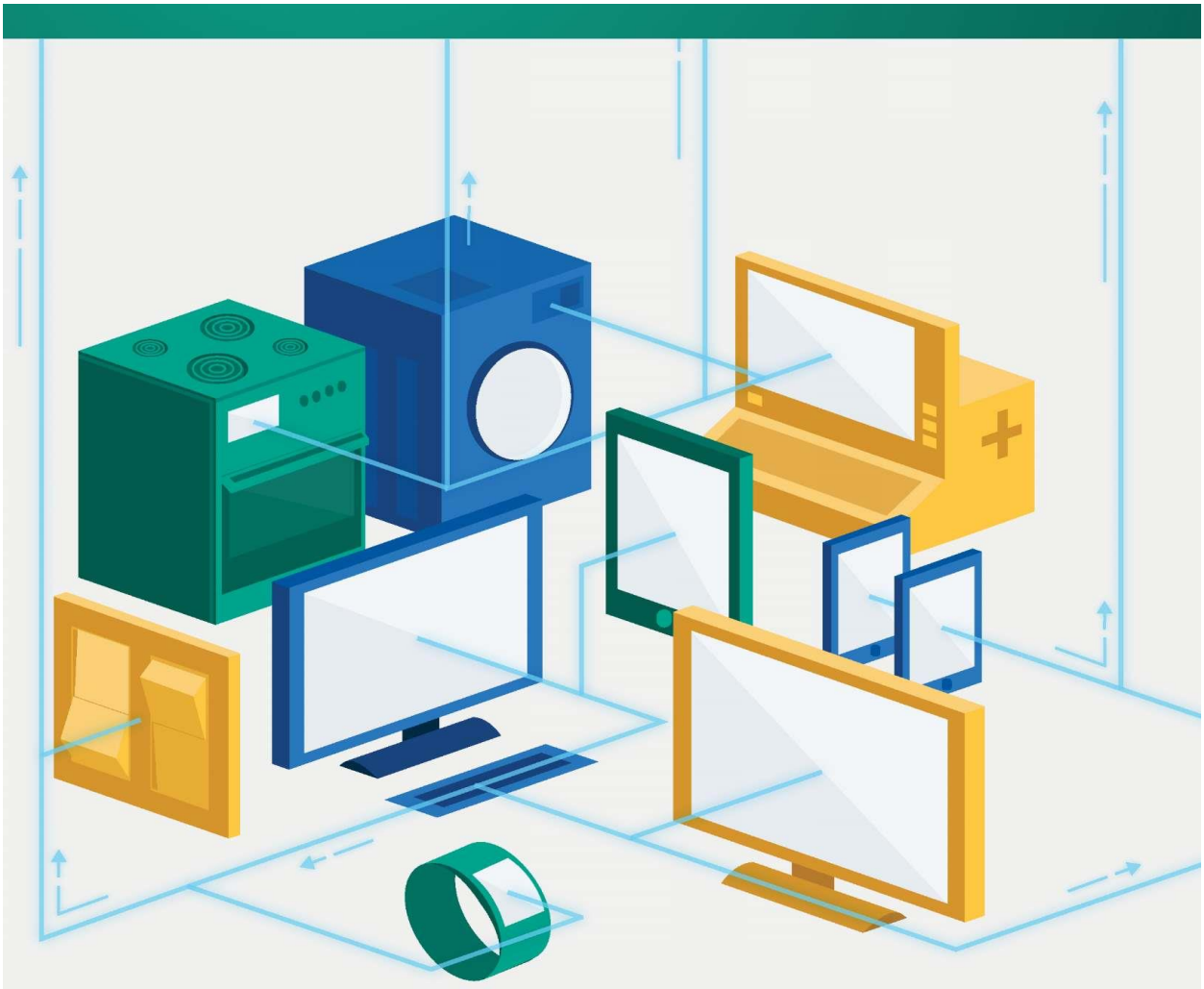
「つながる世界」を
破綻させないための
セキュアなIoT製品開発
１３のステップ



CSA IoT Working Group
翻訳：CSA ジャパン IoT ワーキンググループ

目次

作成者等

執筆とりまとめ

Priya Kuber

Brian Russell

Dr Shyam Sundaram

CSA IoT WG リーダー

Brian Russell, Leidos

主たる執筆者

K S Abhiraj
Sateesh Bolloju
Steve Brukbacher
Giuliana Carullo
Ravi Dhungel
Thomas Donahoe
Raghavender Duddilla
Drew Van Duren
Loic Falletta
.

Luciano Ferrari
Ayoub Figuigui
Masaaki Futagi
Mark Grimes
Aaron Guzman
Heath Hendrickson
Sabri Khemissa
Paul Lanois
Jinesh M.K.

Srinivas Naik
Chalani Perera
Sewmi Rajapaksha
Lakmal Rupasinghe
AK Sharma
Mark Szewczul
Srinivas Tatipamula
Sudharma Thikkavarapu
Kim White

CSA Global スタッフ

John Yeoh

JR Santos

デザイン

Stephen Lumpe

日本語翻訳の発行に寄せて

この文書は、米国 Cloud Security Alliance の IoT ワーキンググループが 2016 年 6 月に発表した *"Future-proofing The Connected World：13 Steps to Develop Secure IoT Products"* の日本語翻訳版です。IoT 製品の安全を確保するために必要な様々な切り口からの考察を含むガイダンスであり、非常に有益なものと考え、CSA ジャパン IoT ワーキンググループのメンバーを中心に分担して翻訳、半年近くをかけてレビューを行った末、ようやくリリースにこぎつけました。翻訳やレビューに協力いただいた皆様に感謝するとともに、この文書が、IoT 開発に携わる方々の仕事に役立つよう期待します。

2017 年 5 月　　CSA ジャパン IoT WG リーダー　　二木真明

翻訳に協力いただいた皆様（敬称略：五十音順）

氏名	所属
阿賀　誠	富士通株式会社
有本　真由	個人
甲斐　賢	株式会社日立製作所
勝見　勉	株式会社情報経済研究所、CSA ジャパン事務局
上村　竜也	個人
新宮　貢	個人
鶴田　浩司	個人
諸角　昌宏	合同会社鯱パートナーズ、CSA ジャパン事務局
山崎　英人	個人
山下　亮一	個人

とりまとめ　　二木　真明　　アルテア・セキュリティ・コンサルティング
翻訳レビュー　（CSA ジャパン IoT WG）

【改版履歴】
日本語版：バージョン 1.1 （6 月 13 日修正）
リンク切れ等を修正しました。
初版に関して岡田良太郎氏（OWASP Japan）より貴重なフィードバックをいただきました。

発行に寄せて

IoT のセキュリティを、頻発する攻撃や、数限りなく存在する攻撃者に対して確保する必要については、近年頻繁に語られている。しかし、それをいかに実現するかについての詳細なガイダンスはなかなか見当たらない。このガイダンスを作るにあたっての難しさの一つは、IoT のコンセプトが多様な業種、多彩な製品やシステムに幅広く当てはまることにある。

クラウドセキュリティアライアンス（CSA）の IoT ワーキンググループは、2015 年 4 月に「Security Guidance for Early Adopters of the IoT」（邦訳 : IoT 早期導入者のためのセキュリティガイダンス）を発行してシステムレベルのセキュリティについてガイダンスを提供した。しかし、IoT システムのセキュリティは、その一番弱いリンクのレベルになってしまう。本書によって、我々は、IoT システムを構成する各製品をセキュアにするために実用性と有効性を伴ったガイダンスを提供しようと試みた。それにより、IoT 製品の全体としてのセキュリティ・レベルが良くなることを目指して。

従来の製品ラインを IoT 対応可能なデバイスに移行することを目指している企業にとって特に、本書が役立つことを期待している。つまり、新たにネットワークにつながった機器を悪用してやろうとする無数の悪いやつらのことを理解する経験や背景を持たない製造業者のために、である。事業者はよく、セキュリティの戦略が不十分だと指摘される。しかし、その不十分なものが何であり、どうすれば無くせるのかを正しく理解できるための参考となるガイドは提供されてこなかった。

スタートアップ段階の企業たちにも、このガイドが役に立つことを期待している。ネットにつながる製品やシステムを扱うスタートアップは早く製品を世に出さなければならない。開発の初期の段階で、その製品をセキュアにする正しい能力を持つ人を確保するのは容易なことではない。本書は、B2C または B2B の IoT 製品に押し寄せる深刻な脅威を、少なくとも緩和するのに役立つセキュリティ戦略を構築するための、出発点となる。

業界では、IoT 製品やシステムをセキュアにすることは、とてもできない相談だとよく聞く。しかし、CSA IoT ワーキンググループの面々のようなボランティアの助けにより、自分の製品が攻撃のリスクにさらされているのは知りつつも、そのリスクを回避するのに何から始めればいいのかがわからない製品開発の従事者に手を差し伸べるようとすることはできる。よってもって、本処理執筆とレビューに参加した多くのボランティアに感謝申し上げたい。併せて、各々の IoT ガイダンスにより支援の手を差し伸べているほかの二つの組織　米連邦通信委員会 FCC と、Securing Smart Cities Initiative にも賛辞を贈りたい。この両者は CSA IoT WG に協力してくれている。

最後に、本書は共有財産であり、ガイダンスは随時更新されるものと期待している。コメントがあればどんなものでも、WG にお寄せいただきたい。次回更新時に反映させていただく。本書が皆様の役に立つことを祈って。

Brian Russell, Chair IoT Working Group, Cloud Security Alliance

はじめに

本書は、網羅性を意識した結果、IoT 製品をセキュアにするための多くの点に触れており、結果として大部のものになっている。一方しばしば、製品開発のライフサイクルにおいて考慮すべき重要なセキュリティ要素を、迅速に洗い出すことが必要となる。その点を考慮して、読者のために、以下に 5 大セキュリティ課題をリストアップした。我々の考えでは、最低限この 5 つの要素に焦点を当てれば、IoT 製品のセキュリティ・レベルの改善に着手することができる。このリストがあるからと言って、本書の残り部分を読まなくていいわけではない。しかし、この 5 つのポイントにより、開発チームが正しい方向に素早く一歩を踏み出すための方策が得られる。

IoT 製品の開発者はまず、以下のセキュリティ工学の実践課題に取り組まなければならない。

序論

本書の目的は、IoT デバイスの開発者に、その製品が直面する脅威についての理解を提供することである。本書はまた、それらの脅威に対する防御に役立つツールやプロセスについても述べる。IoT システムは複雑で、関連する分野はデバイス、ゲートウェイ、モバイル、アプライアンス、Web サービス、分析装置その他多岐にわたるが、本書は主として「デバイス」（すなわち”Things”）に焦点を当てる。

ITU-T Y.2060 は、IoT におけるデバイスの定義を「通信能力が必須であり、検知、作動、データ取得、データ保存、データ処理の能力を任意に持っている機器」と定義している。しかし、セキュアな IoT デバイスの概念はそう簡単に定義できるものではない。本書の目的においては、セキュアな IoT デバイスは、攻撃者を他の目標に向かわせるに足るセキュリティ対策を実装したデバイス、と定義する。ネットにつながるもので完全にセキュアなものはない。しかし、攻撃者が攻撃し続けるのは不合理と考えるのに十分な程度に侵入が高くつくように作り上げることは可能である。

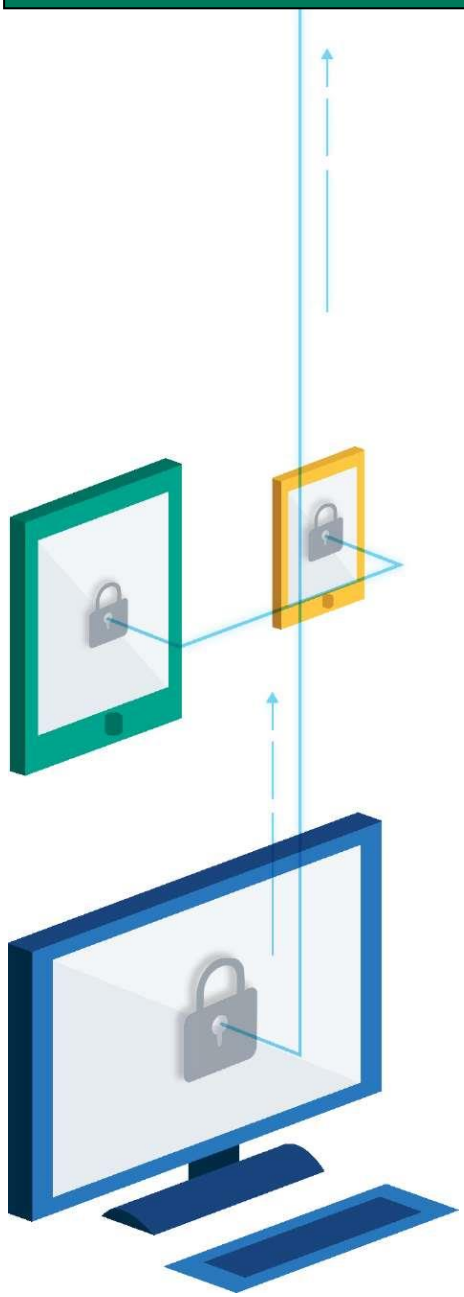
IoT デバイスの構成はさまざまである。特定のハードウェア上の最小限のファームウェアで走る簡単なセンサーで、OS が不要のものであるかもしれない。あるいはスタンドアローンのアプライアンスで、最小単位の複合構成をとり、リアルタイム OS と、ロジックの実装のためのソフトウェアを搭載しているかもしれない。IoT デバイスはまた、デバイスの設定変更のオプションをサポートするフル機能の Web サーバーを備えた物体かもしれない。さらには、何百もの ECU(Electronic Control Unuts)と、可変型無線通信インターフェースを持ち、何百万行ものコードを搭載したコネクテッドカーであるかもしれない。**本書では、IoT デバイスと IoT 製品の用語を区別せずに使う。**

本書の対象範囲

このセキュアな設計と開発のためのガイダンスは、以下のものを提供する：

1. IoT デバイスのセキュリティ課題に関する検討
2. CSA IoT WG が実施した IoT セキュリティの調査結果
3. IoT 開発プラットフォームで利用可能なセキュリティオプションに関する検討
4. IoT デバイスの分類と簡単な脅威分析
5. デバイスのセキュアな設計と開発プロセスに関する推奨事項
6. セキュリティ技術者が開発過程で使うべき詳細なチェックリスト
7. IoT 製品に対応する脅威のマッピングの事例を盛り込んだ付録集

IoT セキュリティの必要性



IoT はすでに目の前にある。それは個人、企業、産業の変革を起こしつつある。2015 年には、様々なタイプの IoT 製品が市場に登場するのを目の当たりにした。同時に、IoT セキュリティが現実の課題であるということを示す実際の調査結果も目にした。本編で簡単に触れる調査の結果により、IoT 製品のセキュリティの必要度が高いことが理解され始めた。その必要性には以下のものがある。

- 個人のプライバシー保護と、個人特定情報及び個人医療情報の開示の制限の必要性
- 営業上のデータの保護と、機微情報の開示の制限の必要性
- IoT 製品が DDoS 攻撃またはネットワーク向け発信拠点に使用されることに対する防御の必要性
- サイバー・フィジカルシステムに侵入されることに伴う（物理的な）破損や傷害に対する防御の必要性

以下に、IoT の世界における事象とトレンドについて述べる。それにより、IoT 製品の開発者が身に付けるべき教訓を示したい。

IoT 製品によるプライバシー侵害

Rapid7 は最近、「赤ちゃんモニターの情報漏洩と脆弱性」というレポートを公表した。それによれば、いくつかの脆弱性があり、1)デバイスへの物理的にアクセス　2)LAN への直接アクセス　3)インターネット経由　で悪用される恐れがあるという。

家庭で使われる他の製品も同様に侵害に遭っている。電子クリーナーなどの子供向けハイテク教育玩具メーカーである VTech は、1200 万人の個人データが流出するセキュリティ侵害が 2015 年 12 月に発生したと発表した。この事案での興味深い発見は、デバイスそのものの侵害はなかったが、デバイスが接続されているオンラインサービスが十分セキュアでなかった点だ。具体的には、レポートによれば、SQL インジェクションに対する脆弱性を突かれた点と、登録サービスに TLS が使われていなかったことが挙げられる。これらの弱点は IoT デバイスそのものの欠陥に関するものでなく、デバイスをサポートするインフラの欠陥であった点である。IoT デバイスはそれ単独で機能するわけではなく、もっとずっと大きなシステムの一部であり、そのシステムも同様に十分にセキュアでなければならないのである。
ウェアラブルデバイスも同様にプライバシーに関する心配がある。一例として、Open Effect が行った調査によれば、多くのウェアラブル製品のメーカーは、Bluetooth4.2 の仕様に取り入れられた新しいプライバシー保護の機能を取り入れていないということである。製品開発に際して、デバイスに、だれでも解析可能なスタティック MAC アドレスを利用している場合、ウェアラブルデバイスを装着した人を追跡することが可能であるという。

1. ユーザー登録には、すべて TLS(Transport Layer Security)による暗号化を用いること
2. 開発チームにはソフトウェア安全確保技術を身に付けさせること。
3. プロトコル仕様のセキュリティ・プライバシー関連のアップデートはとことんチェックしておくこと。

IoT 製品のコンピューティング力が DDoS 攻撃に使われる

IoT 製品を大量に使うことを検討するとすれば、DDoS(Distributed Denial of Service)攻撃を企てている連中にとって、それがどれだけ有用かを考えるべきである。そのような攻撃はすでに起きている。Sucuri 社の研究者は、2 万 5 千台以上の CCTV（監視カメラ）が乗っ取られ、DDoS 攻撃に使われたと報告している。市販されている一部メーカーによる製品の RCE(Remote Code Execution)の欠点が悪用されたと指摘している。この事例は、IoT 市場に参入するベンダーは、ソフトウェアのセキュリティを保証するための技術を身に付けるべきであることを改めて示している。

IoT 製品を悪用してボットネットに用いることは、パッチの当たっていない脆弱性を見つけ出してそれを悪用する手間ほど複雑ではない。多くの IoT 製品は、出荷時にパスワード保護をかけないか、ローカルアクセス用に admin などのデフォルトパスワードを使っている。攻撃者は、このような“下のほうに実っている果物”を見つけ出して、多くの製品を簡単に手に入れ、不正な目的のために活用できてしまう。この実際の例として、Lizardstressor の DDoS ボットネットがある。セキュリティ会社 Arbor Networks は、このボットネットの運営者は、同種のデバイスに同じデフォルトパスワードを設定している IoT デバイスをターゲットにしていると指摘している。

1. 開発チームにはソフトウェアのセキュリティを保証するための技術を身に付けさせること。
2. IoT 製品をパスワード保護なしで出荷しないこと。
3. 同種デバイスすべてに同一の初期パスワードを適用しないこと。あるいは初回利用時に直ちにパスワード更新させること。

医療用装置と医療用標準プロトコルの攻撃に対する脆弱性

ネットに接続される医療用装置のセキュリティに関する懸念は今に始まったことではなく、IoT が一般的になる以前から知られていた。2008 年に、Daniel Halperin とその共同研究者は、「ペースメーカーと埋め込み式除細動器 : 無線によるソフトウェア攻撃とゼロパワー防御」と題する論文を発表した。この論文は無線によりプログラム更新可能な埋め込み型医療装置(IMD)－ペースメーカー、除細動器、埋め込み型薬剤注入装置－について検討したものである。この研究は、これら装置にかかわるプライバシー問題だけでなく、装置の設定変更、治療を狂わせたり停止させること、さらには患者にショックを与えることが可能であることについて指摘している。詳しくはこちらを参照のこと。

さらに、研究者である Jeremy Richards は、薬剤注入用ポンプの一連の製品に数多くの欠陥があることを明らかにした。以下の例が指摘されている。

- telnet セッションでの認証がない。
- 病院内 Wi-Fi ネットワーク用の WPA(Wi-Fi Protected Access)の認証キーがデバイス上に平文で格納されている。
- デバイス上の Web サーバーは脆弱なバージョンを使っていた。（つまり、Web サーバコンポーネントはパッチ適用されていなかった。）
- FTP(File Transfer Protocol)の認証情報がハードコードされていた。

ペースメーカー、病院内薬剤注入用ポンプその他の医療装置の脆弱性は極めて危険で、命を脅かすよ

うな傷害や、死をももたらす恐れがある。研究者はこれらの脆弱性を実演するのに iStan（訳注：人体シミュレータ）を利用した。一例として、FDA（連邦政府医薬食品局）は医療機関に対してある種の注入システムの利用に伴うリスクについて警告を発している。

1. 開発チームにはソフトウェアのセキュリティを保証するための技術を身に付けさせること。
2. すべてのポートへのアクセスに認証を適用すること。
3. デバイス上に格納する認証キーはすべて暗号化すること。
4. 装置管理者がソフトウェアの管理を簡単にできるような仕組みを提供すること。（つまりデバイス上の Web サーバーにパッチをあてること。）
5. 同種デバイスすべてに同一の初期パスワードを適用しないこと。あるいは初回利用時に直ちにパスワード更新させること。

ドローンが偵察活動のプラットフォームとして利用され出している

小型無人飛行システム（sUAS: small Unmanned Aerial Systems）、別名ドローンが普及し、セキュリティの研究者は、このシステムを偵察活動にどのように利用可能か調査し始めている。その一つの用途として、一定範囲の地域で無線プロトコルを使う IoT 製品を識別することがある。Praetorian 社の研究者チームは、2015 年に ZigBee のビーコンのマッピングに活用することで、ドローンがこの用途に有効なツールであることを示した。

1. 使用する製品で使われる IoT 通信プロトコルを注意深く調べ、外部から収集できる情報の量を最小限にするする方式（モード）に設定すること。

国家重要インフラへの IoT エコシステムの活用

相互に接続されたスマートシティあるいはスマート工場の観点からすると、価値のあるデバイスが接続されたスマートグリッドや、信号機あるいはスマート工場のスマートネットワークが、現代社会の機能に直接的に影響しうる巨大な相互接続に依存していることがわかる。もし典型的な産業制御システム(ICS)のことを考えるならば、多くの領域で危惧があることもわかる。これらの（危惧の）領域には、もともとインターネットに接続するよう設計されていないシステムの（インターネットへ

の）接続や、セキュリティ機構が組み込まれていない古いプロトコルの利用、さらにはこれらのサイバー・フィジカルシステム(CPS)がもし侵害された場合、（物理的な）破損や障害を引き起こしうるという事実を含む。

1. サイバーフィジカルシステム(CPS)の中で古いプロトコルをアップグレードし、もっとセキュアなプロトコルを使用するように改良すること。
2. IoT/CPS 製品の設計には、安全工学を適用すること。
3. IoT 製品には、セキュアな接続インターフェースを実装すること。

車はネット接続され自動運転化する

コネクテッドカー（CV: Connected Vehicle））技術は、交通事故や交通事故によるケガを減らすため、様々な形で利用されるだろう。コネクテッドカーの通信は DSRC(Dedicated Short Range Communications)プロトコルを用い、車対車(V2V)、車対インフラ(V2I)、車対アプリ(V2X)間で高速のメッセージ通信（BSMs: Basic Safety Messages）を行う。CV エコシステムは、何種類もの技術やハードウェア・ソフトウェアコンポーネントで構成される。インフラ装置、例えば交通信号機は 1)制限速度　2)信号の状態と変わるタイミング　3)前方の交通状態の情報　などのメッセージを配信する。歩行者も、スマホ上のアプリや専用のドングルを使って、車や交通インフラ装置と通信するだろう。

CV エコシステム全体のセキュリティには、相互接続点のセキュリティを定義するだけでなく、まずもってセキュアな製品を実装することが必要になる。そういった製品には、車それ自体、道路側の各種装置、さらにはまだ市場に登場していないサードパーティーによるドングルが含まれる。
我々は既に、交通制御プラットフォームや、車それ自体に対する侵害を起こすような研究結果を目にしている。

1. 開発チームにはソフトウェアのセキュリティを保証するための技術を身に付けさせること。
2. 同種デバイスすべてに同一の初期パスワードを適用しないこと。あるいは初回利用時に直ちにパスワード更新させること。
3. IoT 製品には、セキュアな接続インターフェースを実装すること。
4. IoT/CPS 製品の設計には、安全工学を適用すること。
5. IoT 製品には、セキュアな接続インターフェースを実装すること。

今後に向けて

スマート工場や相互接続された製造設備について語るまでもなく、センサーやロボットは設定された目的を達成すべく、相互に連携して仕事をし始めている。ブロックチェーンに関する新たな研究を見ると、その能力を機械間の協調動作に利用できるよう拡張することが約束されているようである。こうした能力は、人間を意思決定の過程から追い出してしまうだろう。人間が、IoT 製品の基本的な判断に頼るようになる時、我々はこれらの製品、そのサービス、そしてそれらの相互接続点が、可能な限りセキュアなものとして開発されることを再確認しなければいけなくなるだろう。

IoT 製品のセキュリティに注意しなければならない理由

500 億台を超える機器が 2020 年までにネットワークに接続されるという予想もあるように、IoT 製品は様々なところで導入され、我々の家庭や仕事場や車、そして飛行機にまで入り込んでくるだろう。これらの機器に我々の既存のネットワーク基盤との間の相互接続性を与えることで、多くの攻撃者が狙うであろう新たな攻撃経路を開くことになる。セキュリティ研究者たちは、今日、既にある多くの IoT 機器に関連する脆弱性を特定しようと頑張っているが、すべての脆弱性が特定され、悪意を持った者たちに利用される前に修正されるわけではない。特定の IoT 製品が利用者の重要な情報を侵害するためや、さらには危害や損害を与えるために使用された場合、製品ベンダーにとっては、存立に関わる事態となるだろう。

大規模な攻撃に製品が利用された場合の深刻な結果を知りつつも、セキュリティ専門家は、しばしば製品やシステムをセキュアにするための投資を拡大するためにビジネス的な理由を考えるという気の滅入る仕事に直面する。そうしたことから、我々は開発者が製品のセキュリティに注意すべき、いくつかの理由を列挙することにした。資料［２４］によれば、製品のライフサイクル、つまり初期設計から運用環境までを通じて、セキュアなソフトウェア起動（ブート）、アクセス制御、デバイス認証、ファイアウォールや IPS の適用、アップデートやパッチの適用までを含むセキュリティを組み込まなければならない。IoT に関するセキュアな製品のエンジニアリング、すなわち

1. IoT 製品が偽造される可能性の低減
2. 攻撃者が顧客のプライバシーを侵害する能力を制限すること
3. デバイスが侵害された際の責任の制限
4. サイバーフィジカルシステム（CPS）の場合は、攻撃者が（物理的な）傷害や損傷を与える能力を制限すること
5. 製品への悪評が立つ可能性を低減すること
6. 潜在的なリスクを低減するとともに、製品にセキュリティを組み込むことに制約がある場合は、必要な代替策を講じること

などを目的に、十分な予算を配分すること。

IoT 製品が侵害を受けた場合、その結果の影響は決してプラスには働かない。少なくとも、顧客に、その製品が顧客のシステムや家庭を（攻撃に対して）脆弱にするといった考えに基づき、製品の購入をためらわせるだろう。最悪の場合、侵害されたシステムが原因で（訴訟や訴追などの）法的なアクションを受けたり、（利用者や環境に対して）物理的な損傷や傷害をあたえることになる。

米国連邦取引委員会（FTC）も **IoT ガイドライン**を制定しており、開発者に対してセキュリティを後付けではなく、アーキテクチャの一部として組み込むように推奨している。コンシューマ IoT 機器の脆弱性の原因となる他の要素としては、設計の悪さや、統制されていない、もしくは統制が不十分な環境（での利用）、サードパーティー製のハードウェア、ソフトウェアやセキュアでないクラウド上のリソースの利用などが複合した要因が考えられる。（IoT システムには）ここまでに述べたような多様性や相互依存性が多く存在することから、こうした依存性が増加すれば、責任（非難の矛先）をこれら（他の）サービスに押しつけることも容易になってしまうだろう。（多くの場合、複数の箇所に責任が存在する）

IoT 機器におけるセキュリティの課題

IoT 製品の開発をセキュアに行う方法を理解するためには、セキュリティに関するアプローチを検討するに際して、まずセキュリティ技術者（専門家）が直面している課題について理解することが重要である。そこで、以下にそれらの課題を列挙しておく。

表 1

深刻度	課題
高	IoT 製品が、セキュアでない、もしくは物理的に保護されていない環境に導入される可能性がある
高	多くの製造事業者にとって、「セキュリティ」は新たな領域であり、開発の方法論において、セキュリティに関する計画が不十分である
高	セキュリティはビジネスの推進力と考えられておらず、IoT 製品の開発に際して、セキュリティへの投資や経営層によるサポートが限られてしまう
高	セキュアな IoT 製品開発についての標準や参照アーキテクチャが欠如している
高	IoT 開発チームに必要な、アーキテクト、セキュア開発の技術者、ハードウェアセキュリティ技術者、セキュリティ検査要員などについて要求されるスキルを満たすような人材の雇用が難しい
中	低価格化戦略によって、潜在的な脅威（敵）が増加する
中	組み込みシステムにおける能力の限界から、セキュリティに関する選択肢が限定されてしまう

IoT 製品が、セキュアでない環境に導入される可能性がある

IoT 製品が物理的に保護されていない環境に置かれた場合、それが盗まれてリバースエンジニアリングされ、（共有鍵やパスワードといった）秘密要素や攻撃可能な脆弱性を特定され、後刻、実際に運用されている製品の攻撃に利用される可能性がある。コンシューマ向けの IoT 機器は、しばしば（家庭のような）一般にソフトウェアに存在する脆弱性への攻撃に対する防御が十分でない、もしくはまったくない環境で利用される。こうした環境では、階層的な防御という考え方（Defence-In-Depth）は、ほとんど成立せず、開発者は以下に述べるようなネットワークでの運用を想定しなくてはならない。

- WiFi ルータの認証メカニズムが脆弱な状態で利用されている可能性がある　（訳注：たとえば、デフォルトパスワードを変更せずに利用しているなど）
- WiFi ルータが旧式でパッチも当てられていないまま公開されてしまっている可能性がある

- （ネットワークに繋がっている）ホーム PC で使われている OS が古く、ウイルス対策もなく、パッチもあてられていない可能性がある
- ネットワークに接続されたタブレットやスマートフォンで（サポート切れした）古いファームウェアが利用されている可能性
- ゲーム機がネットワークに接続されている可能性

家庭にあるセキュアでない WiFi ルータ、スマートフォンアプリケーション、さらにはセキュアでない末端の IoT 製品への攻撃により、ネットワーク接続や、場合によっては信頼関係を確認するための（暗号）鍵などが漏洩する可能性がある。

1. ファームウェアやソフトウェアの重要な更新を強制するようなポリシーを実装すること
2. IoT 製品のための、柔軟でセルフサービスが可能な ID マネジメントの仕組みを見いだすこと
3. モバイルアプリケーションの内部に置かれた IoT 機器との信頼関係を確立するための本人確認情報や暗号鍵などを暗号化しておくこと

多くの製造事業者にとって、「セキュリティ」は新たな領域である

伝統的な IT セキュリティは、コンピュータ業界が多くの年月をかけて育んできたものである。これまでセキュリティ上の心配事に向き合う必要がなかった製品開発者にセキュアな形で開発、エンジニアリングを行うスキルがしばしば欠落していることは理解できる。ソフトウェアや（ネットワークへの）接続性を製品の機能として付加しようとする製品開発者の多くが、コネクテッドデバイスに内包されるセキュリティ上の弱点や（それらを排除するための）ソフトウェア保証の基本についてのスキルセットを構築しつつある。しかしながら、依然としてギャップはあり、いくつかの製品は、非常に長いライフサイクルを持っていることから、セキュアでない製品が継続的に市場に流入する原因となっている。セキュリティの評価と認証は IoT 製品の開発企業に対し、競争相手に対する競争優位性に加え、様々な環境に製品を導入できる可能性を高める効果をもたらす。規制、プライバシー、およびコンプライアンスにおける義務として、特定のシステムに追加されたデバイスを、デプロイする前に、特に、機微な情報（PII：個人特定可能情報など）を扱うときにテストして検証することが要求される。安全に製品を開発するために必要なセキュリティ専門知識が不足している開発企業にとっては、有償または無償でセキュリティ評価サービスを提供する多くの第三者機関や企業が存在する。

1. 開発チームのための IoT セキュリティトレーニングプログラムを作ること
2. (ISAC のような）脅威情報共有のためのコミュニティを見つけて参加し、製品の脅威モデリングを行うためのフレームワークを確立すること
3. 製品にセキュリティを組み込む必要性について、あらかじめ経営層の支持をとりつけておくこと

製品設計、開発、テスト、統合に使用されるエンジニアリングプロセスにセキュリティを組み込む必要があるということについては、最初は自明ではないように思えるだろう。多くの開発者は IoT 製品を開発する際に、アジャイル（開発の）原則に従っているが、そこには、しばしば埋めるべきギャップが存在する。たとえば、「セキュリティ要件が含まれるプロセスが事前に十分に識別されることを保証する」というような（セキュリティ上の）要求事項はプロセス全体を通じて追跡され、製品のテスト中にそれらが満たされていることを確認しなければならない。IoT の製品開発者は、「Secure by design:設計によるセキュリティ」と「Privacy by design:設計によるプライバシー」という概念を（自分たちの開発手法に合わせた形で）考慮する必要がある。

コンシューマまたはエンタープライズ IoT 製品を開発する組織は、コードの相互レビューやコード分析を含む開発セキュリティのベストプラクティス（たとえば Microsoft SDL、BSIMM、OpenSAMM など）を取り入れるべきである。また、典型的な利用環境下での最終製品に対するペネトレーションテストも実施しなければならない。ソフトウェア開発のエンジニアリングの取り組みはずっと行われてきたが、開発、運用、セキュリティを統合するという課題は、各チームの固有のエンジニア文化のために厄介な課題だった。（しかし、現在）DevOps と DevSecOps のコンセプトは、エンジニアリング手法の欠点に対処するために進化し続けている。（また）組織内でのペネトレーションテストにより、ソフトウェアの欠陥が最小限に抑えられる。

1. すべての開発ステージでセキュリティを組み込むために開発プロセスを見直して更新する
2. すべての IoT 製品の開発に「Privacy by design: 設計によるプライバシー」の原則を組み込む

IoT 製品の開発に際して、セキュリティへの投資や経営層によるサポートが限られてしまう

CSA は、IoT 開発のセキュリティに関する動機をよりよく理解するために、2015 年に技術ベンチャーに対する調査を実施した。調査の結果、投資家とテクノロジーの新興企業は自社製品のセキュリティに関心がないことが明らかとなった。一方、彼らは製品をすぐに市場に投入し、主要な機能が期待通りに機能するようにすることに重点を置いている。

情報セキュリティの観点から物事をより困難にしているのが、デバイス開発者は使いやすさとセキュリティのトレードオフを考慮する必要があることだ。これらの IoT デバイスの多くは、他のデバイス（例えば、スマートフォン）の機能に依存するため、アーキテクチャレベルでセキュリティを計画する必要性はこれまで以上に高まっている。いくつかのケースでは、コンシューマ向け IoT デバイスの製造元が、家庭の製品所有者がこれらのデバイスをより簡単に設定できるようにするために、セキュリティのベストプラクティス実装をあきらめる意識的な決定を行っている。

確かに、セキュリティはしばしば消費者の不便や複雑さを増大させると見なされ、サポートコストを上昇させ、ユーザーの利便性を低下させると考えられている。パスワード管理は、セキュリティとユーザビリティの間で起こりうるトレードオフの一例だ。単純なパスワードを長く使うことが利用者にとって楽な反面、攻撃者にとっても推測しやすいものになってしまう。ここでの課題は、可能な限りセキュリティを強化し、同時にユーザーエクスペリエンスを向上させるための将来の技術ソリューション開発である。

IoT 製品の開発企業はセキュリティをビジネス上の要求事項の一つとしてとらえるべきである。まず、開発中の各製品を調べて、製品に対する固有の脅威を理解し、それから、それらの脅威が現実化する可能性を軽減するために生成されたセキュリティ要件のバックログを把握する必要がある。

1. 各製品の開発の初期段階で脅威モデリングを実施すること
2. 脅威モデルからセキュリティ要求事項を導出し、それが完結するまで追跡すること

セキュアな IoT 製品開発についての標準や参照アーキテクチャが欠如している

IoT をサポートするために実装された技術は、よりスマートで完全に接続されたエコシステムのニーズに合うよう、（プラットフォーム、通信、ルーティングプロトコル、セキュリティ機能など）数多くの分野にまたがって成長している。参考情報［２２］では情報、物理的および管理モデルという 3 つの基本的なセキュリティの視点からなる IPM と呼ばれる体系的なセキュリティアーキテクチャを提案している。PubNub（参考情報［２３］）は、セキュアなグローバルデータストリームネットワークであり、ユーザーが IoT デバイスとリアルタイムアプリケーションを接続、拡張、管理できる使いやすい API のもう 1 つの良い例である。そのほかにも、産業界、学術界の両方で、IoT 環境の標準化に向けて動き始めている。実際、この目的のために、多くの標準案が提案されている。インターネット・エンジニアリング・タスクフォース（IETF）は現在、資源制約のあるデバイスの通信プロトコルに関して標準化プロセスをリードしており、同時に、低電力および低ロス・ネットワーク用のルーティング・プロトコル（RPL）および制約付きアプリケーション用プロトコル（CoAP）を含むいくつかのインターネット・プロトコルを開発中である。他にも多くの通信およびメッセージングプロトコル標準が、認証および認可標準と共に提案されている。こうした状況下で、どの方式を選択し、どのようにそれらをうまく独自の製品開発に適用していくかを理解するのは簡単ではない。

標準化の観点からの他の主な懸念事項の 1 つは、ベンダー間で受け入れられた参照アーキテクチャが存在しないことである。IoT 製品とサービスでは、多くの技術とプロトコルの連携が求められるため、このことは重要である。実際、IoT-A と呼ばれる参照アーキテクチャが欧州のプロジェクトで提案されているが、それはまだ標準として受け入れられていない。この問題を軽減するために、多くの IoT 製品開発者は、特定の IoT プラットフォームを出発点として選択し、そこから独自のカスタマイズとサービスを構築している。しかし、これらのプラットフォームは、しばしば互いに相互運用性を持たないものである。さらに、開発者と設計者は独立して行動し、しばしばセキュアな選択をしない。また、制限された IoT デバイスからセキュリティログファイルをオフロードするための合意された方法の欠如など、標準自体にもギャップがある。

1. デバイスが導入される環境を慎重に評価し、必要なセキュリティ・レベルに応じて技術を選択する
2. パフォーマンスとセキュリティのトレードオフを評価し、最適なプロトコルスタックを活用してセキュリティリスクと侵害を減らす
3. IoT コンポーネント（TPM ハードウェアなど）によって提供されるセキュリティ機能を評価し、可能な限り使用する

IoT 開発チームに必要なスキルを満たす人材の雇用が難しい

IT セキュリティスタッフは、新たな技術や製品について学ぶ必要性を常にかかえているが、これらのスタッフに対する十分な教育を行うことは、IoT によってさらに難しい課題となってくる。製品セキュリティ担当者とそのチームは、（過去には一般的に使用されていなかった新しい言語を含む）ソフトウェア内の脆弱性、攻撃者が製品のハードウェア機能を侵害する可能性のある方法、そして、当然ながら、ファームウェアやソフトウェアのアップデートを作成して数百万とは言わないまでも、数千ものデバイスに配布するセキュアなメカニズムについて、気を配る必要に迫られている。

新しいトレーニングでは、セキュリティチームに以下を理解させる必要がある。

1. IoT に関連する新技術
2. IoT の脆弱性に関連する新しい脅威プロファイル
3. （潜在的に何百万ものデバイスにまたがる）IoT システム侵害の影響

1. 開発チーム向けの IoT セキュリティトレーニングプログラムを作ること

低価格化戦略によって潜在的な脅威（敵）が増加する

他の部分でも説明したように、典型的な IoT 製品、特に民生機器の低価格化により、研究者や悪意のある者たちは、（デバイスを）入手し、セキュリティ問題の発見と各デバイスに組み込まれたセキュリティ保護の分析に時間を費やすことが簡単になる。これにより、ハードウェアとソフトウェアの両方に関連するセキュリティ上の脆弱性を体系的に発見することができ、その知識を使用して運用環境の弱点を利用することができる。

攻撃者が IoT デバイスを見つけてリバースエンジニアリングできる状況では、共有された対称鍵、秘密鍵、その他の暗号プリミティブなどの重要なデータが侵害される可能性がある。共有対称鍵に依存する IoT ネットワークの場合、攻撃者は同じ鍵を共有する他のデバイス内に保持されているすべてのデータにアクセスできる可能性が高いため、これは特に重要である。さらにデフォルトのパスワードがデバイスで使用されていることを知ることで、ネットワーク内の類似デバイスを簡単に侵害できる可能性もある。

1. デバイス内部の機密（ハード、ソフト）への物理的なアクセスを防ぐための変更や改造の検出(tamper detection）などの物理的な安全対策を検討する
2. パスワードを使用して製品の物理ポート（テストポートを含む）をロックダウンする

組み込みシステムにおける能力の限界により、セキュリティに関する選択肢が限定されてしまう

数年前には、IoT 製品が十分なセキュリティメカニズム（例えば、サイズ、重量、消費電力）を提供することができるかという大きな懸念があったが、今日、それらの懸念は以前ほど大きくはない。依然として暗号化ハードウェアモジュールを実装することができないいくつかの高度に制約されたデバイスがあるが、ソフトウェアベースの暗号化モジュールの作成やセキュリティコプロセッサのようなものを MCU に追加することなどについての進歩が重荷を軽減している。

しかし、今のところ、すべての IoT デバイスが堅牢なセキュリティメカニズムをサポートしているわけではない。多くのデバイスでは、サイズ、重量、消費電力、メモリー/処理能力の間でトレードオフになる。技術の進歩によって、これらの制約は軽減されていくだろう。今日、IoT 開発者として、自分の IoT 製品の基礎となる技術の選択には慎重を期してほしい。

1. 機微な情報補保護するため、可能な限りハードウェアベースのセキュリティ対策を施すこと

IoT セキュリティに関する調査

CSA IoT ワーキンググループは、IoT ソリューションを開発する新興企業に対する調査を実施した。以下に示す結果及びその分析は、本書のようなセキュア開発のための文書の必要性を示している。この調査において企業は、安全な開発方法論と実践に関連する回答を提供するよう求められた。調査からわかった主な知見には、以下のようなものが含まれる。

1. 新興企業は機器に格納された情報を機微なものとは考えていない。（機微な情報はサーバー側にあると考えている）
2. 利用者は情報の共有を望んでいる（と考えている）。（情報共有文化）
3. 新興企業はサードパーティーが提供する出来合いの（Commercial Off-the-Shelf）サービスに大きく依存している。
4. ほとんどの新興企業は暗号化に AES を使用しているが、暗号化そのものを重要とは考えていない
5. ほとんどのデバイスは、デバイス間やサーバー側の管理者との間でマスター暗号鍵を共有していない
6. 開発環境に、まったくセキュリティが適用されていない
7. 製品に関する脅威モデリングを行っていない
8. セキュアなファームウェア更新機能が存在しない
9. 投資家はセキュリティを気にしておらず、機能面にばかり着目しているように見える

我々の調査結果に基づけば、多くの IoT 開発者がセキュリティを重要と考えていないことが容易に理解できる。IoT ソリューションを迅速に市場に投入する必要があることを考えると、これは必ずしも驚くべきことではないが、今後数年間に市場に出る新しい機器のありかたに関わる懸念を引き起こす。

調査結果のもう 1 つの興味深い側面は、（情報）共有の考え方に重きが置かれているということである。つまり、情報は共有されるべきであり、情報の共有を可能にすることがセキュリティの確保より重要だという考え方である。これは多くの商用（コンシューマ向け）IoT デバイスでは当てはまるが、情報の保護と共有に同等の重さを置く企業にとっては好ましくない。加えて、特に懸念されるのが、従業員や委託業者が許可を得て、または許可なく企業の環境に自分のデバイスを持ち込む際に、これらのコンシューマ向けデバイスを企業内に持ち込んでしまう可能性があることである。

セキュアな IoT 開発のためのガイダンス

この章では、合理的な程度にセキュアな IoT デバイスの設計と開発のための考察とガイダンスを提供する。なお、このガイダンスは、基本的なシステムセキュリティエンジニアリングの方法論と技法を理解することに換えるためのものではなく、IoT デバイス開発で見つかったより一般的な問題のいくつかを軽減することを目的としている点に注意して欲しい。ここでは開発組織が IoT 製品のセキュリティ状態を改善し始めることを可能にする、いくつかの活動について、アウトラインを示す。図２はよりセキュアな IoT 機器を開発するためのステップについて示している。

1. セキュアな開発手法から始めよう

現在、多くのソフトウェア開発チームは、要求事項やアーキテクチャ、設計やコーディングにおいてのセキュリティについて、十分な関心を持っていると言えない状況にある。そうであれば、既存の製品を「接続する」ことに関心を持つ製品会社が、安全にそれを行う方法の専門知識を持っているかどうかという点について、多くの疑問が生じる。この課題に取り組むためには、セキュアなエンジニアリングアプローチ、つまりセキュアな開発の方法論に取り組む必要がある。

理想的には、組織のセキュアな開発の方法論は、単に技術的なチェック以上の必要性を喚起するだろう。ドキュメンテーション、ピアレビュー、セキュリティ要件の製品ライフサイクルへの組み込みなども同様に、プロセスとして組み込む必要がある。エンジニアリングプロセスには、製品をより良く、より安全にするための、きちんとしたフィードバックループも含まれている必要がある。これらのフィードバックループの例には、コードの脆弱性の発見時に製品の改善項目を追加することや統合テストの中で問題が発見された際の製品設計手法の更新などが含まれる。（e.g 継続的な統合テストなど）

脅威モデリングの実施

脅威モデリングは、セキュア開発手法の主要なコンポーネントのひとつである。IoT 技術と製品は常に変化している。従って、脅威と（それがもたらす）問題の組についてのリファレンスを確保することが、それらに適切に対処するために重要となる。そのようなリファレンスの良い例のひとつが、IoT システムによって提起されるセキュリティ領域のトップ１０を定義した OWASP（Open Web Applivcation Security Project）である。OWASP は、脅威の実行者、攻撃経路、脆弱性やそれらに関連した影響というような切り口から情報を提供している。これは、製造者、開発者、および消費者に適用され、IoT に関連するセキュリティ問題をよりよく理解し、どのような状況においても IoT 技術に関連するより良いセキュリティ判断（構築、導入、または評価）を可能にする。以下は、対象となる大まかな分野である。

- IoT 機器
- クラウド
- モバイルアプリケーション
- ネットワークインターフェイス
- ソフトウェア
- 暗号の利用

- 認証の利用
- 物理的なセキュリティ
- USB ポート

セキュアな開発プロセスの一環として、ソフトウェアまたはハードウェア上で脅威を識別するために脅威モデリングを実施し、特定された脅威を軽減するために適切な対策を導入する必要がある。一般的なソフトウェアの脅威モデリング手法は、IoT にも適用できるだろう。

- 脅威モデルへのリファレンス.
  マイクロソフト　脅威モデリング
  OWASP アプリケーション脅威モデリング
  IEEE Cybersecurity Secure Design

脅威モデリングへのリファレンスは、Adam Shostack の書籍 "Threat Modeling: Designing for Security." にも掲載されている。マイクロソフトも、新しいシステムによって生じる脅威の深刻さを見極めるための複数のステップからなる、よく検討された脅威モデリング手法を定義している。マイクロソフトの SDL に基づく脅威モデリング手法は、IoT 機器に対する異なるタイプの脅威を考察するのにもよい方法である。

表 2 は、脅威のカテゴリ分類の一例である。

表 2

脅威のタイプ	Discussion：脅威対応のポイント
ID の詐称	マシンの ID のなりすましと製品、機器やサービス間の自動化された信頼関係を破るための攻撃者の能力に関連する脅威について、システムを調査する。さまざまなデバイス（M2M）間、およびこれらのデバイスによって提供されるデータを利用するデバイスとアプリケーション間の安全な通信を確立するために採用されている認証プロトコルを注意深く調査する。ID（デバイス識別のための情報）を各 IoT 機器にプロビジョニングするプロセス（ブートストラッププロセス）を調査する。同シリーズの機器全体で同じ（サービス接続用）ID を共用しないこと。
データの不正操作、改ざん	IoT 製品で保持されているファイルについて、設定されていなければならない制限（r,w,x :読み出し、書き込み、実行パーミッション）を特定する。機器と対になっているスマートフォンアプリケーションについて、機能を実行するために必要不可欠なデータに限定してアクセス許可が与えられていることを確認すること。

（行為や処理を行った事実、データの正当性などの）否認	使用できる状態になる前に、ヘルスチェックを行ってデバイスのセキュリティ機能を確認する。IoT 製品によって生成されるデータに認証および完全性の保護を組み込む。設計段階からメッセージの再使用を制限する方法（例えば、シーケンス番号/タイムスタンプ）を組み込む。クラウド上のデータストアや同様にモバイルアプリケーションのデータストアに対して認証や完全性検証のための対策を実施する。
情報の暴露	API を経由した他の IoT 製品に対するデータ開示を必要なもののみに制限する。IoT 製品が生成するデータへのアクセスを制限するため、スマートフォンアプリケーションやクラウドサービスにおいて暗号化が適用されていることを検証する。機微と考えられるデータを格納、保管する IoT 製品すべてにストレージ上での暗号化（data-at-rest encryption）を適用し、可能であればこれらの操作に関連する暗号鍵はハードウエアセキュリティコンテナ（訳注：TPM/HSM など）に保管すること。機微な情報を保存または処理する IoT 機器には暗号鍵の消去機能を実装すること。他の機器、スマートフォンアプリケーションやクラウドサービスなどとの間のすべての通信を、暗号プロトコル（TLS/DTLS など）を使用して暗号化すること。
サービス妨害	機器が直接クラウドと通信するような場合、IoT 製品向けのクラウドサービスについて、サービス妨害攻撃からの保護を行うこと。必要に応じて、API に通信量（頻度）制限を実装すること。IoT デバイスが乗っ取られてボットネットに組み込まれることを防ぐために、顧客に対してソフトウェアコンポーネントを最新に保ち、特定された脆弱性について迅速にパッチを適用できるような手段を提供すること。セキュリティコミュニティから新しい脆弱性についての情報を積極的に受け取り、研究者と協力して迅速に脆弱性を解消するとともに、パッチを提供可能にすること。
権限の昇格	すべてのユーザー及びサービスのロール（役割）に対し、最小権限付与の原則に則った設計を行うこと。とりわけ、他のパートナーの IoT 製品と組み合わせる場合は、スマートフォンや IoT 製品に関連する権限を必要なものだけに限定すること。
物理セキュリティの回避	製品の利用環境についての脅威を考慮すること。場合によっては、物理的に堅固なセキュリティが必要とされる。少なくとも、（ハードウェアの）デバッグポートはパスワードで保護するか、可能ならば無効化しておくべきである。（物理的な）不正操作の検知やそれに対する対応のための機能などは、重要インフラなどに使用される IoT 機器において導入しておくことが推奨される。

附録 A では、IoT 機器のカテゴリとそれらに対する典型的な脅威についての詳細な調査について記載している。IoT 機器の設計を行う際のリファレンスとして使うとよいだろう。

セキュリティ要件

製品のセキュリティ要件も検討して実装する必要がある。（米国の）政府プログラムでは、DISA セキュリティ技術実装ガイド（STIG）またはセキュリティ推奨ガイド（SRG）から抽出した要件を使用して、製品のセキュリティ要件が作成される。これにより、開発プロセスを通じた要件のトレーサビリティが確保され、少なくとも最小限のセキュリティ制御が機器内に実装されていることが保証される。

セキュアな開発手法は、単に開発のみに着目していればよいわけではない。開発者は、自らの製品にセキュアなプロセスを組み入れる責任も負うべきである。これが意味するところを理解するためには、セキュリティ企業 Cigital が提唱する BSIMM（成熟度モデルによるセキュリティの構築）を検討するべきだろう。

セキュリティプロセス

BSIMM によって、組織は、既存の開発におけるベストプラクティスと、他の多くの組織が採用しているベストプラクティスを比較することができる。BSIMM は、４つのドメインと１２のプラクティス（行うべき事項）から構成される。これらのプラクティスの IoT への適用については表 3 に述べられている。

表 3

BSIMM ドメイン	プラクティス	IoT への適用
ガバナンス	1. 戦略と測定 2. 準拠性とポリシー 3. トレーニング	• 業界基準に対する自組織のセキュリティコントロール（の状況）を測定するための戦略と計画を立てる • テストとコンサルティングを担うサードパーティー組織との関係を構築する • 機器が運用環境において従うべき規制や法令について明らかにし、準拠しているかどうかを追跡する • すべてのチームメンバーに対して、IoT 機器の安全を確保する役割や、機器に対する典型的な攻撃パターンなどについてのトレーニングを実施する

BSIMM ドメイン	プラクティス	IoT への適用
情報収集	4. 攻撃手法 5. セキュリティ仕様と設計 6. 標準と要求事項	• 最適なセキュリティ機能を決定するためのトレードオフの調査と実施（実装効果とコストの比較検討など） • 過去に類似の製品が侵害された際の方法と教訓の調査 • 攻撃のモデルやパターン、参照アーキテクチャなどの構築
SSDL（セキュアソフトウェア開発ライフサイクル）との接点	7. アーキテクチャ分析 8. コードレビュー 9. セキュリティ試験	• IoT 製品サービスの機密性、完全性、可用性をサポートする階層的なセキュリティ機能を、電子的なセキュリティと物理的なセキュリティの両方を必要に応じ組み合わせて設計すること • できる限り、IoT 機器への攻撃経路を削減、防御すること • 連携するサービスについて、大量の機器までセキュアにスケールするよう設計すること • セキュリティ試験を通じてソフトウェア及びハードウエア（の安全性）を評価すること
配備・導入	10. ペネトレーションテスト 11. ソフトウェア環境 12. 構成管理と脆弱性管理(CMVM)	• 実際の運用環境に近い適切な環境においてペネトレーションテストを実施し、潜在的な弱点をあぶり出すこと • （配備中の）IoT 機器に対して有効な構成管理方法を確立すること

（物理的な）安全性への影響評価の実施

IoT の特色の一つが、物理的な世界と電子的な世界の融合である。このことは、IoT 機器やサービスへの侵入により物理的な影響を生じる可能性を意味する。「コネクテッドカー」のような大きな IoT 機器は悪意の事象により潜在的に大きな影響を与える可能性があることは明白だ。たとえ、もっと小さな機器（たとえば、スマートホームのドア鍵など）であっても、侵害によって被害や損傷を与えることがある。開発者はこうした問題を考慮した上で、その設計プロセスの一部として安全影響評価を実施すべきである。

製品やサービスが侵害された場合の安全性への影響を検討する。製品への意図的な操作を仮定し、機器が完全に機能停止してしまった場合（たとえば、サービス妨害攻撃などの場合）に何か危害を与える可能性があるかどうかを考える。機器それ自体に安全上の危惧がなくても、その機器に依存する他の機器やサービスにその可能性はないだろうか。どうしたら機器の不具合による潜在的な危害を回避もしくは最小化できるだろうか。他の人たちが安全に関係する、もしくは危険であると考

える問題は何だろうか。類似の、または関連したシステム導入において、安全に関係するとされた問題または実際に危害を与えた問題はないだろうか、など、様々な考察が必要である。

2. 安全な開発環境とインテグレーション環境を実装する

ソフトウェアとハードウェアシステムを安全に開発する必要性は、新しいものではない。 どのようにしてこれを行うのかという課題に対し、既に取り組んでいる業界があり、私たちは IoT 製品を安全に開発するために、彼らの努力から学ばなければならない。 これの大きな例として、C および C++ のための自動車業界ソフトウェア信頼性協会（MISRA）による安全なコーディングの取り組みがあげられる。 MISRA は、ソフトウェア開発プロセス、トレーニング、コーディングスタイル、ツール選択、テスト方法、検証手順（参考リンク）など、安全性が重要なシステム向けに設計されている。MISRA の主な目標の 1 つは、サイバー・フィジカルシステム（CPS）カテゴリの製品を扱う際に潜在的に重大な問題になる可能性のある、予期せぬ事態を避けることである。しかしこれは、すべての IoT 製品開発者の目標となるべきものである。

これを念頭に置いて、IoT 製品で一般的に使用されているプログラミング言語を見直し、これらの言語ごとに詳細なセキュリティガイダンス（使用可能な場合）を紹介する。

プログラミング言語を評価する

今日よく使用される言語には、Node.js を伴った Javascript や、組み込みシステムで伝統的に使用されている言語（C）がある。 Java、Python およびその他のさまざまな言語も、同様に、すべて IoT プログラミング言語として、よく挙げられている。

私たちの話は安全な IoT デバイスの開発だけではないことを忘れないでいただきたい。 IoT 製品の開発者は、デバイスと連携するスマートフォンアプリケーションや、デバイスから情報を収集するクラウドサービスを開発する責任もしばしば負っている。 適切なプログラミング言語についてセキュリティガイダンスはこれらについても同様に念頭において検討される必要がある。

InformationWeek は、IoT 開発者に使用される上位のプログラミング言語の概要を発表している。 表 4 は、IoT 製品を開発するために選択される可能性のあるプログラミング言語の一覧である。 IoT 製品開発者は、しかるべき（言語の）セキュリティガイダンスに慣れておく必要がある。

表 4

言語	推奨されるセキュリティ・ガイダンス
C	・ Guidelines for the Use of the C Language in Critical Systems, ISBN 978-1-906400-10¬1 (paperback), ISBN 978-1-906400-11-8 (PDF), March 2013. Motor **Industry Software Reliability Association (MISRA)** ・ **SEI CERT C Coding Standard** （**JPCERT/CC 日本語版**） ・ **Motor Industry Software Reliability Association (MISRA)**
C#	・ Microsoft (**C# Programming Guide**) ・ OWASP **.NET Project** **.NET Security Cheat Sheet** ・ **CodeGuru**
C++	・ Guidelines for the Use of the C++ Language in Critical Systems, ISBN 978-906400-03¬3 (paperback), ISBN 978-906400-04-0 (PDF), June 2008. ・ **Motor Industry Software Reliability Association (MISRA)** ・ **SEI CERT C++ Coding Standard**
Embeded C++	・ **Dialect of the C++ programming language for embedded systems**（**Wikipedia**）
Erlang	・ **Rikitake, K and Nakao, K. Application Security of Erlang Concurrent System**. Network Security Incident Response Group, NICT, Japan.
Objective C	・ **Secure Coding Guide** （Apple）
Java	・ **Oracle Secure Coding Guidelines** ・ **The CERT Oracle Secure Coding Standard for Java** ・ **Android Secure Coding Guidelines**
JavaScript	・ **OWASP AJAX Security Cheat Sheet**
Python	・ **OWASP Python Security Project**
Rust,Go, ParaSail,B#	・ **Parallel Specification and Implementation Language ParaSail** ・ **Go**, **Rust**, **B#** (object-oriented, and multi-threaded programming language embedded systems)

統合された開発環境

開発者は通常、デバイスの動作に必要なセンサーだけでなく、製品の出発点として使用する MCU とフレームワークを特定する必要がある。利用するセキュリティサービスを MCU（の機能）から特定

することは、この選択における重要な切り口だが、MCU 自体にセキュリティ機能を実装する能力がない場合にはアドオンハードウェアコンポーネントを組み込むこともできる。 開発者が製品のファームウェア/ソフトウェア機能を作っている間に、これらのハードウェアコンポーネントはすべて、組み立てられ、テストされる。
場合によっては、堅牢なソフトウェア機能が必要である。 IoT 製品は、接続されたコンピュータまたはモバイルデバイス上のウェブブラウザを介したユーザーによる設定をサポートするため、軽量ウェブサーバを伴って構成されることがある。 また同様に、リアルタイムオペレーティングシステムを選択して様々な通信プロトコルをサポートすることもできる。 しばしば、外部で開発されたアプリケーションがデバイスに統合され、多くの場合、上流サービスやその他のデバイスへの接続に API が用いられる。

IoT 製品は独自のやりかたで開発することもできるが、多くの場合、既存の開発キットを活用することは理にかなっている。 現在、多くのキットが利用可能で、様々な種類のデバイス間での相互運用性機能を提供したり、クラウドへの接続性を備えた状態ですぐに使用できる。

これらの技術はすべて開発の間に統合され、ほとんどの場合、それらに加えて製品の差別化機能を提供するためのソフトウェアが作成される。図 1 からわかるように、これらの製品の開発、統合、テスト、および配備のための系統的なプロセスを採用する必要がある。 セキュリティは、開発サイクルのこれらの段階ごとに検討する必要がある。

ソフトウェア開発者はしばしば統合開発環境（IDE）を活用している。IDE そのものの機能により、いくらかのセキュリティサービスを提供することができる。活用できる IDE は多くあるが、Eclipse IDE に絞って調べてみると、OWASP ASIDE プロジェクト[1]のように、対話形式の静的コード解析（Java および PHP 対象）により、開発者自身がコードからソフトウェア脆弱性を検出し、軽減する助けになるものがある。別の例として、Contrastfor Eclipse プラグインは、以下の機能を提供している。

1. OWASP トップ 10 の脆弱性の自動検出
2. Eclipse IDE との透過的な統合
3. 実行中のアプリケーションの詳細なデータフロー分析
4. ソースコードにおける脆弱性のトレースバック
5. 状況に応じた専門家のセキュリティアドバイス。

また、オープンソースプロジェクトを活用して IoT ソリューションを構築するのに役立つ IOT Eclipse プロジェクトもある。

[1] 訳注：このプロジェクトは　2017 年 5 月時点で活動休止しています。

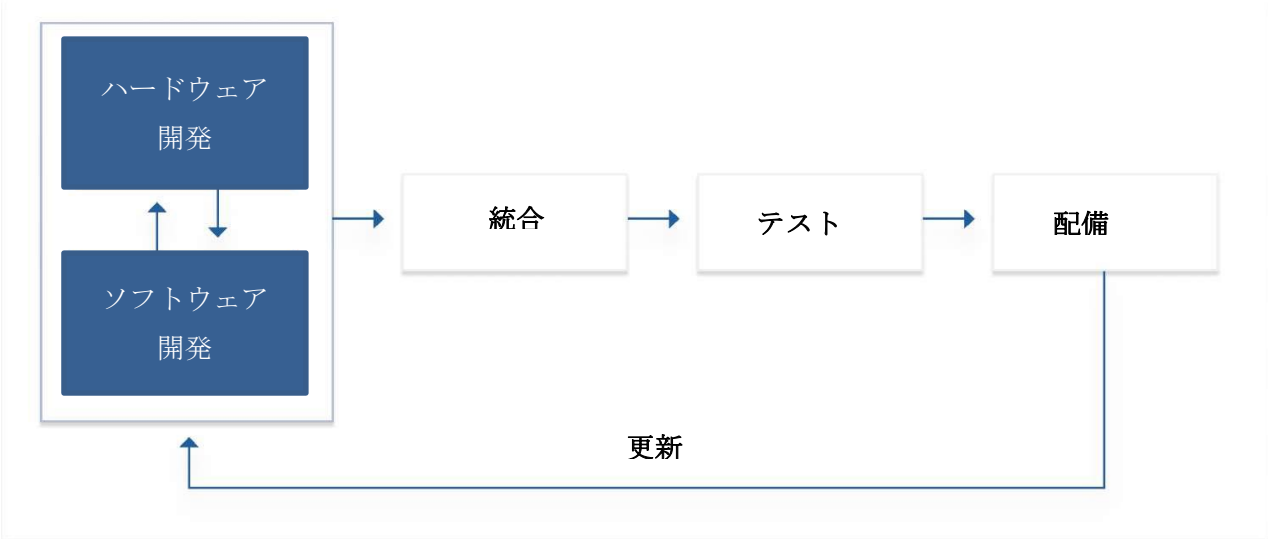


図 1

継続的インテグレーション(CI)のためのプラグイン

Jenkins などの CI サーバーは、OWASP ZAP（Jenkins ZAP Plugin）などのよく知られているプラグインとテストツールで企業のソフトウェア開発ライフサイクル（SDLC）に柔軟性を提供する。 ZAP では、ビルドステップが完了し、受入テストの準備が整うと、コードを動的に自動的にスキャンすることができる。ZAP の自動評価レポートをもとに、スキャンに合格した場合は配備を、スキャンが失敗した場合には、設計段階に戻すことができる。 一般的なスキャンでは、不正なファイルへのアクセス、古いアプリケーション（実行）基盤、さまざまなパラメータインジェクション攻撃、およびログインやログアウトフォームなどのような様々なセキュリティ機能をチェックする必要がある。 ZAP を使用する際には、多くのテストフレームワーク、スキャンポリシー、およびシーケンス構成が用意されており、ソフトウェア開発ライフサイクルへのより良い統合に使用できる。 セキュリティ上の脆弱性が確実に解消されていることを確認するため、常に手動テストで自動スキャンを補完することを推奨する。

ZAP だけでは、IoT 関連の脆弱性を完全に検出することはできない点に注意が必要だ。 （セキュリティの）研究者たちは、IoT 製品の出荷前に見つけられ修復されなければならないメモリー管理、コード内に直接書かれた認証情報、認証(認可)のバイパスなどの脆弱性に関する問題を発見している。

テストとコードの品質

アジャイルアプローチを使用して開発することの最も重要な部分は、変更を行うことと変更を観察することとの間のフィードバックループである。 Martin Fowler のテストピラミッドは製品をテストする理想的なアプローチである。 このテスト結果は、システムの、それぞれ独立したコンポーネントをさまざまな方法で検証する多くの小さなユニットテストで構成されている。 こうしたアプローチにより、開発者は、ハードウェアコンポーネントをリセットするための時間を必要としないテストシステムの高速なフィードバックを通じて、製品に対する高い信頼を維持することができる。 開発者は迅速にテストを反復することができ、ハードウェアに対する作業をユニットテストにおいて検証することでハードウェアとのインテグレーションを確認することができる。 ユニットテストでハードウェアの代わりとして使われたスタブは、その後、配備のためのビルドの際にハードウェアインターフェイスに置き換えられる。

テクノロジーとプラットフォームは、しばしばテストツールの種類や利用の可否を左右する点に注意することが重要である。 セキュリティテストツールの選択肢は、製品のアーキテクチャと関連している。(たとえば) 一般的なハッキングに使用されるのと同じ無線ハッキングツールを IoT アーキテクチャ (におけるテストの) の一部に適用できる。 IoT の攻撃ベクトルは、依然として、その (攻撃対象となる) 技術ドメインと密接に関係している。(それが IoT システムの一部であろうとも) Web アプリケーションは Web アプリケーションであり、また、サポートされているリンクレイヤー上のプロトコルは改ざん (パケットインジェクション) することができる。 IoT ドメインには正式なプロトコル (CoAP、ETSI SmartM2M、MQTT、LwM2M) があるが、SCADA (監視制御およびデータ収集) にもまた正式なプロトコル (MODBUS、DNP、UCA) がある。 これらのプロトコルの仕様には、適切な認証やアクセス制御がなければその弱点を広く攻撃される可能性がある、プロトコルの重要な部分について述べられている。 いくつかのセキュリティテストツール、および多くの政府のコンプライアンス基準 (STIGS、NIST 800-53v4 など) は、認証、認定への準拠性を通じて SDLC のセキュリティに影響を与える可能性がある。

また、末端の機器については、電力やブロードバンド帯域の使用量が重要な場合などのケースも考慮する必要がある。 また、政府の免責がなければ、いくつかのテストシナリオは不可能かもしれない。例えば、地域コードをハッキングすることによって、無線カードの送信電力を 20db から 30db に増加させるようなテストである。 - しかしながら、これらのテストシナリオは、実際にあり得る中間者攻撃 (MiTM) シナリオに対応する。例えば、同じ ESSID (拡張基本サービスセット識別子) を持つ別の BSSID (基本サービスセット識別子) <訳注 : 無線 LAN の場合はアクセスポイント機器の MAC アドレスを意味する。つまり、同一の ESSID を持つ偽のアクセスポイントの意味>の出力を正規のアクセスポイントより増加させて、エンドデバイスが偽のアクセスポイントに誘導されることで意図されたアクセスポイントに接続しないようにするような攻撃である。

セキュアな開発環境のもう一つの重要な部分は、コード品質ダッシュボードの必要性である。 IoT デ

バイスは、ソフトウェアの品質に関する情報を提供するレポートで裏付けられる必要がある。 これらのレポートは、CI（継続的インテグレーション）におけるビルドに関連付けられ、単体テスト結果におけるコードのカバレッジ率やすべての静的分析チェックの結果を提供する。

管理プロセス

IoT 製品で使用されるライブラリはオープンソースである可能性がある。開発者が製品で使用するライブラリを検証するためのプロセスを用意しておくことが望ましい。レビューでは、少なくとも使用されているライブラリが信頼できる組織が公開しているものであることを確認する。さらに、ダウンロードしたライブラリが改ざんされていないことを確認することが重要である。検証済みのコード用のセキュアなリポジトリを持つことで、開発者は改ざんされたソフトウェアを使用していないという確信を得ることができる。

構成管理（CM）は注意を払うべきもう一つの領域である。GitHub や GitLab のようなツールは今日でもよく使われている。ソースコードを定期的に監視することでリポジトリに不正なコードが挿入されるのを防ぐ（この問題に関連するリンク）。 また、対策においては、開発者のログインの安全を保つ事や、リポジトリに資格情報（API の認証鍵等）をアップロードしないことに気を配る必要がある。このトピックに関する情報はこちらを参照いただきたい。

3. フレームワークとプラットフォームのセキュリティ機能の確認

デバイスの技術的アーキテクチャの設計を始めると、独自の方式で IoT デバイスを構築するか、既存のフレームワークを使用するかを決定する必要に直面する。 このようなフレームワークは相互運用可能なデバイスを開発することを可能にする。

統合フレームワークの選定

このセクションでは、IoT 開発作業のために現在利用可能な一般的な統合フレームワークのセキュリティ特性を検証する。 統合フレームワークは、自動化されたワークフローの一部として他の機器と通信が必要な IoT 機器を開発する場合に特に役立つ。家庭環境内で機器が連携し、他の機器やセンサーからの入力に基づいて状態を変更するような例を考えてみてほしい。 スマートなドアベルが夜に鳴ると、例えば、特定のライトを自動的にオンにする信号を送信するような場合や、誰かが起きたときにスマートウォッチがコーヒーメーカーにドリップを開始するよう指示する信号を送信する場合などである。

これらの機器対機器のユースケースでは、それらをすべて安全に連携させるための課題が明らかになる。 これらがすべて異なるベンダーによって製造されており、異種のプロトコル、暗号アルゴリズム、さらには異なる証明書フォーマットが利用されているような場合、それらを一緒に使用することは困難になる。 統合フレームワークは、機器が同一のベンダーによって製造されていない場合でも、それらが単一のエコシステムとして動作するために必要なツールを提供する。

セキュリティの観点から見れば、この相互運用性をセキュアに実装するために、必要なツールを開発者に提供するフレームワークを選択しなければならない。 これは、機器をシステムへセキュアに参加させる（Secure onboarding）というコンセプトから始まり、デバイスの能動的な管理と通信の保護をサポートする能力にまで及んでいる。 フレームワークの選択時に評価されるべきツールのいくつかを調べてみよう。

機器のシステムへの参加：これは機器を特定のネットワーク（システム）に参加させるプロセスであ

る。 フレームワークはこのプロセスを可能にするメカニズムを提供すべきである。 これには、ネットワーク認証情報の初期セットのプロビジョニングが含まれる。 また、後で所有権の変更が可能である必要もある。

設定機能：これには、IoT 機器の名前の付与と変更、パスワードの設定、機器のリセットなどが含まれる。

資産管理：これには、機器の所在地情報の管理、機器のハードウェア/ソフトウェアプロファイル、機器のファームウェア/ソフトウェアの更新を管理する機能などが含まれる。

検出機能：これは、他の機器やサービスの検出を可能にする機能である。

セキュアな接続：これは、ある機器と他の機器/サービスとの間の認証をサポートするために必要なプロトコルとアルゴリズムを含む。 これにはサーバーや他の機器の認証要素の検証、および通信の暗号化が含まれる。 また、TLS や DTLS などのプロトコルの実装が含まれることがある。 実装の際には十分に強力な暗号アルゴリズムも利用可能にすべきである。

クラウドゲートウェイ：ゲートウェイは必ずしも物理的に別個のデバイスである必要はないが、そのような形を取ることは可能である。 これらのゲートウェイは、グローバルな連携をサポートするために、ローカルネットワークとクラウド間のリンクを提供する。 ゲートウェイは、クラウドサービス事業者（CSP）に対し、暗号化、認証、および完全性が適用されたセキュアな API を実装する必要がある。

表 5

フレームワーク	通信方式	サポートされている言語	プラットフォーム	機器のシステムへの参加	資産管理	設定機能	セキュアな接続
AllJoyn	WiFi Ethernet Serial PLC	C C++ Obj-C Java	Arduino Linux Android iOS Windows Mac	●	●	●	●
Homekit	WiFi Bluetooth-LE (on top of HomeKit Accessory Protocol(HAP))	Objective-C	iOS				●
IoTivity		C C++ Java JavaScript		●		●	●
ThingWorx		Java JavaScript					●

Xively		JavaScript	Android iOS	●	●	●	●
Oracle Java Embedded		Java JavaScript					●

さて、フレームワークプロバイダのいくつかをもっと深く見てみよう。 表 6 は、これらの各フレームワークのセキュリティ機能の説明を示している。

表 6

F/W	議論
AllJoyn	AllSeen Alliance の AllJoyn は、共同開発されたオープンソースのソフトウェアフレームワークで、IoT デバイス間の相互運用可能な検出および通信を、使用されるデバイスまたはオペレーティングシステムのタイプに関係なくサポートする。 AllJoyn は、システムへの参加、資産管理、設定管理、安全な接続のコア機能をサポートしている。 これらの基本サービスのそれぞれには、いくつかの API が用意されている。 「ポリシー」は AllJoyn ベースのデバイスで重要な役割を果たす。 開発者は、ポリシーテンプレートを定義して、操作中にデバイス管理者が使用できる選択肢を制約することができる。 ポリシーテンプレートには、アクセス許可ルールのリストがある。 テンプレートは照会され、その後、セキュリティ管理アプリケーションがユーザーに特定のポリシーを適用するために使用される。運用段階では、管理者は、ポリシーファイルを照会、インストール、更新、または削除することができる。 AllJoyn は多くの機能のために証明書を活用している。 主な証明書には次の 2 種類がある。**アイデンティティ**：デバイスの識別情報を提示するために使用される。 **メンバーシップ**：グループへの所属情報を提示するために使用される。セキュリティ管理アプリケーションは、メンバーシップ証明書（参照リンク）を生成および配布するために使用される。 AllJoyn では暗号化もサポートされている。 機器間通信の暗号化では、128 ビット鍵長の AES アルゴリズムを使用可能であり、また認証においては事前共有鍵（PSK）または楕円曲線デジタル署名アルゴリズム（ECDSA）を使用可能である。
FIWARE	FIWARE は、欧州のプロジェクトから生まれたオープンプラットフォームである。 複数の業種でスマートアプリケーションを簡単に開発できるように、一連の API と機能を提供する。 FI-WARE は、MQTT、COAP、OMA LWM2M など、さまざまな通信プロトコルをサポートしている。

HomeKit	これらのガイドラインにより、iOS デバイス上のホームオートメーションアプリケーションを使用して、アクセサリの製造元に関係なく、接続された宅内のアクセサリを容易に設定、制御できる。 HomeKit は、家庭向けの多くの消費者向け IoT 対応の機能をサポートしている。 ・ 家、部屋、ゾーンを設定する ・ 電球やサーモスタットなどのアクセサリの追加、検索、および削除 ・ 複数のアクセサリを組み合わせた動作を定義する ・ ユーザーの管理 ・ Siri を使用して自宅を管理する ・ Homekit は、署名と鍵交換のために、3072 ビット鍵長の暗号と、Elliptic Curve 25519 を含む、WiFi と Bluetooth-LE 通信の両方に対する強力な暗号化要件を定義している。
IoTivity	IoTivity オープンソースプロジェクトは Open Interconnect Consortium がスポンサーである。 IoTivity は、IoT の新たなニーズに対応するシームレスなデバイス間接続を可能にするオープンソース・ソフトウェア・フレームワークである。システムに参加するデバイスの処理には、 1）Just Works、2）Random PIN、3）Certificate のオプションがある。IoTivity デバイスには、ポリシーに基づいてリソース要求をフィルタリングし、認証情報の管理とアクセスコントロールリストの管理を行うセキュリティ管理機能を提供するセキュリティリソースマネージャ（SRM）が含まれる。IoTivity JAVA API ドキュメントは、ここで見つけることができる。 IoTivity のセキュリティに関する詳細は、こちらをご参照いただきたい。
ThingWorx	ThingWorx は、Raspberry Pi などの人気のある IoT プラットフォーム用のスターターキットを提供している。ThingWorx は、デバイス間の安全な接続、デバイス管理、監視と制御、およびデバイスからのリモートデータ収集を提供する。 監視機能は、成功したログインと失敗したログインやログアウトなどのイベントをサポートする。 (ThingWorx は)さまざまな認証タイプ（http 基本、カスタム、トークンなど）もサポートしている。(ThingWorx は)ユーザーグループ（管理者、デザイナー、開発者、ゲスト、セキュリティ管理者、ユーザー）とアクセス許可をサポートする。 ThingWorx は、Amazon Web Services IoT および Microsoft Azure Cloud に直接組み込むこともできる。

Xively	Xively は、通信プロトコルとして HTTP、WebSockets、MQTT をサポートし、そのチャネルにおける TLS の使用を必須にし、エンドツーエンドのセキュリティを確保する。Xively を使用すると、開発者はデバイスを接続および管理し、IoT エコシステム全体でデータを共有できる。Xively は、デバイス管理を含む機能をサポートする REST API を提供する。デバイス、アプリケーション、サービスの認証は、Xively で API キーを使用して行われる。 TLS を使用して REST 通信を保護することができ、またそうすべきである。Xively は、IoT デバイスの能力に応じて、多くの TLS 暗号スイートオプションを提供している。Xively では、暗号安全乱数発生器（CRNG）を備えたデバイスは、ECDHE-RSA ハンドシェイクプロトコルで標準的な安全な TLS 接続を確立できる。 この場合、OCSP ステープリングにより、サーバー側証明書の有効性が保証される。デバイスが nonce ベースの OCSP をサポートしていない場合は、証明書の妥当性チェックを確実に行うために正確なクロックが必要である。 擬似乱数発生器（PRNG）および書き込み可能メモリーを備えたデバイスは、同様の標準的な安全な TLS 接続を確立できる。これには、デバイスに埋め込まれた、元となる暗号学的にランダムなシードが必要である。これらのデバイスの場合、PRNG 状態は常に確保され、Xively はソフトウェアベースの CRNG を提供する。 これらの機能を持たないデバイスは、TLS-PSK 標準を使用して Xively に接続できる。TLS-PSK 標準では、デバイスに埋め込まれる固有の暗号学的にランダムな認証情報のみが必要である。この動作モードには、通常、証明書ベースのモードが提供するセキュリティ上の利点はない。
Oracle Java Embedded	Oracle の Java ME Embedded は、リソースが制約されたデバイスでの操作をサポートする。Java ME Embedded Profile は、保護ドメインに基づいたセキュリティモデルを提供する。（Java ME Embedded Profile は、）アプリケーションの署名を可能にし、デバイスの設定変更可能なセキュリティ・ポリシーをサポートする。セキュリティモデルの詳細はこちらをご参照いただきたい。

プラットフォームのセキュリティ機能を評価する

デバイスのテクノロジーアーキテクチャの設計を開始するときには、各ハードウェアおよびソフトウェアコンポーネントによって提供されるさまざまなセキュリティ機能を検討すべきである。図 2 は、典型的な IoT 製品の技術レイヤーを示している。製品を保護するための多層防御アーキテクチャを作成するために、これらのレイヤーごとにセキュリティ機能を評価することができる。

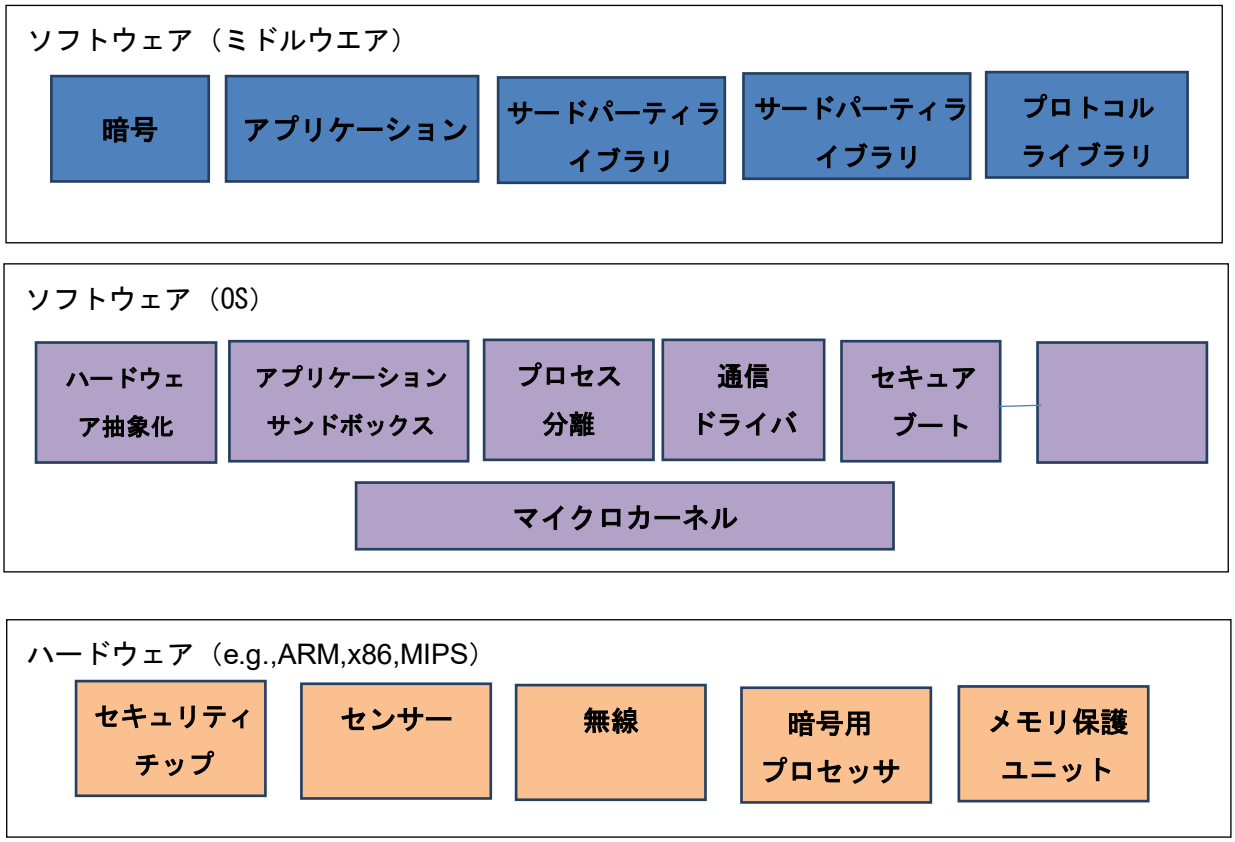


図 2

可能であれば、マイクロハードウェアのセキュリティ保護に加えて、安全なオペレーティングシステムの使用が適切である。多くの IoT デバイスプロファイルは、厳密なアクセス制御、信頼できる実行環境、高いセキュリティのマイクロカーネル、カーネル分離、およびその他のセキュリティ機能を備えたさまざまなセキュアブートオペレーティングシステムを含む、リアルタイムオペレーティングシステム（RTOS）を実行できる小型で強力な SoC ユニットに詰め込まれている。また、IoT デバイスのカテゴリが異なると、図 3 に概説されているように異なる RTOS ソリューションを必要とすることにも注意する。

Saftey Critical	安全性の証明を満たす必要があるかもしれない
産業用	堅牢な信頼性とセキュリティ機能が必要
ビジネス用	プロセス分離と追加のセキュリティ機能の要件を導入
消費者用	安全性/セキュリティ機能よりパフォーマンスとユーザビリティのために調整されることが多い

図 3

この分類のトップにある（高安全用途の IoT デバイス）では、RTOS の選択は、業界固有の認証を満たす必要があるかどうかを重視して判断しなければならない。これらの認証の例には、[21]がある。

表 7

認証	詳細	
DO-178B	航空用システムソフトウェアの要件と、航空電子システム用機器の認証	Link
IEC 61508	産業用制御システムの機能安全	Link
ISO 62304	医療機器ソフトウェア-医療機器のソフトウェアライフサイクルプロセス	Link
SIL3/SIL4	輸送および原子力システムの安全度水準	Link

Wei Dong と Chun Chen による 2010 年の完成度の高い IEEE 論文は、無線センサーをサポートするように設計されたオペレーティングシステムに関する考慮事項について確かな理解を与えてくれる。すべての IoT デバイスが無線センサーネットワーク（WSN）ベースのコンポーネントが直面するのと同じ課題に直面するわけではないが、多くの IoT デバイスは、WSN に類似の特性を持つ（バッテリー駆動、限られた処理能力、限られた記憶スペースまたは記憶スペースがないなど）。Dong と Chen による論文は、は RTOS によって提供される一連の典型的な機能を概説している：

- タスクスケジューリング
- ダイナミックリンクとロード
- リソースの抽象化
- センサーインターフェイス
- ハードウェア抽象化

高安全用途の IoT システムを扱う際に考慮すべき、非常に堅牢な RTOS の例として LynxOS や Green Hills Software がある。これらは一般にサイバー・フィジカルシステムと呼ばれる。

IoT デバイスに適した他の多くのオペレーティングシステムがある。 表 8 は、これらの選択肢の多くを示している。 これらの RTOS の多くは、インストール可能なプラットフォームの種類に制限があることに注意すること。

表 8

O/S	詳細
TinyOS	TinyOS は、低電力組み込みシステム用に設計されたオペレーティングシステムである。従来の意味での OS ではなく、組み込みシステム用のプログラミングフレームワークであり、各アプリケーションにアプリケーション固有の OS を構築できるコンポーネントセットである。TinyOS は C の方言である NesC 言語で書かれており、分割フェーズのインターフェース、非同期イベント、および遅延計算で呼ばれるタスク（参照リンク）に基づいたイベント駆動型の同時実行モデルをサポートしている。
Contiki	UDP、TCP、HTTP 機能、および 6loWPAN と CoAP を含む完全な IP スタックをサポートする。超低電力システムで動作するように設計されている。802.15.4 通信用のリンク層暗号化を含む。
Mantis	ワイヤレスセンサープラットフォーム用の組み込みマルチスレッドオペレーティングシステム。カーネル、スケジューラ、およびネットワークスタックを含む、500 バイト未満の小さな RAM に収まるフットプリントで実装されている。リモートアップデートとリモートログインによるセンサーのリモート管理をサポートする。**スリープモードによるエネルギー効率を実現する。**（*sha, carlson，et al.，MANTIS OS：an embedded multithreaded operating system for wireless micro sensor platforms*、*ACM Digital Library*）
Nano-RK	監視と環境モニタリングに特化している。エネルギー効率を実現する。プリエンプティブなマルチタスクをサポートする。CPU とネットワーク帯域幅を確保する。ネットワークをサポートする。2KB の RAM と 18KB の ROM で動作する。
LiteOS	（Unix に似ている）階層ファイルシステムと無線でアクセス可能なシェルが含まれている。また、リモートデバッグシステムを提供する。10KB を使用し、多くの接続されたマシンで動作するように設計されている。
FreeRTOS	多種類のマイクロコントローラ/マイクロプロセッサで使用される。TCP ネットワーク機能を追加する機能が含まれている。通信リンクの SSL / TLS セキュリティもサポートする。Wolf SSL のような暗号ライブラリが FreeRTOS 上に移植されている。
SapphireOS	メッシュネットワークとデバイス検出をサポートする。Python ツールがロードされ、RESTful な API サーバーが付属している。

BrilloOS	もともとホームベース（消費者向け）の IoT デバイス向けに設計/最適化されており、32～64MB の RAM が必要である。
uCLinux	組込みマイクロ Linux。ユーザーアプリケーション、ライブラリ、ツールチェーンのコレクションが含まれている。マルチタスクをサポートしていない。uCLinux の詳細はこちらをご参照いただきたい。
ARM mbed OS	ARM ベースのプロセッサを使用する IoT デバイス用に、mbed OS は uVisor を提供している。uVisor は、メモリー保護ユニット（MPU）（参照リンク）を備えた Cortex M3、M4、および M7 MCU 上の隔離されたセキュリティドメインの作成をサポートする mbed OS の監視カーネルである。アプリケーション開発者は、uVisor を使用して暗号鍵を安全に保管することができる。
RIOT OS	8 ビット、16 ビット、および 32 ビットプラットフォームで実行できる。TCP/IP ネットワーキングスタックを提供し、6LoWPAN、IPv6、UDP、CoaP などの IoT で使用されるプロトコルをサポートする。C および C++をサポートする。マルチスレッドをサポートする。1.5KB の RAM と 5KB の ROM が必要である。
VXWorks	ウインドリバーが提供。2 つのバージョン：VXWorks と VXWorks +。 オプションのアドオン・セキュリティ・プロファイルには、セキュア・パーティショニング、セキュア・ブート、セキュア・ランタイム・ローダーおよび高度なユーザー管理などの機能が含まれる。 また、ネットワークセキュリティ機能と暗号化されたコンテナも含まれている。
LynxOS	TCP / IP、IPv6、およびセルラー（2G、3G、4G）通信を含む多くのネットワーキング機能をサポートする。また、802.11 WiFi、ZigBee、Bluetooth をサポートしている。強力な暗号化サポート、アクセスコントロール、監査、アカウント管理など、さまざまなセキュリティ機能を備えている。
Zephyr	リソースが制約されたシステム向けに設計されている。安全な開発に重点を置いたオープンソースプロジェクト。ナノカーネルとマイクロカーネルを実装する。Bluetooth、Bluetooth-LE および IEEE802.15.4（6LoWPAN）。
Windows 10 IoT	Bitlocker の暗号化と安全な起動をサポートする。DeviceGuard と CredentialGuard の機能も含まれている。Windows Server Update Service（WSUS）を使用して更新をサポートする。
QNA	車載情報（infortainment）システムでよく使用されるオペレーティングシステム。サンドボックスでアプリケーションを起動する機能や、システム特権レベルをきめ細かく制御する機能など、さまざまなセキュリティ機能を提供する。
Ubuntu Core	読み取り専用のルートファイルシステム。アプリケーションのセキュリティサンドボックス。OS とは別の（独立した）アプリケーションのアップデートをサポートする。アプリケーションを信頼できる、または信頼できない、に分類する機能を（OS によって）サポートする。Unified Extensible Firmware Interface（UEFI）を使ったセキュア・ブートをサポートする。Ubuntu Core セキュリティ機能の詳細はこちらをご参照いただきたい。

考慮すべきもう一つの注意点は、多くの IoT デバイスがバッテリー消費を最適化するように設計されていることである（例えば、OS はデバイスを通常はスリープ状態にし、アクティビティを実行する必要があるときにのみ起動する)。 デバイスを稼動状態にしておくことで、製品のバッテリーを消耗させようとしている者（攻撃者など）がいると考えておくべきである。

4. プライバシー保護の確立

米国連邦取引委員会(FTC)の委員長であるエディス・ラミレスは、2016 年 1 月、IoT デバイスの激増によって、収集されるパーソナル情報がどう使われているか、適切に保護されているかどうかという懸念が生じると警告した。

詳細な IoT に関する報告書によると、FTC は消費者のプライバシーやセキュリティを強化し、保護するために企業がとることができる一連の具体的なステップを推奨している。

- 設計プロセスにおいて、後付けではなく最初からデバイスにセキュリティを組み込むこと。
- セキュリティの重要性について従業員を訓練し、組織のセキュリティが適切なレベルに管理されていることを確実にすること。
- 外部サービスプロバイダーを採用する場合は、彼らに適切なセキュリティを維持する能力があることを確認し、彼らに対して適切な監督を実施すること。
- セキュリティリスクが特定されたとき、リスクに対して複数のセキュリティ層を使用して防御する"多層防御(defens-in-depth)"戦略を検討すること。
- 利用者のデバイス、データ、あるいはネットワークに保存された個人情報に対して、未認可のユーザーにアクセスさせないよう、必要な処置を検討すること。
- 接続するデバイスをそのライフサイクルを通してモニターすること、そして、可能であれば、発見されたリスクに対応したセキュリティパッチを提供すること。
- データの最小化を考慮すること。すなわち、消費者データの収集を制限すること、必要な期間だけ保持し、無期限に保持しないこと。

EU のデータ保護指令第 29 条作業部会(WP29)は、2014 年に報告書を発行し、EU のデータ保護ルールがいかに IoT に適用するかを詳述している。WP29 は EU 加盟各国のデータ保護当局の代表で構成されたアドバイザリーグループである。したがって、見解は法的拘束力を持たないが、説得力があり、EU の規制当局が特定の問題についてどのような決定をするかについての良い示唆を与える。WP29 の見解は IoT 製造業者、開発者、そしてデータ収集者が考慮すべきプライバシー問題に関するいくつかのガイダンスを提供する。推奨として以下を含む。

- プライバシー影響評価(PIA)は新しいデバイスやアプリケーションが発売される前に実行されるべきである。
- IoT 関係者は一般的に集約されたデータのみを必要とするため、処理に必要なデータの抽出後、直ちに生データはデバイスから削除されなければならない。
- プライバシーバイデザイン、そしてプライバシーバイデフォルトの原則に従う。

- データ主体とユーザーは自分たちの権利を実行できなければならない、そしていつでもそれらのデータを”管理”できなければならない。
  EU データ保護ルール下では、ユーザーは収集され処理されるデータのタイプ、受信されるデータのタイプ、そしてそれがどのように処理され使われ組み合わせられるかについて通知されなければならない。
- ユーザーが理解できるプライバシー通知を提供し、ユーザーの同意を得るか拒否する権利を提供すること。デバイスをネットに接続して使用すること、およびデータ処理の結果に対するユーザーの同意は、告知され誰にでも提供されなければならない。
- デバイスの提供主体が実施するデータ処理について、そのユーザーとそのユーザーが関わり合う相手（例えばウェアラブルカメラによって記録される人々）の両方に通知するようにデバイスを設計すること。
- データを転送する前に収集されたデータをユーザーに通知し、ユーザーがデータにアクセス、確認、編集できるようにすること。
- ユーザーが、データ収集の時間と頻度ばかりでなく、処理タイプについて細かく選択できるようにすること。アプリ開発者とデバイス製造業者はエンドユーザーに対し、充分なレベルの情報を提供し、可能であれば、単純にオプトアウトするか、細部にわたって同意を得るか、その両方の機会を提供すること。さらに、同意が得られない場合、データ管理者は利用目的の変更、あるいは第三者への提供をする前にデータを匿名化しなければならない。

WP29 の見解では、そのような PIA で用いられる手法は、2011 年 1 月 12 日に WP29 が RFID アプリケーションに関して採用したプライバシーとデータ保護の影響評価フレームワークに基づくのが妥当である。もし、デバイスがプライベートな個人情報(PPI)を収集し、処理し、あるいは蓄積することが判明した場合には、より厳しいコントロールが求められることになる。これらのコントロールは、ポリシーに基づくものと技術的な手段によるものの両面で行なうべきである。例えば、デバイスの配備は管理者のしかるべき承認が必要となる。PPI データを IoT デバイス上に保存する必要があるか判断するには、PIA を実施しなければならない。デバイス上に保存されるデータは、十分に強い暗号化アルゴリズムを使って暗号化すべきである。デバイスが送受信するデータは十分に強い暗号アルゴリズムを使って暗号化すべきである。デバイスへのアクセスは、物理アクセス、論理アクセスを問わず、権限を持つ個人に限定すべきである。

IoT デバイスがさらされるリスクを理解することは、複雑な問題である。デバイス製造業者は、デバイス上で加工されまたは保存されるデータ、およびデバイスが接続されたモバイルアプリケーションまたはサービスに転送されるデータのタイプについて、理解する責任がある。このことの理解は、組織がその情報資産を保護できるようにするためのセキュリティ管理策を開発過程で設計するための基礎となる。デバイス開発に際しては、追加の設計ガイダンスを検討すること。

必要最低限のデータのみを収集する IoT デバイス、サービス、そしてシステムの設計

これは簡単なことに思えるかも知れないが、データ収集に関連するもの以外を間違って取得する場合を考慮する必要がある。IoT デバイス製造業者はプライバシーを視野に入れてデータモデルを見直すべきである。この目的を達成するために以下のことを行うべきである。

- 蓄積するデータは、最低限に減らす
- データ漏えいを防ぐ

実際に、センサーがデバイスと相互作用する何らかの相手に意図せずデータを漏えいすることや、何者かがデバイスが取り込んだデータを悪用して、人の特徴を推測することが可能である。

スマートメーターのケースは良い例である。もし誰かがスマートメーターデータ(例えばエネルギー使用履歴など)に対してアクセスできたら 、家の持ち主の生活パターンを推測できるのではないか？

IoT デバイスは単独で動作するものではなく、大きなシステムの一部として動作する。デバイスで処理されたデータに関して、他のデバイスが収集したデータと組み合わせることによってセンシティブなデータが漏えいする可能性があることを、デバイス製造者は認識しなければならない。これらのガイドラインに準拠することは、とりわけ健康など、センシティブ情報を管理するアプリケーションにとって、特に重要である。(いくつかの参照シナリオはここ。)

可能な限り、匿名化できるように、IoT デバイス、サービス、システムを設計すること。個人またはモノと結びつく可能性のある情報は、匿名化のアプローチが可能かどうか、厳密に評価されるべきである。例として、コネクテッドカー(connected vehicles)やコネクテッドカーに使われるコンポーネントの製造業者は、収集し伝送したデータから運転者が特定されないことを確実にするために細心の注意を払わなければならない。

IoT 業界の状況も考えると、IoT デバイスの新しい使用事例が常に発生する可能性があることにも、注意が必要である。製品が市場に受け入れられた後で、当初想定した使用法と完全に異なる新しい使用法で使われることがあり得る。

法令遵守の要求をサポートするための、必要に応じたデバイスユースケースの分析

IoT やサイバーフィジカルシステム(CPS)を取り扱う場合の、規制や法令の遵守は、その重要性が増している。アメリカ国立標準技術研究所(NIST)は次のように指摘した。(参照リンク)

“スマートシステムとは、現実世界とコンピュータが相互に対話し、互いに協調して働く相互作用型(interactive)ネットワークである。このスマートシステムは、我々の重要インフラの基礎となり、現在、および将来のスマートサービスの基盤を形成し、いろいろな分野で我々の生活の質を向上するだろう。”

これらの進歩に伴い、法令遵守や標準適合に関する課題が生じている。例えば、健康に関係するデバイス(例えばウェアラブルデバイス)では、これらが重要な要素となる。実際、このようなデバイスは、単にカロリー計算や、フィットネスレベルの追跡の機能のみであっても、着用者の健康状態に関するより機微な情報を推定できる可能性もある。この場合、デバイスは HIPAA への準拠が要求される可能性がある。その他の法令遵守として「子供のオンラインプライバシー保護法(Children’s Online Privacy Protection Act –COPPA)があり、13 歳未満の児童が使用するように設計されているデバイスの場合に課題となる可能性がある。

特に 2015 年には、プライバシーとデータセキュリティについて法令遵守の側面から、より広い範囲で大きな注目を集めた。
その幾つかを挙げると、

- EU データ保護指令:EU で販売あるいは使用されるデバイスが対象。
- EU一般データ保護規則:(制定中)
- EU-US プライバシーシールド:EU からアメリカに個人情報を転送することを想定している場合

以上をまとめるならば、デバイスの開発者は以下のような疑問に答える必要があるだろう。例えば、健康に関連するデバイス(例えばウェアラブル)の場合は、アメリカにおいて使用される場合 HIPAA に準拠しているか？ アメリカにおいて 13 歳未満の児童が使うデバイスの場合、COPPA に準拠しているか？ EU で販売されたり、使われるデバイスの場合、EU データ保護ルール(例えば EU データ保護指令に準拠しているか？ EU からアメリカにパーソナル情報を転送することがあるならば EU-US プライバシーシールドなど)に準拠しているか？など。

IoT デバイス、サービス、システム機能のオプトイン要件の設計

多くの IoT デバイスはヘッドレスな(ディスプレイやキーボードなどの入出力機器が接続されていない)ため、ユーザーにオプトインに関する了解事項を提供するマンマシンインタフェースがない。ごく小さいディスプレイを持つIoTデバイスもあるが、これらの了解事項を表示するためには不都合なものである。しかしながら、そうであっても、IoT デバイス開発者はユーザーにプライベート情報の蓄積や共有についてオプトインする方法を考えるべきである。また、このオプトインは可能な限り詳細なものにすべきである。

IoT デバイスとインターフェースするデバイス(例えばスマートフォン)でこれらのオプトインの手段を提供することは、消費者マーケット向けには論理的な選択肢だ。業務用に使われる IoT デバイスの場合、開発者は収集された全てのデータを記述するデータシートを提供することで目的を果たすことができる。これらのデータシートは、デバイスを配備しようとする企業がステークホルダーに対しデバイスによらない方法で情報提供し、

オプトインを可能にさせることに利用できる。

技術的なプライバシー保護の実装

プライバシーが強化された検出機能

Bluetooth デバイスが、誰でも手に入れられる静的 MAC アドレスをサポートする設計になっていないことを確認すること。代わりに、ペアリングプロセスの間に MAC アドレス解決を、暗号を用いて実施できる Bluetooth 4.2 の仕様で導入された BLE のプライバー機能を実装すること。

証明書のローテーション

いくつかの場合に、いかなる情報も IoT デバイスの所有者とひもつけることは容認されない。コネクテッドカーのケースにおいて、トランザクションにデジタル署名するために使用される証明書の鍵ペアをプールしておいて、それを常にローテーションするスキームがある。プライバシー要求が絶対的である場合、このようなアプローチを考えること。

5. ハードウェアベースのセキュリティ機能の設計

IoT において、製品開発者は開発製品のハードウェア保護メカニズムの評価および実装を行わなければならない。IoT 製品で懸念すべき攻撃方法は、ソフトウェア脆弱性や設定ミスだけでない。IoT のハードウェアをセキュアにするために次の点を考慮する必要がある。

1. 内部ハードウェアコンポーネントから直接データ（ファームウェアを含む）を抽出できるツールが利用可能であること
2. 不正アクセスに使用される可能性がある内部のハードウェアデバッグ用インターフェースは、ラベルがなくても攻撃者は見つけ出すことができること
3. 攻撃者がファームウェアを盗み出した場合、攻撃者は改竄したファームウェアをアップロードして、製品が本来意図する動作を変更することができること

IoT デバイスは、通常ハードウェアとソフトウェアが混在している。コンポーネントの組み合わせは、マイクロコントローラ（MCU）とセンサーをプリント回路基板（PCB）上で組み合わせて使用するだけの簡単なものにすることができる。それ以外の実装ははるかに複雑で、その中には意図的に（何らかの目的があって）露出され保護されないままの多数のハードウェアポート（例えば、JTAG）を持つものもある。

マイクロコントローラ（MCU)

最初に行う選択のひとつは、使用するマイクロコントローラ（MCU）を決定することである。現時点の IoT の主なアーキテクチャには、ARM、MIPS、および x86 がある。ハードウェアセキュリティメカニズムの詳細については、PRPL Foundation によってリリースされたレポート（Security Guidance for Critical Areas of Embedded Computing）を参照すること。

IoT 開発用の MCU の選択は、技術選択プロセスの典型的な出発点である。 MCU の選択は、IoT デバイスの（低電力アプリケーション、パフォーマンスアプリケーション、さらにはワイヤレスアプリケーションといった）機能要件に大きく左右される。これらのシステムオンチップ（SoC）ソリュ

ーションは、一部の IoT デバイスに必要な多くの基本機能を提供する。 一例として、SoC ソリューションは、単一チップ上に緊密に統合された NFC（Near Field Communication）トランスポンダを MCU に提供することができる。

いくつかの IoT デバイスはより複雑であるが、多くのセンサーは消費電力と計算能力の両方で大幅に制約されているため、選択された SoC ソリューションの上に追加する技術コンポーネントを最小限にすることが要求される。 いずれにしても、IoT デバイス開発基盤用 SoC の選択は、セキュリティに関する重要な考慮事項である。 SoC を選択する際は、次の点を考慮する必要がある。

1. SoC はセキュアなファームウェアの更新をサポートするために活用できる暗号化ブートローダー[20]を提供しているか？
2. SoC は、効率的な暗号処理をサポートするためのハードウェア暗号アクセラレータを提供しているか？　どのアルゴリズムをアクセラレータがサポートしているか？
3. SoC はセキュアなメモリー保護を提供しているか？
4. SoC は、ビルトインされた改ざん防止機能（例えば、JTAG セキュリティヒューズ）を提供しているか？
5. SoC はリバースエンジニアリングに対する保護を提供しているか？
6. SoC は、不揮発性メモリーに暗号鍵を保存するための安全なメカニズムを提供しているか？

従来からの情報技術（IT）の意味では、ハードウェアセキュリティ保護は、HSM（Hardware Security Module）や TPM（Trusted　Platform Module）のような形態をとる。これらのデバイスは、IoT エコシステム全体では一定の役割を依然として担っているかもしれないが、デバイスレベルでは、（リソースの制約とコストを考慮すると）適用範囲が限られる。

しかし、いくつかの堅牢な IoT プラットフォームは、TPM を含む機能をサポートしている場合がある。 このセクションでは、IoT デバイスの設計者や開発者が、セキュリティコンポーネントを使用してデバイスを保護するための計画を立てる際の選択肢について検討する。 また、IoT 製品を悪意のある物理的なアクセスから保護するために利用できるいくつかの選択肢についても検討する。

ハードウェアを信頼ゾーンと非信頼ゾーンに分離をサポートする技術もある。ARM の Trust Zone などのアプローチは、サンドボックス化、ファームウェア保護、安全な信頼の基点(root of trust)、コード分離などの機能を提供する。 図 4 は、ARM TrustZone の分離を示したものである。

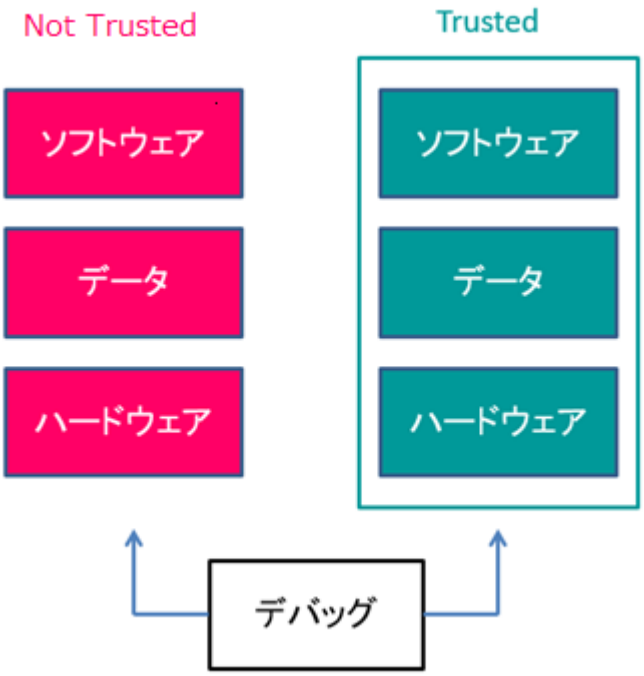


図 4

TPM（Trusted Platform Module）

常にできるとは限らないが、IoT デバイスに TPM（Trusted Platform Module）を組み込み、デバイスの暗号境界(cryptgraphic boundary)の保護を検討すること。 これはコストを増加させ、制約のある IoT デバイスでは達成できないことが多いが、機密情報を処理または保存するデバイスを作成する場合は、このオプションを検討することを強く推奨する。TPM は信頼ゾーン(zone of trust)を、他の部分の設計に拡張し、それらが侵害されないことを保証するために利用できる。TPM は、暗号化、復号、および認証に関する既知の標準を使用して、システムから、もしくはシステムへの送信を認証および認可（参照リンク）することにより、これを実現する。

メモリー保護ユニット（MPU）の使用

いくつかの MCU 製品には、MCU と統合できるオプションのメモリー保護ユニット（MPU）が含まれるものがある。MPU はメモリー上の領域へのアクセスルールを提供する。MPU を組み込んだ IoT 製品は、（ソフトウェアが）どのメモリー領域を読み込み、書き込み、実行できるかを制御できる。

物理的複製防止機能(PUF)を組み込む

物理的複製防止(PUF)デバイスは、デバイスが持つ物理的特性（のばらつき）から取得した値を使用してデバイス固有の ID を発生する情報セキュリティデバイスである（参照：R. Pappu、B. Recht、J. Taylor、and N. Gershenfeld、Physical　one-way functions、Science 297、pp.2026-2030,2002）。そのような ID の検証は簡単だが、（どのような ID が生成されるかの）予測は困難である。 ロバート・ヤングのような研究者は、PUF の概念をナノスケールに適応させる興味深く有用な研究を行っている。 量子閉じ込め PUF（QC-PUF）に関するヤング博士の研究は、デバイスにおいて分子レベルの情報から ID を導き出すものである。 彼らによって多くのプラットフォームで使用できるツールキットが提供されている。

Microsemi の SmartFusion FPGA（Field Programmable Gate Array）などのデバイスで、これらの機能（リファレンスリンク）が提供されている。

セキュリティ専用のチップ/コプロセッサの使用

多くの MCU ベンダーは、資格情報やその他の暗号化データを IoT デバイスに安全に保存することに特化したセキュリティチップを販売している。ウィキペディアはセキュリティコプロセッサによって提供される機能の一覧を提供している。

- 改ざんの検出および改ざんの証拠の保全
- 内部信号の読み取りを防ぐチップ内の導電性シールド層
- 処理の時間差による秘密情報の漏洩を防ぐ制御された処理
- チップに対する不正操作（tampering）が発生した場合の秘密情報の抹消
- オペレーティングシステムをロードする前に認証することにより信頼チェーンを保証するブートローダーのための機能
- ロードする前にアプリケーションソフトウェアを認証することにより信頼チェーンを保証するオペレーティングシステムのための機能
- 一方向の特権分離モデルを実装する、ハードウェアベースの機能レジスタ

表 9 は、現在市販されているこれらの MCU の一部を示している。 開発しようとしている IoT 製品のハードウェアに合わせ、各々のソリューションを評価する必要がある。

表 9

ベンダー	製品	内容
NXP	A710X	このファミリには、オプションの X.509 証明書ベースの認証モジュールが搭載された専用の MX51 セキュリティ CPU とオンチップの暗号化ライブラリが搭載されている。 またこれらには AES コプロセッサとアクティブシールド技術が含まれている。 この MCU ファミリの詳細は**こちら**を参照のこと。
Atmel	Crypto Authentification	ハードウェアベースの鍵ストレージを提供する。 安全なダウンロード/ブート、複製防止とメッセージセキュリティをサポートする。 詳細は**こちら**を参照のこと。
Atmel	CryptoMemory	高セキュリティ EEPROM メモリーを 16 の異なるセクションに分けることができる。 詳細は**こちら**を参照のこと。
Atmel	ATECC508A	暗号コプロセッサ
microchip/ ATMEL	CEC1302	鍵生成および認証機能を提供する。 暗号処理のアクセラレーションをサポートする。
ST Microelectronics	(ST32G512A/ ST33G1M2A)	コネクテッドカー市場向けに作られているチップ。信頼の基点(root of trust)を提供するための機能を内蔵し、認証をサポートする。
Freescale	K80	MMU を含む。 暗号アルゴリズム（AES; SHA-256 など）のハードウェア実装を含む。 暗号化されたファームウェアアップデートをサポートする。
Infineon	OPTIGA TRUST PSLJ 52ACA	認証、セキュア・ブート、セキュアアップデート、鍵生成/鍵のストレージ、保護されたメモリーなど複数の機能をサポートする。

IoT デバイスのハードウェアとファームウェアへの脅威を十分に考慮して、それに対応するための要求条件とサプライチェーン全体にわたる調達手順を設計に組み込むことで、これらの脅威を最小限に抑えること。IoT デバイスを設計する際には、特別な注意と保護が必要なセキュリティ境界を特定することが有用である。例えば、デバイスの暗号境界(cryptgraphic boundary)を特定することは、最も重要な保護を提供しなければならないハードウェア/ソフトウェアの領域はどこかということを示してくれる。暗号境界の内部には、暗号処理や鍵管理などの機能があることが確認できるだろう。

暗号モジュールの使用

特に注意と保護が必要となるセキュリティ境界を特定することが有用である。例えば、デバイスの物理的な暗号による保護がどのようなものであるかを確認することは、最も重要な保護を提供され

なければならないハードウェア/ソフトウェアの領域はどこかということを示してくれる。暗号の利用と全体的な暗号の安全性も重要である。 暗号境界の内部には、暗号処理や鍵管理などの機能があり、暗号鍵の生成材料やその他の機密性の高いセキュリティパラメータの漏洩を防ぐための追加のセキュリティ保護機能を実現している。

国立標準技術研究所（NIST）は、安全な暗号モジュールのための重要であり有用な文書とツールを提供している。IoT デバイス内で暗号保護を実装するときは常に、FIPS（Federal Information Processing Standard）140-2 に従う必要がある。 IoT 開発者は、FIPS 140-2 で検証済みのモジュールを調達するか、独自のモジュールを作成して認定を受けることもできる。 暗号モジュールのセキュリティ要件の文書 FIPS 140-2 は、暗号化モジュールの設計要件の要点を示しており、それは最も緩やかなレベル 1 から最も厳しいレベル 4 までである。（表 10）

表 10

	セキュリティ・レベル 1	**セキュリティ・レベル 2**	**セキュリティ・レベル 3**	**セキュリティ・レベル 4**
暗号化モジュール仕様	暗号モジュール、暗号境界、承認されたアルゴリズム、および承認された動作モードの仕様。 すべてのハードウェア、ソフトウェア、およびファームウェアコンポーネントを含む暗号モジュールの説明。 モジュールセキュリティポリシーの宣言。			
暗号化モジュールポートおよびインターフェース	基本・オプションの区別なく、使用するすべての入力と出力データパスのインターフェース。		論理的または物理的に他のデータポートから分離された保護されていないクリティカル・セキュリティ・パラメータのデータポート。	
役割サービスと認証	基本サービスとオプションサービスの論理的分離	役割ベース、または ID ベースのオペレータ認証	ID ベースのオペレータ認証	
有限状態モデル	有限状態モデルの仕様。 基本状態とオプション状態。 状態遷移図と状態遷移の指定			
物理セキュリティ	製品と呼べる品質の機器	動作のロック、または、改ざん証拠保持	不正操作を検出して隔離(covers and doors)、	不正操作を検出し封印(envelope). EFP または EFT.
暗号鍵管理	鍵管理メカニズム：乱数と鍵の生成、鍵の確立、鍵の配布、鍵の入出力、鍵の保管、鍵の無効化（zeroization）。			
	手作業で作成した秘密鍵と私有鍵は、平文で入出力されることがある		手作業で作成した秘密鍵と私有鍵は、暗号化された状態で入出力するか、知識分割手順(split knowledge procedures）で出力しなければならない。	
EMI/EMC	連邦規則集４７編 第 15 部（47 CFR FCC Part 15）Subpart B，Class A(ビジネス向け)。適切な FCC 要件(無線向け)。		連邦規則集４７編 第 15 部（47 CFR FCC Part 15）Subpart B，Class B(家庭向け)。	
自己テスト	パワーアップテストにおいて：暗号アルゴリズムテスト、ソフトウェア/ファームウェア完全性テスト、重要な機能テストを実施。 その他、条件に応じてテストを実施。			
設計保障	構成管理(Configuration management(CM))セ	構成管理(CM)システム セキュアな配布 機能仕様	高水準言語(High-level language)の実装	形式モデル(Formal model)

	キュアなインストールと生成(Generation). 設計ポリシーの対応ガイダンス文書			詳細な説明（インフォームドプルーフ） 前提条件と事後条件
その他の攻撃の軽減	現在テスト可能な要件が存在しない攻撃を低減する仕様			

デバイスの物理的保護

リング社のビデオ ドアベルという製品名で知られている IoT デバイスに関する調査では、IoT デバイスメーカが検討すべき多層防御の対応策の必要について興味深い見解を示している。製品は家の外に設置されることを意図したものであるため、調査員が製品を取り外し、ホームネットワークの Pre-Shared Key（PSK）など、製品の動作に使用されている暗号化されていない設定データを見ることができた。簡単に外せる標準的なネジが使われている結果取り外しが可能であり、かつ設定データの機密保持がされていない。最悪の場合を想定して、（デバイスをこじ開けるような）不正開封の検知機能などの物理的対策や、さらに徹底する場合には暗号鍵の抹消などのオプションを検討すること。

最近の IoT のセキュリティ事件では、攻撃者はリバースエンジニアリング技術を使用してハードウェアとファームウェアを分析し、脆弱性やセキュリティ機能を無効にする方法を見つけ出した。一部の重要な用途に使われるデバイスでは、ファームウェアの取得、分析、変更を防止するために、リバースエンジニアリングに対する保護対策を検討する必要がある。例えば、ファームウェア更新パッケージの暗号化は、不正な取得を防ぐことができる可能性がある。不正開封に対する耐性を持つハードウェアが最も望ましい。ただし、大規模な IoT 製品の実装ではコストがかかりすぎるかもしれない。

不正開封防止

不正開封防止機能は、IoT デバイスに保存されている機密情報の侵害に対する保護を提供する。不正開封防止機能の実装方法としては、デバイスの物理的な侵害またはその試みに対する警告（不正開封明示）や、不正開封に対する積極的な防御（不正開封防止）がある。

IHS の Electronics360 のサイト上の記事で Warren Miller 氏は最近、不正開封防止の実装に関して、次の優れたアドバイスを提供した。

「あなたの PCB（プリント基板）への攻撃を検知するための、簡単な仕組みは、MCU 出力ポートから余分な信号を入力ポートに追加することである。これらの信号はボード上に縦横に配置が可能で、ハッカーがアクセスしようとする可能性のある重要な信号の前後に置くことができる。ハッカーが重要な信号にアクセスするためにボードの配線をカットすると不正検出用の信号が遮断され、攻撃が起きていることをハードウェアに警告する。ハードウェアは、システムのリセットや、コードやパスワードなどの重要な情報の消去など、さまざまな対策を講じることができる。」（参照リンク）

サプライチェーンの防御

IoT 製品のサプライチェーンの保護の議論には、多くの考慮事項がある。重要インフラストラクチャに使用するシステムでは、コンポーネントとライブラリを製品に統合することが許可されている、承認済み製品リスト（APL）を用いることは賢明な対策である。 この場合は、各コンポーネントまたはライブラリは、その使用が製品のセキュリティにリスクをもたらすかどうか、分析するために評価を受ける。

さらに、IoT 製品で使用されるすべてのオープンソースライブラリは、カスタム開発コードに対するのと同等のセキュリティチェックをクリアすることを推奨する。

また、開発者はこれらのライブラリに対するパッチの更新についても注意を払い、オープンソースコードの脆弱性が IoT 製品に侵害をもたらす可能性を減らすために、更新を可能な限り早く行うべきである。

セルフテスト

起動時にデバイスのセキュリティ機能を検証するセルフテストを実装すること。

エラーを検出した場合、エラーをログに記録し、可能な限り処理を中止すること。

セキュアな物理インターフェース

物理インターフェースの安全を確保する目的のひとつは、分析と変更のための製品ファームウェアの負荷を軽減することである。高保証タイプの実装（例えば、FIPS 140-2 レベル 4）では、物理的にパッケージが破壊された場合にデバイスを役に立たないようにする自律的破壊対抗措置（active tamper protections）が要件となっている。 ほとんどの IoT 製品にはこのようなセキュリティ上の高度な機能がないため、デバイスの物理的保護のスキームに自律的破壊対抗措置のような保護が組み込まれる可能性は低い。

これは、攻撃者（および研究者）がデバイスの内部に簡単にアクセスできることを意味する。一旦内部にアクセスできてしまえば、内部コンポーネントを分析して、物理ポートが外から見えデバイスへの、（例えばファームウェアまたはコマンドプロンプトなどへの）不正アクセスができるかどうかを判断することができる。IoT 製品に関して２つの問題点が指摘されている。

- UART（ユニバーサル非同期レシーバトランスミッタ）ピン
- JTAG（ジョイントテストアクショングループ）インターフェース

カスタムツールを使用して、攻撃者はこれらのインターフェースに接続できる。 したがって開発プロセスに、出荷前にこれらのインターフェースを無効にするステップを含めることが重要である。さらに、これらのインターフェースにパスワード保護を設定することもできる。

IoT 製品は、他にも物理インターフェースを持つ場合がある。 可能であれば、USB ポートをロックダウンして信頼できる接続のみに限定すること。デバイス内に組み込まれたサードパーティー製の入出力機能のセキュリティをいかに管理するかを考えること。

また、デバイスが他のデバイス（上流のコンピュータやスマートフォンなど）に接続されているかどうかも検討すること。デバイス間の信頼関係を利用して、そのインターフェースを悪用することはできるか？ IoT 製品が接続するコンピュータは実際に信頼できるか？ オペレータに対し、信頼されていないデバイスとデータを共有することについての許可を求める通知を送信することを検討すること。

最後に、運用製品からデバッグコードを削除し、攻撃対象にされないようにすること。

6. データ保護

IoT デバイス開発者の主要な課題の 1 つは、異なる IoT プロトコル間での相互作用や各プロトコルにてセキュリティを階層化するのに最適なアプローチについて理解することである。これらのプロトコルは、IoT デバイスが通信を確立するための、多数のオプションがあり、データリンク層における認証や暗号化の機能を提供する。例えば、IoT の通信にて多用される Zigbee、ZWave、Bluetooth-LE には、認証やデータの整合性・機密性を保護するための設定オプションが用意されている。これらプロトコル各々が IoT デバイスの無線ネットワークを構築する機能をサポートしている。また Wi-Fi も多くの IoT デバイスの無線ネットワーク構築における選択肢となる。

IoT の通信プロトコル上で流れるのは、データ通信主体のプロトコルである。多くのプロトコルはセキュリティ機能のための下位サービスを必要とし、IoT 通信プロトコルや DTLS や SASL といったセキュリティに特化したプロトコルが用いられる。IoT におけるデータ通信主体のプロトコルは 2 つのカテゴリに分類できる。CoAP のような REST（Representational State Transfer）型のプロトコルと DDS や MQTT のようなパブリッシュ／サブスクライブ（Publish/Subscribe）型のプロトコルである。これらのプロトコルの多くは IP 層を必要とするが、MQTT-SN といった一部プロトコルのように Zigbee のような無線リンク上でそのまま動作するものもある。

IoT に関して注意すべき重要なポイントの 1 つは、従来の（今日知られている）インターネットは人間 - マシン通信が中心である。しかし、IoT の急速な普及と進化により、マシン - マシン（M2M）通信を考慮しなければならない。これはシステムにもう一つの決定的相違をもたらし、デバイスの性質や使われ方の結果、一定のデータ流量にはならない（むしろバースト的流量）。データパケットはターゲットが限定され簡潔なものとなる。実際の集約や分析の処理は、IoT より上流、つまりゲートウェイ装置またはクラウドサービス上で実施される。

また、IoT デバイスはコスト最適化の観点からも考慮する必要がある（想定すべき使用量からすれば、50 ドル以下、かつセンサーは 1 ドル以下）。その結果として重いプロトコルスタックの実装は無意味になってしまう場合がある。

データ定義については、IoT における管理の容易性の観点、およびコンテンツの簡易な特性と小さなデータパケットサイズをサポートできるかについて見ておく必要がある。例えば以下のような選択肢がある。

- Apache Avro
- Apache Thrift
- Protocol buffer

IoT の特性からすると多くの IoT プロトコルの選択肢があり、これらのプロトコルの多くはトランスポート層として TCP ではなく UDP を用いる。トランスポート層の障害耐性はデータストリーム

の正常化によって対処することができる。例えば、数分ごとに温度データを報告する環境センサーからの数分のデータロスは許容できるとみなすことができる。この点は、求められる障害耐性水準が処理されるデータの種類に大きく左右されるため、IoT デバイスにて用いる適切なプロトコル群を選択する際に、考慮に入れる必要がある。

IoT 通信プロトコル選択のためのセキュリティ上の考慮事項

選択肢が多くあるので、通信プロトコルの選択は明らかに IoT 製品のユースケースに依存する。今日の環境では、メーカーは多くの場合、限られた容量の電池を持つデバイス向けに最適化された短距離通信を取り入れる。そこではデバイスは、通常はスリープモードにあり、データ送受信の都度起動状態となるように作られている。

IoT 通信プロトコルに関連する様々な要素は急速に変化している。現在、携帯電話通信網は全ての IoT デバイスタイプで広く使われているわけではないが、今後、第 5 世代携帯電話網（5G）の導入によって変わる可能性がある。第 5 世代携帯電話網（5G）は、広帯域をサポートし数百万のデバイスの接続を可能にするように最適化されている。こうした変化は、ゲートウェイ装置に過度に依存するアプローチから脱却することが可能になるため、将来 IoT システムのアーキテクチャに大きな影響をもたらす可能性がある。第 5 世代携帯電話網（5G）や従来の携帯電話通信網においては、デバイスの認証や識別のための情報をセキュアに格納するために SIM カードが一般的に用いられている。これらの SIM カードのサイズは、多様な IoT デバイスで利用可能なまでに大幅に縮小化が進んでいる。

通信プロトコル選択の際に IoT 製品開発者が防御すべき、様々な攻撃手法が存在する。選択した通信プロトコルがこうした攻撃すべてを防ぐことができない場合でも、必要に応じ製品／システムに追加的防御策を実装するために、多層防御の概念を適用すべきである。

- 有線／無線ネットワークに対するスキャンとマッピングの攻撃
- プロトコル自体への攻撃
- 盗聴（機密保持の喪失）
- 暗号アルゴリズムや暗号鍵管理に対する攻撃
- スプーフィングと成りすまし（認証に対する攻撃）
- サービス不能（DoS）攻撃や通信かく乱
-

IoT 製品開発者は利用可能な様々なプロトコルを理解するために時間を割く必要があるが、あるプロトコル（物理層／データリンク層）が他のプロトコルの基盤として機能している。例えば IEEE 802.15.4 は、ZigBee などの他の IoT プロトコルの下層の標準となっている。

どの通信プロトコルの場合でも、検討すべき重要なセキュリティ上の考慮事項の 1 つに、ネットワ

ークへの参加やペアリングを行うプロセスがある。例えばあるデバイスが既設のネットワークに参加しようとする際に、正規のデバイスだけがネットワークに接続できることを確実にするためにどのようなコントロールが実装されるべきか。いくつかのプロトコルではデバイスのネットワークへの追加接続を極めて容易にする“確認なし（Just Works）”のようなオプションが用意されているが、これは家庭やビジネス環境のネットワークへの不正なデバイス接続に関するセキュリティ上の問題をもたらすことになる。

ペアリング処理の例として、Bluetooth-LE におけるプロセスを示す。

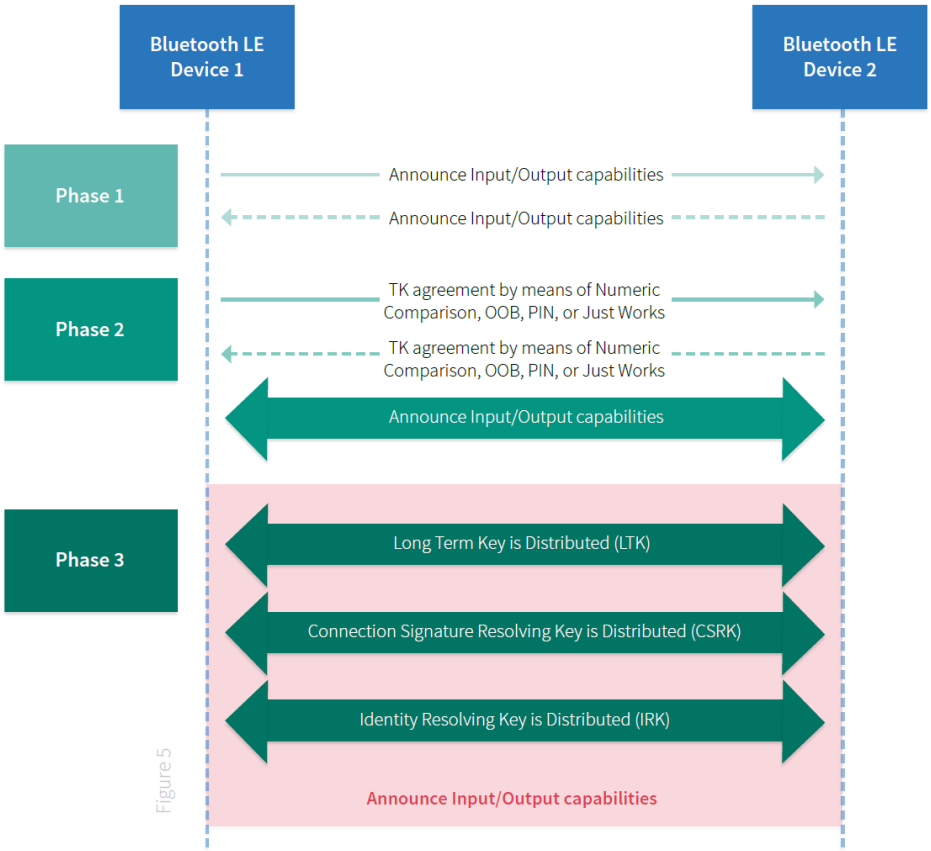


図 5

Bluetooth-LE のペアリングプロセスについて読み取れることは、プロセスの３つのフェーズに対して生成されるべき複数の鍵がある、ということである。フェーズ１は、認証や機密保護のためにどのモードを使うかを決定するセキュリティ機能のネゴシエーションである。フェーズ２はコンポーネントが認証されるフェーズであり、いくつかのメカニズムが使われる。そのメカニズムとしては、数値コード入力による認証（PIN）、NFC 等の他の通信による認証（OOB）、ECDH 鍵を用いた数値コード表示確認（Numeric Comparison）等、複数の方式が提供されている。認証のための「確認なし（Just Works）」という方式が最も利便性が高いが、可能な限り避けるべきである。

通信プロトコルが提供する、デバイス間、またはデバイスとアプリケーション間の信頼関係の維持も、セキュリティ上の考慮事項となる。その通信プロトコルでは信頼できる通信セッションを維持す

るために再認証が必要かどうかである。

必要となるゲートウェイ装置が危険にさらされる可能性も考慮する必要がある。その通信プロトコルがゲートウェイ装置を必要とし、そのゲートウェイ装置が侵害された場合、クラウド上のサーバーのような最終送信先にデータが届く前に攻撃者がデータを操作することが可能である。一般消費者向け IoT の場合、スマートフォンアプリや、Wi-Fi ルータでさえゲートウェイ装置となる。

いくつかのプロトコルはデバイスを検出するためにビーコンを用いる。ビーコンは多くの場合一定の範囲内に配信されている。認証不要でビーコンメッセージが配信されている場合、簡単に成りすまされてしまう可能性がある。

表 11

IoT 通信 プロトコル	セキュリティ検討上のポイント
LTE	認証および鍵合意のための手順（AKA：Authentication and Key Agreement）が、利用者機器（UE、つまり IoT デバイス）と基幹ネットワークの間の認証の基盤として機能する。利用者機器は、モバイル機器と、認証情報をセキュアに保持する USIM により構成される。基幹ネットワークでは、Home Subscriber Server (HSS)と Authentication Center (AuC)が利用者認証のための情報を管理する。 キャリアが AuC（ネットワーク側）と USIM（利用者側）に、事前共有対称鍵を格納する。ユーザーとネットワークの相互認証により、派生鍵 ASME（Access Security Management Entity）が生成され、その鍵（ASME 鍵）は、NAS、RRC（Radio Resource Control）シグナリング、ユーザプレーン用の暗号鍵・および完全性保証鍵の生成に使われる。事前共有対称鍵は、製造時に USIM に組み込まれ、（派生鍵は）ネットワーク参加時にネットワーク側に渡される。
GPRS	GSM におけるパケット通信サービス。 全てのデータおよび信号は論理リンク制御（LLC）層にて GPRS 暗号方式（GEA）により暗号化される。認証情報や鍵は SIM カードに格納される。
GSM	携帯電話の通信方式。時分割多重アクセス（TDMA）方式を用いる。 IoT デバイスや携帯電話にて、認証情報や秘密鍵は SIM カードに格納される。 発生が想定される問題としては、悪意ある基地局の設置、弱い鍵（小さいビット数の鍵）の利用、平文での鍵送信等が挙げられる。

IoT 通信 プロトコル	セキュリティ検討上のポイント
UMTS	UMTS：Universal Mobile Telecommunications System。携帯電話の通信方式。W-CDMA 方式とも呼称される。 ユーザデータおよび信号は暗号化されている。128 ビット暗号鍵を使用。暗号化および完全性保証のため KASUMI アルゴリズムを用いる。
CDMA	携帯電話の通信方式。符号分割多重アクセス（CDMA）方式を用いる。SIM カードは使用しない。
Long Range Wide Area Network (LoRaWAN)	0.3 kbps ～ 50 kbps の通信帯域をサポート。 LoRa アライアンスによれば、LoRaWAN では以下の 3 種類の鍵を用いる。 ユニークなネットワーク鍵(Unique Network Key) ユニークなアプリケーション鍵(Unique Application Key) （アプリケーション層でのエンド to エンドのセキュリティ） デバイス固有の鍵(Device-specific key) 詳細は LoRaWAN のセキュリティに関する MWR Labs 社 Robert Miller 氏による**ホワイトペーパー**を参照のこと。
802.11	古くからある無線通信方式（Wi-Fi）。Wi-Fi ネットワークへの接続のためにデバイスに鍵を装備する必要があるか検討すること。
802.15.4	小型・低価格・低電力を重視した無線通信方式。 認証／暗号化／完全性の制御は上位層のプロトコルにて定義できる。
6LoWPAN	可能であればネットワークへの自動接続をサポートするよう設計された低電力無線通信方式。 6LoWPAN デバイスにブートストラップ情報を配信するための 6LoWPAN ブートストラップサーバの導入が必要。これは接続デバイスに対してセキュリティ資格情報等の情報を配信するために用いられる。 セキュア版の 6LoWPAN ネットワークには、拡張認証プロトコル（EAP）などの認証をサポートするためのサーバーが必要。 6LoWPAN ブートストラップサーバはブラックリストを追跡する機能を持つ。

IoT 通信 プロトコル	セキュリティ検討上のポイント
ZigBee	センサーネットワークを主目的とする近距離無線通信方式。 IEEE 802.15.4-2003 にて定義される物理層と MAC 層を用いる。 ZigBee はスター型、ツリー型、メッシュ型トポロジーのネットワークを提供する。ZigBee セキュリティサービスは鍵交換、鍵配送、フレーム保護、デバイス管理といった機能を提供する。通信元と通信先が同じ鍵を用いることでエンド to エンドのセキュリティを提供する。MAC 層は AES-CTR または AES-CCM により保護される。 ZigBee デバイスのネットワークへの初回接続（例：デバイスがネットワークに事前に登録されていない場合）に弱点が存在する。初回接続では保護されていない状態で 1 つ目の鍵が送信され、この鍵を傍受されてしまうことが脆弱性となる。
ZWave	ホームオートメーション／センサーネットワークを主目的とする無線通信方式。 鍵にはフレーム暗号化鍵とデータ発信元認証鍵が含まれる。鍵は平文で送られることはない。ZWave のセキュリティに関する詳細情報はこちらを参照のこと。
Thread	コネクテッド・ホームを主目的とする無線通信方式。 IEEE 802.15.4 物理層と MAC 層上で構築される。最大 250 デバイスの接続をサポート。AES 暗号に対応。 Password Authenticated Key Exchange (PAKE)方式を使用。802.15.4 MAC 層ではネットワーク鍵を使用。新たなデバイスのネットワーク参加には、Key Encryption Key (KEK)にて包まれたネットワーク鍵が必要であり、"Commissioner"（と呼ばれるサーバー）が必要となる。新規ノード追加時はネットワークパラメータ暗号化のために鍵ペアを用いる DTLS 方式が認証に用いられる。 Thread のセキュリティに関する詳細情報は**こちら**を参照のこと。
Sigfox	超狭帯域（UNB：Ultra Narrow Band）を用いた広域無線通信方式。 915 MHz 帯（米国）、868 MHz 帯（欧州）の超狭帯域（UNB）を使用。メッセージ署名にデバイス秘密鍵を用いる。1 日あたりのメッセージ数 140 という上限がある。anti-replay 保護機能が取り入れられている。
Bluetooth / Bluetooth-LE	近距離無線通信方式。 バージョン 4 仕様にて接続処理に鍵交換に楕円曲線 Diffie Hellman 方

IoT 通信 プロトコル	セキュリティ検討上のポイント
	式が追加された（数値コード表示確認方式：Numeric Comparison Association Method）。Bluetooth-LE では機密性保護に 128 ビット対称鍵が用いられる。 限定的なものからより堅牢性を有するものまで、複数レベルのセキュリティ機能が用意されている。 最新仕様ではプライバシー保護強化のために MAC アドレスを固定せずローテーションするための機能がサポートされている。
NFC	近距離無線通信方式。 限定的なセキュリティ保護機能しか持たない。多くの場合、他のプロトコルと組み合わせて用いられる。
Wave 1609	Wave：Wireless Access in Vehicular Environment。 コネクテッドカーの通信にて多用される。属性タグを用いる IEEE 1609.2 証明書に大きく依存している。

7. セキュアなアプリケーションとサービスの結合

IoT の一つの面をセキュアにすることに注力するだけでは不十分である。 IoT デバイスは、より大きなエコシステムの一部として機能する。 各接続ポイントは新しい接続経路になるが、その経路は情報システムあるいは制御システムへの不正アクセスに使用できる可能性がある。IoT デバイスと組み合わせて使われるアプリケーションやサービスは、確実にセキュア開発のベストプラクティスを用いて開発されるように配慮する必要がある.

スマートフォンアプリは、よく IoT デバイスの設定や、例えば画像情報の表示のように、IoT デバイスと連携して動作するために使用される。スマートフォンアプリはまた、IoT デバイスからクラウドへのデータ転送を行う際にゲートウェイの機能を提供する。スマートフォンアプリケーション開発者は、IoT デバイスとの認証と完全性を保護する通信を可能にするセキュリティ資格情報を利用すべきである。場合によっては、開発者は、信頼できないネットワークにおける中間者攻撃（Man-in-the-Middel attack）を防止するために、Certificate Pinning（証明書をデバイスに埋め込むこと）の実装も考慮すべきである。

IoT 製品の開発者は、モバイルアプリに与えられる特権の制限も考慮しなければならない。モバイルアプリが、ある特定の IoT デバイスに影響を及ぼすことができる機能の範囲を限定しなければならない。例えば、インターネットに接続するコネクテッドカーの機能をコントロールするアプリは、リモートから管理されることが明確に想定されていないコントロールに影響を及ぼすべきでない。IoT デバイスの設定とそれらに対して連携して動作するアプリケーションの両方に対して、特権アクセス機能を検討する必要がある。

モバイルアプリケーションのセキュリティに関する包括的なガイダンスについては、CSA Mobile Application Security Testing (MAST)を参照のこと。

IoT デバイスは、多くの場合、業種を越えてデバイスとサービスを結びつけるクラウドサービスと接続する。このような接続により、異なる所有者に属し、一部には完全にスレッド化された自動運転を行うような IoT コンポーネントの間でのデータ収集、分析、および処理をサポートしたり、よりうまくできるようにしたりする。多くの組織がパズルのピースを理解し、新しいやり方での処理の方法

についてブレインストーミングを始めると、常時面白い新しいアプリケーションが開発されるようになる。モバイルアプリケーションと同様に、クラウドサービスの開発者は、システムのセキュリティを厳しくする必要がある。 多くの場合、クラウド・サービスは、ゲートウェイから、もしくは直接 IoT デバイスから送信される MQTT、REST、その他の通信用のインターフェースを提供する。

クラウドサービスの開発者やクラウドサービスを使用する開発者は、内製、およびサードパーティー製のサービスをセキュアにするために、CSA の Cloud Controls Matrix (CCM)のような標準のプロセスやフレームワークを活用すべきである。

8. 論理インターフェース /API の保護

IoT 製品を開発する際に最も重要になる考慮事項の一つは、インターフェースのセキュリティである。IoT 製品がインターフェースする相手は多くのクラウドサービスがあり、カスタム開発されたスマートフォンアプリケーション、さらには他の IoT 製品がある。

適切に保護されていないと、API はサービスプロバイダーをサービス拒否攻撃に晒してしまう。乗っ取られた IoT 製品がサービスをリクエストで溢れさせようとすることに対して防御するために、帯域を制限するような技術を使用すべきである。

ゲートウェイは、許可されたデータタイプだけを通すようにチェックするのと同時に、メッセージが正しいフォーマットであることをチェックすべきである。これにより、API 通信に悪意あるコードを埋め込んで IoT クラウドサービスへの侵害を引き起こす可能性に対して防御できる。また、スキーマの検証も重要である。

エラー処理も考慮する必要がある。レスポンスを詳細にし過ぎないように注意が必要である。これらのレスポンスはトラブルシューティングには役立つが、その反面、攻撃方法を見つけようとする攻撃者に対して、攻撃対象点を非常に多く与えてしまう。

タイムスタンプやカウンタをメッセージ構造に埋め込むような手法を用いて、リプレイ攻撃に対して防御しなければならない。

セキュリティ対策の第一の手段としてプログラム/デバイスに API キーを使用することに頼ってはいけない。可能な限り、より堅牢な認証および認可の制御を実装すること。
また GitLab のようなシステムにコミットする際にコードを晒してしまう場合に、API キーまで公開しないように保護する必要がある。

さらに、すべての API 通信を暗号化しなければならない。IoT メッセージングプロトコルが TCP または UDP のどちらをサポートするかに応じて、TLS および DTLS などのプロトコルを使用する。Certificate Pinning によって、GET リクエストに含まれる機密情報の送信を保護しなければならな

い。

OWASP REST Security Cheat Sheet を参照して、IoT 製品での REST の使用に関する適切なアドバイスを確認すること。

次の点も考慮すること。

1. どのアプリケーション層 API が公開されているのか？
2. セキュリティのインターフェースは、それが書かれた言語の種類に影響されるか？（言い換えると、Java ではパスワードを String クラスオブジェクトとして格納してはいけない。なぜなら、Java String は作成後にその状態を変更できないので、メモリー上でゼロ化/消去ができないため。）
3. しばしば、相互運用の問題は、必然的にセキュリティ低下を引き起こす。信頼性が低下しても許容可能か？
4. 外部に存在するソフトウェアエージェントはあるか？ 管理、監視、トラブルシューティングなどに使用されているか？ それはどこにあるか？ 信頼関係とは何か？ エージェントのセキュリティプロファイルはどうなっているか？
5. デバイスは他のデバイスまたはサービスとの接続をイニシエイトするか、他のデバイスまたはサービスがデバイスとの接続をイニシエイトするか、またはその両方か？

アプリケーションプログラミングインターフェイス（API）を使用すると、クラウドまたはモバイルデバイス/ゲートウェイに直接接続できる。 たとえば、Amazon Web Services（AWS）の新しい IoT サービスには、AWS の IoT Gateway への直接接続を可能にする API が付属している。API は REST をサポートし、多数のセキュリティに関する動作（参照リンク）を含んでいる。

また、AWS IoT API を使用すると、開発者は IoT データの処理に関するルールを設定できる。 例えば、データは、dynamoDB テーブル、Kinesis ストリームに書き込むことができ、または Lambda 関数を呼び出しできる。 データは、別の MQTT トピックとして IoT デバイスに再発行することもできる。 Create-Topic-Rule 操作を使用すると、開発者は IoT データの正確な処理を指定できる。

AWS IoT API の興味深い点は、2048 ビット RSA のみをサポートし、本文書の発行時点(2016 年 6 月)では IoT にとって使いやすい（つまり効率的な）楕円曲線暗号をまだサポートしていないことである。 多くの場合、IoT は消費電力に制約のあるデバイスの使用を伴うため、効率的な暗号化を採用することが IoT を扱う際の重要な要素である。

マイクロソフトもまた、Azure IoT Hub との接続用に一連の REST ベースの IoT API を提供している。 例えば、Azure IoT Hub REST API は、デバイスの識別子(アイデンティティ)の作成と、認証のための対称鍵のプロビジョニングを可能にする。 Microsoft Azure はまた、IoT デバイスと Azure IoT Hub の間の双方向の REST メッセージ通信のためのメッセージング API を提供している。 マイクロソフトは API ドキュメントで、開発者は接続のセキュリティを確保する責任があると指摘し

ている。 その対象としては、デバイスのクラウドサービスへの認証（およびその逆）、接続の暗号化と完全性保護（TLS 経由）、安全な通信・アクセス・データ転送のためのポリシーの設定が含まれる。

Certificate Pinning サポートの実装

IoT のファームウェアとモバイルアプリケーションにおける Certificate Pinning 実装により、IoT デバイスが悪意あるサーバーあるいはプロキシと通信するように設定されている(Man-in-the-middle)攻撃に対して防御することができる。
これは、例えば認証局が改竄され不正な証明書を発行された場合に発生する可能性がある。Certificate Pinning により、アプリケーション開発者はアプリケーションの証明書または公開鍵をアプリケーション/ファームウェア自体に組み込める。これにより、ライフサイクルでのメンテナンスの問題が生じるものの、Certificate Pinning によって IoT デバイスの通信セキュリティに対する保護を加えることができる。

SSL pinning（SSL 証明書の埋め込み）は認証局が侵害されている場合には、セキュリティ上大きな利益があるが、実装した証明書を許可なく上書きされることから保護する必要があることを認識しておくことが重要である。このためには、各プラットフォームにセキュアなデータ格納領域が必要であると同時に、必要に応じて証明書を安全に更新できることが必要である。OWASP は、さまざまな開発言語での実装例と Certificate Pinning に関するチートシートを提供し大きな貢献をしている。

9. 安全な更新機能の提供

ファームウェア更新のセキュリティ対策が不十分だと、悪意のあるユーザーが正規のファームウェアを改竄し、新たな悪意のあるファームウェアを製品に書き込む可能性がある。 そのような悪意のあるファームウェアは、セキュリティ機能を無効にしたり、新たな機能を実装したり、データを流出させる機能を仕込む可能性がある。 IoT 製品がこのように変更されないようにしなければならない。

IoT 製品のファームウェアとソフトウェアをアップデートするための方法を決定することは、最も困難な問題の 1 つである。 考慮すべき点として、デザインを「アクティブ/パッシブ」または「アクティブ/アクティブ」にする必要があるかどうかの判断などがある。パッチを適用し、即座にパッチ適用されたモジュールに入れ替えることは可能か？ソフトウェアやファームウェアは実装前に、サンドボックス内に隔離してセキュリティを検証することは可能か？

ファームウェアはエンドツーエンドで保護されなければならず、ライフサイクル全体について考慮しなければならない。例えば、最初に入れるファームウェアは、セキュアなプロセスを使用して、セキュアな施設にてロードされているか。アップデートセッションが中断するような場合の不測の事態を想定した設計を行っているか。つまり、機能停止させないように以前のバージョンへロールバックできるか。

同じように、更新プロセスと、これを実行する権限を対応付ける必要があることを理解しなければならない。この点については、更新を開始（実行）する責任は誰にあるか。製品のオーナーか、あるいはバックエンドのデバイスインフラか。ユーザーがタイミングよくアップデートできないことも有りがちで、潜在的に製品を脆弱性に晒す可能性があるため、時には IoT 製品を自動的にアップデートする方が良いこともある。 ただし、攻撃者が更新プロセスを使用してデバイスに対してサービス提供不能（DoS）攻撃を実行できるかどうかについても考慮する必要がある。

アップデートトランザクションの認証に関しては、ファームウェアの認証がエンドツーエンドであることを理解することが望ましい。このためには、デバイスがインフラストラクチャの内部のファームウェアに適用される署名を検証するために使用することができる、信頼を担保する(root of trust)ためのセキュアなストレージを持つことが必要となる。もちろんこれは、新たな懸念事項の原因となるので、インフラストラクチャの中でファームウェアをアップデートする署名を使用できる鍵はセキュアにしなければならない。このタスクには、HSM などのソリューションを活用する。そしてまた、アップデートサーバ自体を安全に設定することを確認し、開発ライフサイクルのどの時点でも、

誰かがファームウェアにアクセスし意図的な（あるいは意図しない）バグを混入させる可能性があることを忘れてはならない。このドキュメントで説明しているように、CI(継続的インテグレーション)環境を使用して、脆弱性およびよくある誤りをソフトウェアアップデートから取り除くこと（SAST：Static Application Security Testing）。

もう少し掘り下げてみると、ファームウェアにデジタル署名するために利用される暗号化スイートは徹底的に検査されているか、という点がある。これらの目的に使用される、FIPS 140-2 のような暗号ライブラリの実装のための認証を取得していることを確認すること。暗号化されたデータ送信を提供するためにこれらのツール（暗号化スイート）を活用すること。理想的にはファームウェアイメージも暗号化すること。これは、復号鍵のためのデバイス上のセキュアなストレージを前提とする。また、不正なファームウェア変更を防ぐために、製品の書き込み防止を忘れないようにする必要がある。

更新メカニズムの選択に関しては、**The Update Framework（TUF）**の採用も検討すること。

10. 認証、認可、アクセスコントロール機能の実装

IoT 製品開発において防御するための重要な考慮事項のひとつは、認証情報の盗難である。パスワード、アクセストークン、秘密鍵のいずれの盗難であっても、認証情報は防御されなければならず、さもなければアクセスを制限する能力が無効化される。このことは、開発者がセキュアなストレージを実装する方法に取り組まなければならないと同時に、デバイスを静止状態にすることでメモリーを覗き見られたりする場合にも認証情報が漏洩しないようにしなければならないことを意味する。本文書は、セキュアな認証情報の保管と利用の全過程に関するいくつかの推奨事項を提供する。

IoT における認証と認可に関する課題のひとつは、エンドユーザーが関与するデバイスの規模である。スマートホーム内の数十あるいは数百個のデバイスであれ、企業における数十万から数百万個のデバイスであれ、ユーザーに手動で各デバイスの設定を要求することは困難な命題である。デバイス間の通信が導入されると、問題はますます複雑になる。なぜならば、結果的に一緒に動作する数多くのデバイスが様々なベンダーの製品であり、さらに異なるフレームワーク上に存在することもあるためである。

IoT 製品の開発者として、今日の ID 管理、認証と認可プロトコル、システムは、IoT ソリューションにとって最適化されていないという声を聞くかもしれない。なぜこれが真実であるかを振り返って考えるならば、今日の仕組みはユーザーID をサポートするように作られていることに気付く。例えばスマートフォンのようなデバイスがサポートされる場合でも、通常はユーザーID と結び付けられている。IoT デバイスでは、デバイスの ID は、特定の個人ユーザーと結び付けられるかもしれないし、結び付けられないかもしれない（多分後者）。さらに開発者は、これらデバイスの管理（例えば、企業の技術者による）をサポートする能力を利用者に提供しなければならない。時には OEM（Original Equipment Manufacturer）元でさえ、ファームウェア／ソフトウェアの更新やデバイスの設定のために、デバイスに直接アクセスすることを必要とするかもしれない。

認証、認可、アクセス制御の機能を考えるとき、IoT 製品がいかにして使われるか管理されるかを理解しなければならない。消費者向け製品では、多くの場合、ユーザーのスマートフォンとデバイスとが直接的に信頼する関係にある。この場合、スマートフォンのアプリケーション（IoT app）は IoT 製品を認証すべきであり、理想的にはその逆も行われるべきである。しかしながら議論はそれで終わりではない。IoT の利点とは、デバイスがお互いに、望ましくは自動的な方法で通信できることにあ

る。これは（家庭の環境あるいはビジネスの環境で）付加価値のある IoT エコシステムにするために、IoT デバイスが他の IoT デバイスと信頼関係を構築できなければならないことを意味する。このデバイス間通信は安全に確立されなければならず、そうでなければネットワークへのエントリポイントを利用した悪意のある行為のリスクに晒される。

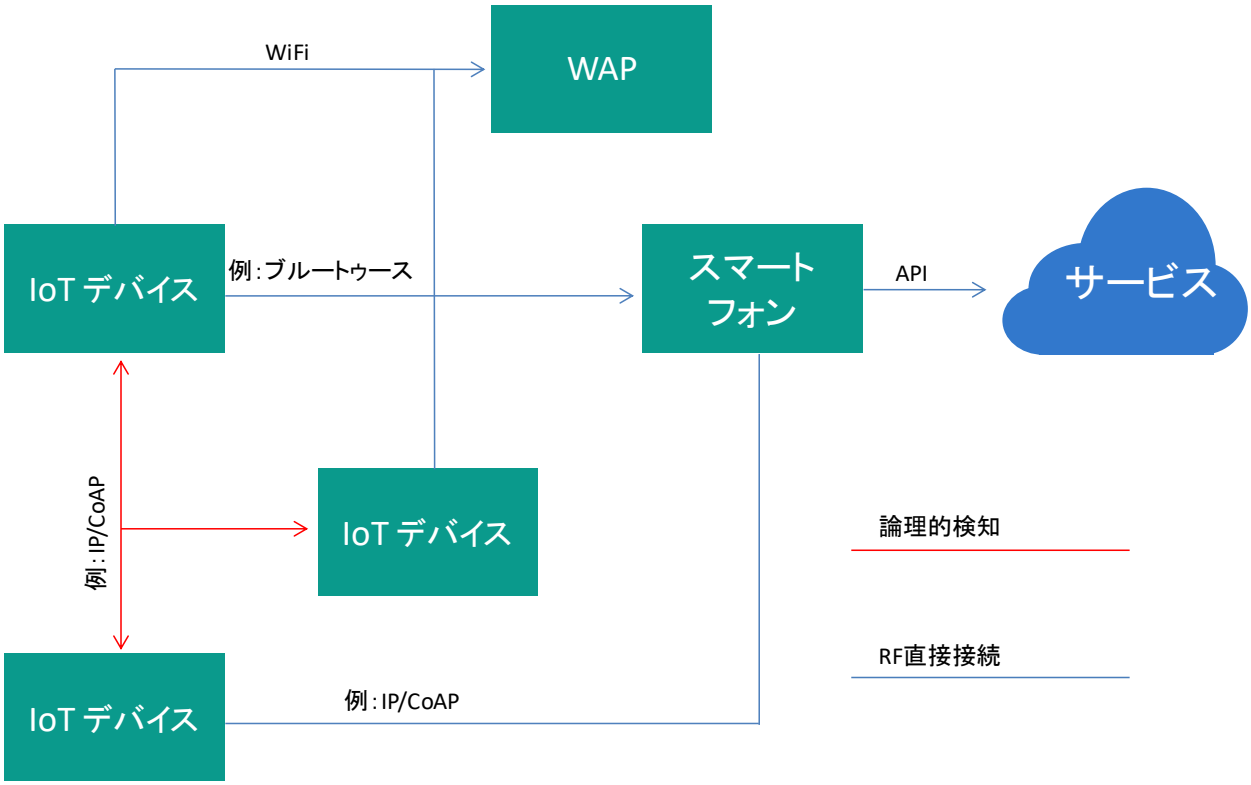


図 6

幸運にも、デバイス間通信をサポートする IoT プロトコルの多くは、セキュアな接続を設定する機能がある。表 12 に、これらプロトコルのいくつかで利用可能な設定オプションの一覧を示す。

表 12

プロトコル	M2M 認証オプション	分析
MQTT	ユーザー名//パスワード	MQTT はユーザー名とパスワードを伝送できるが、パスワードの長さは 12 文字までに制限される。ユーザー名とパスワードは平文で転送されるため、MQTT の利用においては、TLS が必須となる。
CoAP	事前共有鍵 公開鍵証明書	CoAP はデバイス間通信において複数の認証オプションをサポートする。データグラム TLS（D-TLS）と組み合わせることで、より高いセキュリティを確保できる。

XMPP	プロトコルに依存する、複数の認証オプションが利用可能	XMPP は SASL（Simple Authentication and Security Layer – RFC4422）を介して、様々な認証方式をサポートする。一方向の匿名認証や双方向の暗号化パスワードを使った相互認証、電子証明書、SASL の抽象化レイヤで実装可能なその他の方法がある。
DDS	X.509 証明書（PKI） トークン	エンドポイント認証に加え、その後実施されるメッセージデータの（HMAC 等を使用した）発信元認証のための鍵交換を提供する。電子証明書や、様々なタイプの ID・認可トークンがサポートされている。
HTTP/REST	ベーシック認証 （平文または TLS） OAUTH2	HTTP/REST は、一般に認証と機密性維持のために TLS プロトコルのサポートが必要である。ベーシック認証（クレデンシャルが平文で渡される）は TLS のもとで利用できるものの、これは推奨される方法ではない。代わりに、OAUTH2 のような、トークンベースの認証方式の利用を試みること。

認証による保護は、それがエンド間（end-to-end, E2E）であるときに最良の方法であることを覚えておくと良い。IoT は、直接的な接続を分割するゲートウェイに依存することが多いため、E2E の認証による保護は常に実現できるとは限らない。しかしながら通常は、スタック内で上位にいくほど（例えばメッセージレベル）、これら E2E 保護を確立しやすくなる。

たとえばネイティブなプロトコルの認証に加えて TLS の証明書ベースの認証で互いに多層化するような、複数の認証オプションを設計することは、エンド間のデータ保護を提供できる。

デバイスはまた、（仲介なく）直接的にクラウドサービスと通信するかもしれない。多くのクラウドサービスプロバイダ（CSP）は今や、MQTT と REST 通信をサポートするゲートウェイを提供している。デバイスとクラウド間の通信は多くの場合、デバイスと通信する特定のサービスのための API キーを利用する。これは MQTT や REST 通信のプロトコルに向けた、TLS 実装への追加である。そしてこれは、多くの場合双方向の証明書ベースの認証の実装を可能にする。

IoT 製品開発者が考慮すべき認証と認可の方法は複数ある。多くの場合、二要素認証をサポートすることは意味がある。強固な認証と認可の戦略を製品に取り込むかどうかは、それらデバイスが運用される中での脅威の状態の情報に大きく左右される。計画された認証／認可のプロセスは、その環境内で必要な保護レベルにとって十分であるか？デバイスに対する特定のどんな脅威を、認証／認可のスキームが緩和できるか？すべての IoT 製品についてこれらの質問に答えられるようにすること。以下では、認証と認可のための一般的なメカニズムに関する考察を提供する。

認証のための証明書の使用

証明書は、暗号鍵のペア（公開鍵／秘密鍵）に結び付けられる。公開鍵は証明書の中に直接的に埋め

込まれ、取引の真正性の確認を必要とする誰もが利用可能である。秘密鍵は保護され、認証のときにはオブジェクトをデジタル署名するのに使われる。オブジェクトに紐づいた署名の正当性を確認するために、対応する公開鍵が使われることで検証が行われる。

このアプローチを使ってメリットを得るために、証明書を発行するためにセキュアな公開鍵基盤（Public Key Infrastructure, PKI）が利用可能であることを、開発者は確実にしなければならない。鍵ペア自身は一般的に、IoT 製品上で生成されるべきであるが、PKI の証明書は受領した証明書署名要求（Certificate Signing Request, CSR）に基づいて発行される。（つまり、PKI の証明書は証明書署名要求（Certificate Signing Request, CSR）を送らなければ発行されない。）このプロセスを助けるプロトコルもあり、Simple Certificate Enrollment Protocol（SCEP）や Enrollment over Secure Transport（EST）プロトコルがある。

デバイスは、秘密鍵の所有と第三者機関（PKI）に対する信頼にもとづき認証される。

可能なら、双方向の証明書による認証の実装を検討すべきである。標準的な X.509 証明書は、プロトコル特有のセキュアなメッセージング用、および TLS 通信用の価値のある多層セキュリティを提供する。双方向の証明書による認証によって、通信対象のサービスあるいはデバイス／ゲートウェイが検証するための証明書を、IoT 製品が渡すことができる。これは特に、平文でユーザー名／パスワードを送ることだけをサポートする MQTT のようなプロトコルを扱うときに有用である。

認証のためのバイオメトリクス

バイオメトリクスもまた、認証トランザクションにおける識別の方法として利用できると考えられる。指紋あるいは声紋のようなバイオメトリクスは、認証の可否を決定するときに使われるエンティティ（人間）についての情報を渡すのに使うことができ、現に使われている。

証明書レス認証暗号（Certificate-Less Authenticated Encryption, CLAE）

IoT における認証を提供する比較的新しい方法として、証明書レス認証暗号（Certificate-Less Authenticated Encryption, CLAE）がある。CLAE は、メッセージを暗号化するまえに、メッセージの送信者がサーバーの公開パラメータを検証することができる。これにより、中央の認証局（Certificate Authority, CA）によって発行される証明書を使う代わりに、メッセージを暗号化するのに使う公開パラメータが改ざんされていないことを検証できる。以下は Connect in Private 社から提供されたイメージである。

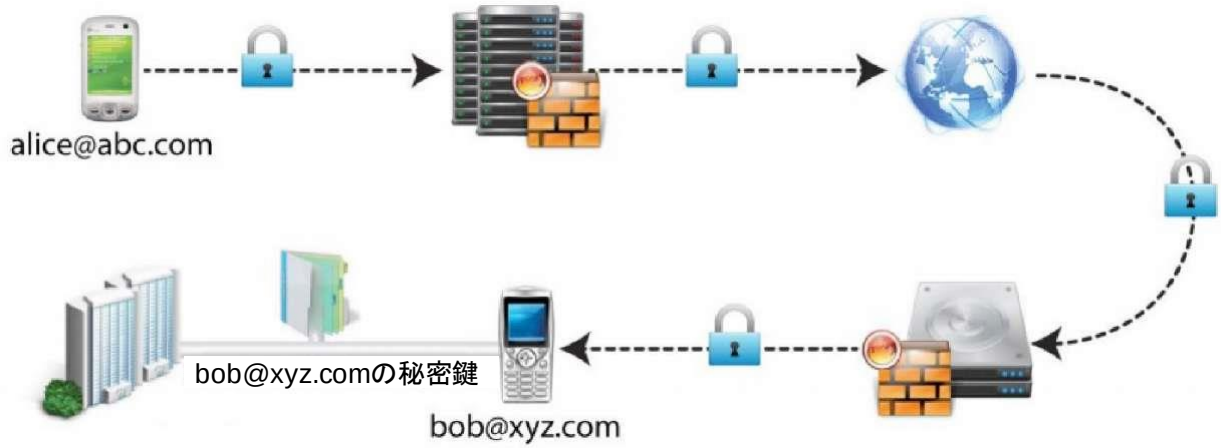


図 7

OAuth 2.0

OAuth 2.0 は、ユーザー/デバイスの ID を検証し、アクセスのためのトークンを発行する認可サーバーを必要とする。これは集中化されたメカニズムであり、IoT 製品が運用される環境から認可サーバーにアクセスできることが必要である。これはまた、サーバーが攻撃から保護されなければならないことを意味する。

OAuth 2.0 は現在、IoT のアクセス要件を満たすために使われている。例えば、あるアーキテクチャは相互に通信しなければならない数多くの多様な IoT デバイスを含むが、異なる IoT デバイスがリクエスタとリソースプロバイダとして振る舞う一方で、スマートハブあるいはクラウド上のサービスは OAuth 2.0 トランザクションのための認可サービスとして機能する。これらは、REST や MQTT のようなスキームでの TLS で保護された接続の上で行われる。

Amazon の Alexa とそのプラットフォーム用に生成できる Smart Skills は、良い例である。Amazon Smart Home Skill の API は、OAuth 2.0 認証を必要とする。この例では、サポートされているサービスの IoT 開発者は、認可サーバーと保護されたリソースとを用意する。（Smart Skills をホストする）Alexa サービスは、クライアントとして振る舞う。Figure 8 に簡略化したワークフローを示す。

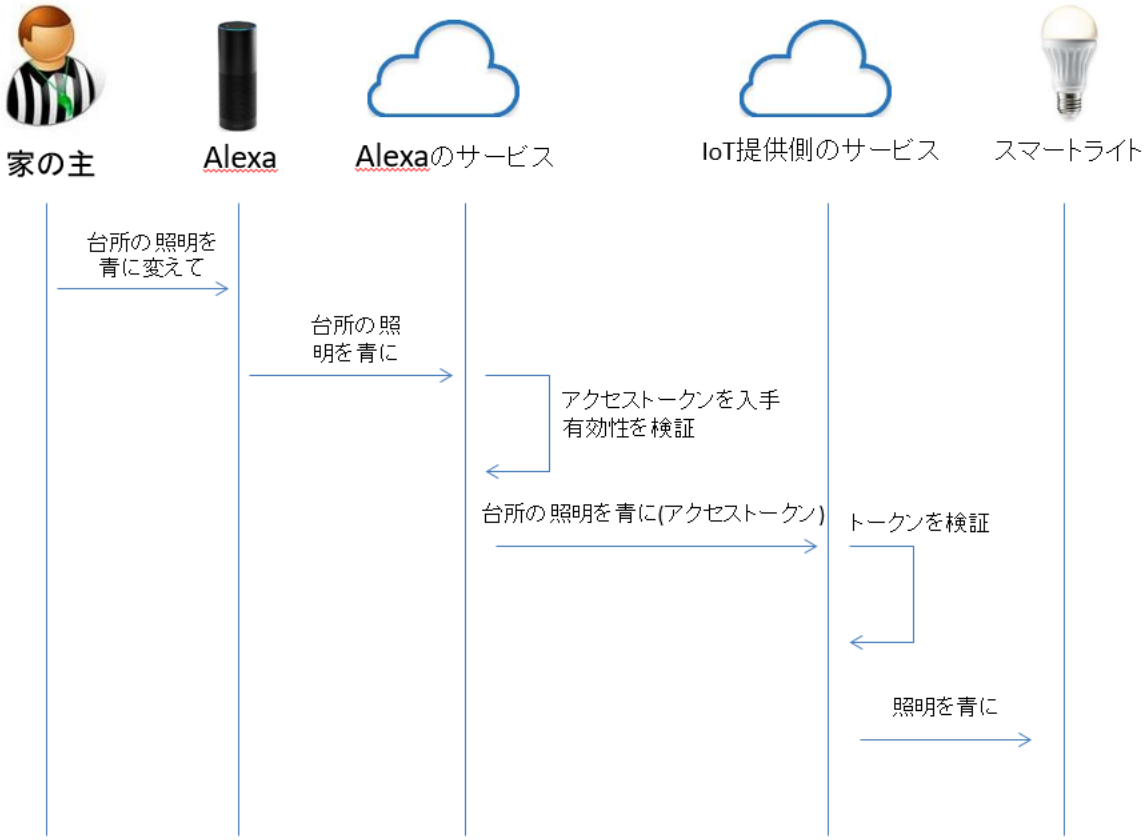


図 8

ユーザーは、IoT サービスに外から接続する機能を持った Alexa スマートフォンアプリを使うことで、Smart Skill と IoT アプリとの間を繋ぐことができる。Alexa の Smart Skill の認証と認可プロセスについては、ここを参照せよ。

注意しなければならないが、もしモバイルアプリケーションか高機能アプリケーションが OAuth 2.0 を使っているならば、クライアント認証を行う認可サーバーが、各アプリケーション内に配布されハードコードされた秘密鍵（client_secret）を使ってクライアント認証することは推奨されない。さらにすべてのトークンは、開発者がデバイス上のハードウェアセキュリティ機能を使えるようにする iOS キーチェーンや Android の Keystore を使って安全に保管されるべきである。インターネットドラフト「Best Current Practice: OAuth 2.0 for Native Apps」は、セキュアな iOS と Android の OAuth 2.0 実装の追加的なガイダンスを提供する。

User Managed Access (UMA)

どのアプリとどのデバイスがお互いにアクセスするかに関する複雑なポリシーをどうやって管理す

るか？これに対しては、ポリシー管理のために UMA を使うことができる。このサービスが物理的にどこに位置するかはまだ議論中であるが、無線 LAN アクセスポイントあるいは企業のゲートウェイに配置することができるだろう。

UUMA は、OAuth にもとづくアクセス管理プロトコルである（リンクを参照）。その標準は、Kantara initiative によって承認された。UMA により、利用者は情報の共有や取消の認可を行うことができる。この共有は、動的に構成することができる。UMA は、人がデバイス間のトランザクションに関わっているシナリオにおいて、もっとも高い価値を提供するように見える（例えば、リアルタイムで共有する、さらに重要なことは、共有するかしないかをリアルタイムで選択できるようにするという考え方において）。UMA は、プライバシーが大変重要で共有の環境が複雑なときに、検討すべき標準である。

11. セキュアな鍵管理システムの構築

鍵管理（key management）は暗号と同じような難解なテーマに属するもので、しばしば安全でない方法で実装されたり、全く実装されないこともある。暗号鍵が不正に公開されれば、暗号の使用は無意味なものとなる。暗号化するために使用される様々なタイプの変数が開示されてしまうと、暗号処理を実行した数年後であっても壊滅的なデータの漏洩を招く可能性がある。それゆえ鍵管理は、暗号鍵がどのように管理されるかについて扱うものであり、次の内容を含む。

表 13

鍵管理のフェーズ	説明
鍵生成（key generation）	鍵が、いつ、どのデバイス上で、どのようにして生成されたか。
鍵導出（key derivation）	他の鍵やその他の変数から暗号鍵を生成すること。
鍵確立（key establishment） 鍵共有（key agreement） 鍵配送（key transport）	二当事者がそれぞれ暗号鍵を生成するための計算アルゴリズム セキュアな鍵のラッピングと、デバイスから他のデバイスへの鍵の配送
鍵保管（key storage）	鍵のセキュアな保管方法（しばしば、別の暗号鍵を用いて鍵が暗号化される）、及び、保管されるデバイスの種類
鍵の有効期間（key lifetime）	鍵が破棄（ゼロ化）されるまでの期間
鍵のゼロ化（key zeroization）	鍵及び鍵材料（material）のセキュアな破棄方法
アカウンティング（accounting）	エンティティ間において、鍵及び鍵材料の生成・配送・破棄について、特定・追跡・記録をすること。

一般的に、導入されている公開鍵基盤（Public Key Infrastructures）に伴う技術やプロセスはセキュアな鍵管理をベースとするものであるが、重要なのは、必ずしもすべての IoT デバイスが PKI の証明書を認識し、利用できるわけではないために、鍵管理の基本を押さえた適切な実装に立ち返る必要がある、ということである。多くの場合、例えば、対称鍵暗号及び認証鍵の使用にとどまる。IoT に関しては、セキュリティと性能をバランスさせることが重要である。この問題は、暗号鍵の有効期間において顕著となる。一般的に、鍵が漏えいした場合、有効期間が短い方が影響が小さ

い。しかし、鍵の有効期間が短ければ短いほど、鍵の生成や確立、あるいは記録を頻繁に行わなければならない。

鍵の有効期間が経過した場合、以下のとおり、様々な方法によって、新しい鍵が提供される。

1．事業者の鍵管理ソフトウェアを用いて、中央鍵管理サーバーから鍵を送付又は取得する。
2．新たなソフトウェア又はファームウェアの中に安全に組み込まれる。
3．デバイス自身によって鍵を生成する。
4．他のデバイスによって鍵を確立する。

セキュアな鍵管理においては、ベンダーが特にデバイスの製造・配送プロセスにおける鍵の階層に精通していることも必要である。鍵材料を予め組み込んだデバイスが、製造業者から出荷されるが（この場合、製造業者はこれらの鍵の保護について十分注意を払わなければならない)、おそらくエンドユーザーによって上書きされ、使用され、または廃棄され得る。また、それぞれの鍵が、デバイスを新たな状態に遷移させるための、又は、フィールドに配備する際のブートプロセスにおいて必須条件となっていることもある。メーカーは、各製品を安全に配備するために用いられる鍵管理プロセス・手順・システムについて十分に文書化しなければならない。さらに、メーカー自身はその鍵を、できればオフラインのシステム、かつ、安全が保たれた設備内で保管しなければならない。

暗号鍵がたった一つ消失・漏えいした場合であっても、その派生的影響が大きいことを考慮するならば、鍵管理システムへのアクセスコントロールは、厳しく制限されるべきである。

表 14

鍵管理		
#	質問	回答
1	鍵はセキュアに保管されているか。	セキュアな保管が行われているということは、鍵が物理的にも（理想的には）暗号技術的にも保護されているということを意味する。
2	鍵は、使用後又は有効期間経過後にどのように廃棄されているか。	平文の暗号鍵を持つ変数は、使用後に確実に廃棄されない限り、長期間にわたってメモリー上に残存するということが知られている。良い暗号ライブラリでは、必要な場合に鍵を自動的に消去することもあるが、一方で、鍵消去サービスの発動をアプリケーションに委ねるライブラリもある。ポリシー又は技術コントロールを通じて、必要な場合に鍵を消去するという方針もしくは技術的な管理策を伴うセキュリティ・ポリシーを確立することが重要である。例えば、TLS セッションキーは、当該セッションの終了後に直ちに消去されるべきである。

3	どのような鍵長が使用されているか。	適切に構築されたアルゴリズムにおいては、鍵長は、暗号解読攻撃や総当たり攻撃（brute forcing attacks）への耐性と関連づけられることも多い。一般に、米国政府等によって採用されているベストプラクティスを遵守することが強く推奨される。暗号アルゴリズムが脆弱であることが判明した場合、その鍵長の議論も終了する。NIST Special publication 800-131 は、当面推奨されるアルゴリズム及び鍵長に関し、参考になるガイダンスを提供している。
4	乱数供給源は十分にランダムか。	IoT デバイスが暗号鍵やそのシードを生成する場合、乱数生成器（又はランダム・ビット・ジェネレーター）が、(1) 適切に設計されたものであり（例: ANSI X9.31 or NIST SP 800-90）、かつ、(2) 良質な乱数供給源から適切にランダムな値が供給されていることが不可欠である。デバイスにおいては、電気回路のランダムノイズをデジタル化したものや OS におけるランダムに発生するイベント等の、様々な供給源を用いることが可能である。IoT デバイスを設計する場合、当該デバイスの乱数生成器の乱数供給源において min（ミニマム）エントロピー解析を行うべきである。
5	証明書はどのように検証されるか。	証明書が明確に信頼できること、あるいはより一般的には、証明書チェーンの連鎖（信頼の連鎖）によって信頼できることを確実にすべきである。
6	証明書はどのように失効、もしくは期限切れするか。	デバイスがおかしな動きをしたり侵害されたりした場合、典型的な PKI と同様に、証明書を失効させる機能を備えていることが不可欠である。さらに、デジタル証明書（例: X509）の場合、鍵の有効期間が当該証明書に記載されている。デバイスが他のデバイスを信頼して作動する場合、他のデバイスの証明書の有効期限を確認し、期限切れした証明書を信頼しないようにプログラムされ、設定されていることが重要である。 さらに、暗号鍵（例: 証明書）の有効期間が過度に長く、使用されている暗号アルゴリズムや鍵長の推奨有効期間を超過したりしている場合は注意が必要である。例えば、今日においても多くの SHA-1 ベースの証明書が未だに有効であるが、SHA-1 アルゴリズムは脆弱性を抱えており、（SHA-2 などへの）段階的移行が図られている。

7	誰が鍵管理システムへのアクセス権限を有しているか。そのアクセス方法はどのようなものか。	暗号鍵管理システムへのアクセスは、物理的かつ論理的に厳重に制限されていなければならない。物理的なアクセスコントロールには、保護された建物と、入室を制限された部屋（又はケージ）が重要である。 また、管理者及びユーザーによるアクセスも、職務の分離（アプリケーションとホストシステム・サービス）及び複数人の操作による統制（センシティブなサービスを起動するにあたって複数の担当者の許可を必要とすること）の見地から、注意深く管理されなければならない。
8	PFS（Perfect Forward Secrecy）を採用しているか。	「PFS（Perfect Forward Secrecy）」は、多くの暗号鍵交換及び鍵確立プロトコルにおいて提供されている。これにより、一組のセッションキーが破られても、次に生成されるセッションキーには影響が及ばないため、PFSの採用が望ましい。例えば、Diffie-Hellman（DH）又は楕円曲線Diffie-Hellman（ECDH）においては、PFSスキームによって、一度限りのDH/ECDHキーが生成され、共有シークレットとその最終的な暗号鍵が常にセッションごとに固有なものとなる。
9	どの鍵管理プロトコルを使用しているか。	この質問は今後の課題を含むものである。なぜなら、現時点で入手可能な鍵管理プロトコルは比較的数少ないうえ、その多くが特定の企業の固有のソリューションに組み込まれているからである。しかし、業界は、OASISのグループと連携して、KMIP（Key Management Interoperability Protocol）を、送信者/受信者間の鍵管理・交換のためのシンプルなプロトコルとして作成し、採用し始めている。このプロトコルは、数多くの暗号化アルゴリズムをサポートし、マルチベンダー間の相互運用の課題を解決するために設計されたものである。KMIPは、様々なプログラミング言語で使用することができ、いかなる種類のアプリケーションレイヤーや管理レイヤーのプロトコルにおいても、その下で使用することができる。

セキュアなブートストラップ（接続初期化機能）の設計

セキュアなブートストラップ（ネットワークやサービスへの初期接続）処理は、IoT デバイスの様々なセキュリティ機能の基礎である。ブートストラップ処理により、様々なシステム及び他のデバイスとの認証をサポートする初期の証明書を提供する。

多くの場合において、開発者は、IoTデバイスに事前に証明書をインストールしており、それによって当該IoTデバイスはセキュアに新しいシステムに接続できる。例えば、インストールされたメーカーの証明書を利用して、実際に使用する証明書のためのCSR（証明書署名要求; certificate signing request）に署名する。こうした場合の適切な対処として、追加の確認手続きをプロセスに組み込むこと、例えばデバイスを新たなPKIに登録する間に管理者によりデバイスの固有情報を保証することなどがある。

12. ログ機能の提供

企業をセキュアに保つための最も重要な事項の 1 つが、どのような(セキュリティ)イベントが企業に起きているかを知ることである。ユーザーがデバイスやサービスにログインしたこと（特にログインの失敗)を知ることができることはセキュリティの管理者にとって重要である。IoT 開発者として、お客様がそのセキュリティ方針上注視する必要がある情報を提供しなければならない.

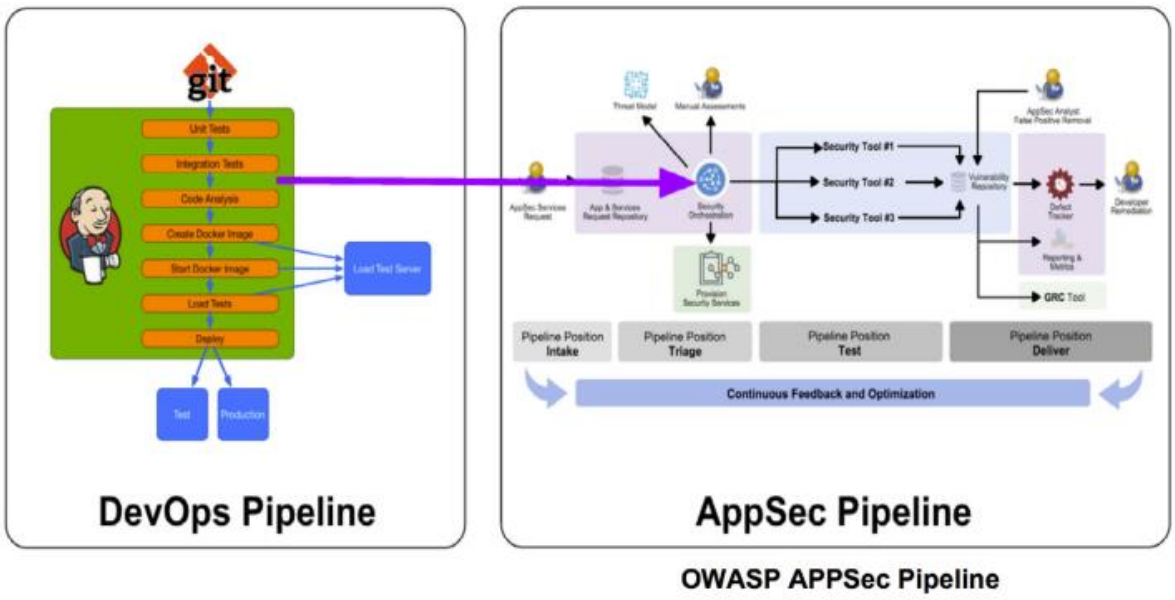
IT の装置は多くの場合、syslog をロギング機能として用いている。そのことによって管理者は、デバイスからのイベントデータに容易にアクセスでき、それを強力な SIEM(Security Information Event Management System)に相関分析のために転送できる。IoT においては、多くの機器が一定レベルのログ情報を、REST API を介して提供している。顧客にログデータを提供する必要を満たすために様々な方法があるが、企業ユーザー向けの IoT デバイスは、そのデバイス上で起きている事象に対する十分な可視性を提供することが重要である。ログには少なくとも以下の内容が必要である。

- 接続要求
- 認証(失敗／成功)
- 特権の悪用の試み、権限昇格の試み
- 不正なフォーマットでのメッセージ受信
- ファームウェアやソフトウェアの更新の成功や失敗
- ローカルログインの試み
- 設定の変更
- アカウントの更新
- プロテクトされたメモリーへのアクセス
- 物理的な改変

13. セキュリティレビューの実施
(内部レビューと外部レビュー)

OWASP(Open Web Application Security Project)は、セキュアなソフトウェアの開発についての、ソフトウェアから見た課題、および設計・開発・テストを統合するフィードバックループの使い方を理解するための素晴らしいツールを提供している。しかし IoT において、ソフトウェアは作業（同様にレビューすべきハードウェアを含む）の一部であることを忘れてはいけない。 *（ISO/IEC 15408 で定義しているセキュリティ評価基準の）*コモンクライテリアのようなアプローチは、製品におけるハードウェアベースのセキュリティ機能を構築する際の参照先として利用できる。

フィードバックループにより、テスト環境やフィールドでの脆弱性の特定に基づく設計の改良が可能になる。図 9 は AppSec パイプラインに関する OWASP の解釈である。



OWASP APPSec Pipeline

図 9

OWASPのAppSec Pipelineは、IoT製品のライフサイクルにわたる継続的なフィードバックと最適化が必要であることを示している。発見された不良や脆弱性は、設計および脅威モデルプロセスにフィードバックされる必要性があり、その結果ハードウェアとソフトウェアのベースラインがアップデートされ、さらにその後にパッチが新しい脆弱性を持ち込ん

でいないことを確認するために再テストしなければならない。IoTデバイスのセキュリティテストプロセスは、このような脆弱性を発見するように設計されている。
IoT機器の開発に採用でき、機器のセキュリティ特性を維持するために重要な役割を持つ多くのテストがある。

1. 静的アプリケーションセキュリティテスト(SAST：Static Application Security Testing)
2. 動的アプリケーションセキュリティテスト(DAST：Dynamic Application Security Testing)
3. 複合的なアプリケーションセキュリティテスト(IAST：Interactive Application Security Testing)
4. 攻撃対象と攻撃ベクターに基づくテスト
5. サードパーティー製開発ライブラリの検証
6. ファジング
7. 攻撃ベクターごとにカスタマイズしたテスト

IoT 機器製造元はセキュリティの評価を行う為に、第三者の評価機関を利用すべきである。有償のサービスの他に、ボランティアの研究者によるハードウェアテストサービスもある。企業によっては社内に同様の分析を行うことのできる評価チームを持っている。いずれの方法を選択するにせよ、評価結果のフィードバックは、デバイスのセキュリティ特性に関する価値ある観察を提供し、必要な対策を特定することができるようにする。第三者による評価を提供する専門家の組織の例として、"builditsecure.ly"のようなコミュニティや"crowdstrike"のような企業がある。

ICSA Labs もまた IoT 機器のテスト用の評価プログラムを開発している。ICSA は特に 6 項目のテストにフォーカスしている。

- コミュニケーション
- アラート／ログ
- プラットフォームのセキュリティ
- 暗号
- 物理セキュリティ
- 認証

ICSA labs の要求項目はこちらを参照。

UL(Underwriters Laboratory)も同様に、新たな IoT 機器向けのソフトウェアとセキュリティの評価サービスを開発した。UL の評価サービスについてはこちらを参照。.

Appendix A - IoT 機器の分類

IoT 機器開発に固有の脅威の特定を支援するために、このセクションでは、さまざまなタイプの IoT 機器を分類し、これらのデバイスが配備される代表的な環境の例を示し、各実装における典型的な脅威について概説する。このセクションでは、次のタイプの IoT 機器の例を示す。

1. 消費者向け
2. 医療・ヘルスケア向け
3. 産業向け
4. スマートシティ向け

表 15 では、本書で議論されている様々な脅威主体の定義を提供する。

表 15

脅威主体	説明
国家	セキュリティインシデントを（直接的/間接的に）引き起こす敵国にとっての動機は、金銭的利益、知的財産へのアクセス、政治的な優位性の獲得、または重大な情報システムの損害を引き起こすことである。
サイバーテロリスト	イデオロギー的または政治的な目標を持って不安やパニックを引き起こすように設計された攻撃を実行する。一般に、これらの脅威主体は、既知のテロ組織の一部である。
ハクティビスト	言論の自由や人権などの政治的な事柄に注意を引きつけたり、主義，主張への支持を妨げたりするために攻撃を行う者。政治的信条に基づいている。
ハッカー	コンピュータを使用して、データに不正アクセスする人 .
犯罪組織	違法行為を行う犯罪者のグループであり、活動は金銭的な欲求に基づいている。攻撃は、被害者から金銭を奪い取るか、あるいはそのような攻撃をすることによって報酬を得る。

個人	単独で行動する特定の個人またはグループで、他のグループまたは関連団体とは関係しない。他のカテゴリには該当しない。 • 愉快犯 • 内部者/システムユーザー：権限のあるユーザーが、自らの認証情報を用いて権限の無いデータにアクセスする。 • 窃盗犯
インサイダー／システムユーザー	権限のあるユーザー。自身の認証情報を権限外のデータへのアクセスに利用。

消費者向け IoT 機器

低コストかつ簡単に手に入ることから、消費者が利用する IoT デバイスは、セキュリティコミュニティから大きな注目を集めている。このクラスの IoT デバイスは企業のセキュリティ特性に影響を与えていないようにみえるが、最近の OpenDNS の報告によると、これらのタイプのデバイスが企業環境内で使用されており、企業ネットワークに配置されることが多い。これらのタイプのデバイスの十分なセキュリティ特性を維持することは、誰にとっても有益である。

.

表 16 は、さまざまなタイプの消費者向け IoT 機器、およびそれらが動作する環境のリストを提供する。

表 16

IoT 機器のタイプ	例	主な環境	主な脅威
ウェアラブル	フィットネス/ライフスタイル/妊活支援/スマートウォッチ	屋内/屋外 保健施設 ジム 事務所	•アクセスできたことを自慢しようとする個人 ・誤報に基づく危険な薬の摂取/生活習慣の変化 •アドレス帳の全情報とスマートウォッチのデータの漏洩 • 脆弱性によって位置情報がストーカーにアクセスされる可能性

スマートホーム	照明・サーモスタット・ドアロック・ガレージドアの遠隔操作・プールのポンプ・スプリンクラー・冷蔵庫・掃除機・テレビ・食洗機・洗濯機・音響システム・幼児モニター・監視カメラ・音声モニター・呼び鈴	宅内 ホテル レストラン 事務所 病院/医療施設 乳幼児施設	・心配や混乱を引き起こす愉快犯 ・アクセスできたことを自慢しようとする個人 ・モニターを利用してネットワークに侵入を試みるハッカー ・IoT 機器を誤作動させて、電気・ガス・水道などを浪費させ、金銭的被害をもたらす ・アラームを作動させることによって、生活パターンや在宅状況を把握する窃盗犯

消費者向け IoT 機器が直面する典型的な脅威は上記のようなものである。しかし家庭環境と企業環境の両方で接続性が向上していることから、標的型攻撃により家庭内ネットワークへ侵入し、企業を攻撃するための有用な情報を得るケースが出てくるであろう。また、さまざまな金融犯罪に悪用できる情報を収集する方法として、犯罪者が消費者向け IoT の実装で見られる弱点を標的にし始める可能性がある。このことを考慮すると、消費者向け IoT 機器メーカーは、製品のセキュリティ特性を真剣に受け止め、最新の技術を用いて製品の使い勝手と安全性を両立させるという課題を解決することが重要である。

スマートヘルス機器

IoT から最も利益を受ける産業の 1 つは医療である。医療機器は、オフィスやクラウドのコンピュータにワイヤレスで接続するように変化している。製薬会社は、錠剤容器を IoT 対応にし、研究者は、埋め込み型診断ツールの開発に余念がない。スマートヘルス機器が提供する機能性を考えれば、これらの製品を安全に開発することは、患者を被害から守るためには不可欠である。

表 17 は、さまざまなタイプのスマートヘルス機器およびそれらの動作する環境のリストを提供する。

表 17

IoT 機器のタイプ	例	主な環境	主な脅威
医療機器	輸液ポンプ 血糖値モニター 乳児モニター	家庭 病院 外来施設	• 患者に危害を与えようとする犯罪組織 • 医療情報の漏洩

自己診断機器	Diabeto（糖尿病管理） Azoi（移動式ヘルスモニタリング装置）	家庭 外来施設	• 機器のソフトウェアあるいはハードウェアの不具合による信頼性の低い診断結果 • VIP 患者に危害を与えようとする犯罪組織
医療器具	ワイヤレス聴診器 体温計	家庭 病院 外来施設	• 機器のソフトウェアあるいはハードウェアの不具合による信頼性の低い診断結果
埋め込み型	ペースメーカー 呑み込み型カメラ	身体	• 生命にかかわる事態 • 機器に虚偽の情報を示して有害な判断をさせることによって、人命を危機にさらす
補助機器	スマートピルボックス	家庭用薬	• 人体への未知の悪影響

スマートヘルス機器はしばしば様々な法的規制を遵守しなければならない。例えば、米国では、デバイスおよび多くのデバイスの更新は、販売される前に FDA によってレビューされなければならない。スマートヘルス製品を開発する組織は、以下のガイダンスを検討する必要がある。

- **Cybersecurity for Medical Devices and Hospital Networks: FDA Safety Communication**
- **Medical Devices: Digital Health**
- **ISO14971 Application of Risk Management for Medical Devices**

産業用 IoT 機器

産業用ユースケースは、機械または電子デバイス間の自動データ伝送および測定を支援するために長い間使用されてきた。これらは、産業部門において重要な役割を果たし続けている。生産性の向上と安全性の向上という名目のために、さまざまなウェアラブル機器と産業用ビーコンが工場に組み込まれている。

表 18 にいくつかの実装例と主な脅威のリストを示す。

表 18

IoT 機器のタイプ	例	主な環境	主な脅威
SCADA システム	製造業 水道・電力・ガス 原子力 プログラマブルロジックコントローラ（PLC） リモートターミナルユニット（RTU）	工場(物理的施設)	• サイバーまたは環境テロリストが物理的な破壊または危害を引き起こそうとしている。 • ・操業を妨害しようとする愉快犯。 • ・雇用主に仕返しをしようとする内部者。 • ・国家による大量破壊の試み。 • ・雇用主に仕返しをしようとする内部者。

			• ・購入した部品に関連するサプライチェーンを通じた攻撃。
産業用無人航空機システム (UAS)	小規模な UAS 配送システム モニタリング・監視	遠い場所や到達するのが困難な場所	• 配達物やドローンそのものを盗もうとする窃盗犯 • サイバーテロリストによる妨害や危害(消火活動用の固定翼機を用いて人混みに突っ込ませたり、あるいは、そういったことを実行する) • 制御を奪取して楽しむ愉快犯
鉱業	自動掘削リグ 自動運搬トラック 圧力センサー 接近センサー 温度計 モーター ポンプ ジャイロスコープ	工場 (物理的施設)	• 物理的破壊や危害を試みるサイバーテロリスト • 操業を妨害しようとする愉快犯 • 雇用主に仕返しをしようとする内部者
製造業	ロボット		• 物理的破壊や危害を試みるサイバーテロリスト • 操業を妨害しようとする愉快犯 • 雇用主に仕返しをしようとする内部者

以下のガイダンスは、ICS（産業用制御システム）ベースの IoT 製品を開発する組織によって参照されるべきである。

- **Recommended Practice: Improving Industrial Control Systems Cybersecurity with Defense-In-Depth Strategies**
- **Guide to Industrial Control Systems (ICS) Security**

スマートシティ

一般向けの IoT システムとデバイスを、スマートコミュニティにおけるさまざまな機能をサポートする機器として分類する。

表 19

IoT 機器のタイプ	例	主な環境	主な脅威
スマートシティのインフラとサービス	信号機 環境センサー デジタル広告版 スマートコミュニケーション（スマートアンテナ） 高齢者の監視 スマートホスピタリティ（ホテルの部屋ロック） 危機管理	物理的に露出、 地理的に分散	• 物理的破壊や危害を試みるサイバーテロリスト • 操業を妨害しようとする愉快犯 • 雇用主に仕返しをしようとする内部者

	環境モニタリング		
スマートトランスポーテーション	コネクテッドカー 自動運転車 路側設備 交通管制センター 歩行者警報装置	物理的に露出、 地理的に分散、 移動する場合もある	• 雇用主に仕返しをしようとする内部者 • 物理的破壊や危害を試みるサイバーテロリ • 操業を妨害しようとする愉快犯 • 雇用主に仕返しをしようとする内部者

Appendix B -参考情報

[1]. http://www.securityweek.com/serious-security-flaws-found-hospira-lifecare-drug-pumps

[2]. http://www.devttys0.com/2012/11/reverse-engineering-serial-ports/

[3]. **Y.2060. Overview of the Internet of things. International Telecommunications Union (ITU-T), 6/2012. Available at https://www.itu.int/rec/T-REC-Y.2060-201206-I**

[4]. https://www.rapid7.com/docs/Hacking-IoT-A-Case-Study-on-Baby-Monitor-Exposures-and-Vulnerabilities.pdf http://devops.com/2014/10/13/devops-iot/

[5]. http://www.itproportal.com/2015/05/05/what-internet-of-things-means-for-devops/

[6]. http://theagileadmin.com/what-is-devops/

[7]. http://techcrunch.com/2015/05/15/what-is-devops/#.2rzef7r:5XYN

[8]. http://www.informationweek.com/mobile/mobile-applications/11-iot-programming-languages-worth- knowing/d/d-id/1319375?image_number=1

[9]. http://www.emdt.co.uk/daily-buzz/how-deal-it-security-threats-connected-medical-devices]http://www.adlawbyrequest.com/wp-content/uploads/sites/491/2015/01/IOT-Report-Lah-1.29.15.pdf

[10]. http://www.ti.com/lit/wp/slay041/slay041.pdf

[11]. http://micrium.com/iot/iot-rtos/

[12]. H. Ning and H. Liu, "Cyber-Physical-Social Based Security Architecture for Future Internet of Things", Advances in Internet of Things, vol. 02, no. 01, pp. 1-7, 2012.

[13]. PubNub, "A New Approach to IoT Security", 2015.

[14]. Wind River, "SECURITY IN THE INTERNET OF THINGS", 2015.

Appendix C - IoTの標準化団体

団体	説明
AIOTI	IoTイノベーションのためのアライアンス（AIOTI）は、標準化ポリシーを含む、欧州におけるIoTエコシステムの開発を支援するために、欧州委員会が立ち上げた。 https://ec.europa.eu/digital-agenda/en/alliance-internet-things-innovation-aioti
AllSeen Alliance	180のメンバからなる業界グループであるAllSeen Allianceは、IoTのデバイスとアプリケーション向けに、「AllJoyn」に基づく相互接続可能な通信フレームワークの広範な普及を促進する団体である。 https://allseenalliance.org/
Cloud Security Alliance (CSA)	（CSAの）IoTワーキンググループはITU-T Y.2060の仕様の拡張・構築に取り組んでおり、ITUや他の標準がカバーする、クラウド上のシステムとエッジデバイスの間を接続するための諸要素について検討している。 https://cloudsecurityalliance.org/group/internet-of-things/
ETSI	ETSIのConnecting Thingsは、何十億というスマートオブジェクトが通信ネットワークに接続できるためのデータセキュリティ、データマネジメント、データ転送とデータ処理に関する標準を開発している。 http://www.etsi.org/technologies-clusters/clusters/connecting-things
IEEE (including P2413)	IEEEには、IoT技術の調査、実装、応用、利用に関与する技術コミュニティのための、専門のIoTイニシアチブと情報センター（クリアリングハウス）がある。 http://iot.ieee.org/
IERC	IoTに関する欧州調査機関（IERC）は、欧州全域を対象に、IoT分野で取り組まれている活動の調整に当たる組織である。 http://www.internet-of-things-research.eu/
Internet Engineering Task Force (IETF)	インターネットにおける第一の標準策定機関であるIETFはIoT理事会を持ち、傘下のワーキンググループ間の関連する取組みの調整、首尾一貫性のための仕様のレビュー、他の標準化団体におけるIoT関連の活動のモニタリングを行っている。 https://trac.tools.ietf.org/area/int/trac/wiki/IOTDirWiki

IIC	産業インターネットコンソーシアム（IIC）は、OIC と協力し、産業向けの IoT アーキテクチャフレームワークの提供を促進する。IIC は IoT のためのリファレンスアーキテクチャを 2015 年に公表した。 http://www.industrialinternetconsortium.org/
Internet Governance Forum	IGF（インターネットガバナンスフォーラム）による IoT に関する動的な連合（Dynamic Coalition on IoT）は、IoT の運用に関して焦点を当てなければならないグローバルな課題を議論するためのオープンなミーティングの場を提供する。 http://www.intgovforum.org/cms/component/content/article?id=1217:dynamic-coalition-on-the-internet-ofthings
Internet of Things Consortium	この業界団体（IoT コンソーシアム）は、IoT 製品とサービスの採用を推進することを目的として、消費者調査とマーケット教育を行っている。 http://iofthings.org/#home
IoT Security Foundation (IoTSF)	本グループは純粋に、IoT セキュリティの標準の構築に取り組んでいる。 https://iotsecurityfoundation.org/
IP for Smart Objects (IPSO) Alliance	IPSO は、IoT の実用化のために、接続するスマートオブジェクトの基礎となる IP（知的財産）の確立に向けて、教育、調査、販促に取り組んでいる。 http://www.ipso-alliance.org/
ISO/IEC JTC-1/SC 27	ISO は、2014 年に IoT とスマートシティの予備調査報告書を発行した。このグループ（訳注：SC27）では、各々の領域に関する小委員会が継続中である。(ISO/IEC JTC 1/SC 27 10) http://isotc.iso.org/livelink/livelink/open/jtc1sc27 http://www.iso.org/iso/internet_of_things_report-jtc1.pdf
ISOC's Internet of Food SIG	本 SIG（Special Interest Group）は、食品産業で将来必要となるインフラに関する技術標準についての議論を主導している。 http://internet-of-food.org/
ITU	ITU では、IoT グローバル標準イニシアチブが取り組まれ、2015 年 7 月にその活動を終えたが、それを受け継いで IoT アプリケーションに焦点を当てた Study Group 20 が形成されている。 http://www.itu.int/en/ITU-T/studygroups/2017-2020/20/Pages/default.aspx （翻訳者注：リンク切れだったため、リンクをアップデートした）
MAPI Foundation	生産性とイノベーションのための製造業者アライアンス（MAPI）は、IoT を産業に適用するための「Industrie 4.0」を開発している。 https://www.mapi.net/forecasts-data/internet-things-industrie-40-vs-industrial-internet （翻訳者注：リンク切れだったため、リンクをアップデートした）

OASIS	OASIS は、IoT の相互接続を確実にするためのオープンなプロトコルを開発している。本グループは、IoT のためのメッセージングプロトコルとして Message Queuing Telemetry Transport (MQTT)を選択し、無線センサーネットワーク向けに MQTTS-N を最適化した。OASIS は、IoT で 3 つの技術委員会を持ち、MQTT の他に 2 つの標準：Advanced Message Queuing Protocol (AMQP) と OASIS Open Building Information Exchange (oBIX)を管轄している。 https://www.oasis-open.org/committees/tc_cat.php?cat=iot
oneM2M	M2M 通信アーキテクチャと標準を開発することを目指し、マルチベンダーからなる本グループは、遠隔診療、産業オートメーション、ホームオートメーションに取り組んでいる。その目指すところは、ハードウェアとソフトウェアに埋め込むことのできる、共通の M2M サービスレイヤである。 http://www.onem2m.org/
Online Trust Alliance	セキュリティベンダからなる本グループは、セキュリティ、プライバシ、サスティナビリティに焦点を当てた、IoT アプリケーションのための信頼フレームワークのドラフトを作成した。 https://otalliance.org/initiatives/internet-things
Open Interconnection Consortium	OIC は、業界標準にもとづいて、IoT デバイス間の情報フローの無線接続と管理のための共通通信フレームワークの策定に取り組んでいる。OIC は、デバイス間の接続性のためのオープンソースソフトウェアのフレームワークである IoTivity プロジェクトを支援している。 http://openinterconnect.org/
The Open Management Group	本技術標準コンソーシアムは、データ配布サービス（DDS）やインタラクションフローモデリング言語（IFML）など、いくつかの IoT 標準を作成している。これらはリアルタイムおよび組み込みシステムのための、ディペンダビリティフレームワーク、脅威モデリング、統合コンポーネントモデルに沿ったものである。 http://www.omg.org/hot-topics/iot-standards.htm
Open Web Application Security Project	OWASP の中の IoT トップ 10 プロジェクト[2]では、IoT における攻撃対象となる最重要な領域をリスト化することで、製造業者、開発者、消費者がセキュリティに関連する問題を理解できるよう支援している。 https://www.owasp.org/index.php/OWASP_Internet_of_Things_Top_Ten_Project
Smart Grid Interoperability Panel	SGIP は、エネルギー業界での IoT の新たな活用機会に焦点をあてた、EnergyIoT という取組みを行っている。このグループの OpenFMB は、共通ユーティリティデータモデルと IoT 通信プロトコルを統合して「Open Field Message Bus」を開発するための、ライフライン事業者主導のプロジェクトである。

[2] このプロジェクトは 2017 年 5 月時点では Internet of Things プロジェクトに統合されています

	http://sgip.org/
Thread Group	スマートホームベンダからなる本グループは、家電製品、照明、セキュリティシステムなど、家庭における IP 接続デバイスをサポートするための、共通のネットワークプロトコルを開発している。 http://threadgroup.org/About.aspx
W3C Web of Things (WoT)	欧州の FP7 プロジェクトがサポートするプロジェクトであり、IoT と Web データを結合するために Web 技術が果たす役割に関して、アプリケーションとサービスのオープンな市場に求められる要件の抽出とユースケースの収集に取り組んでいる。 https://www.w3.org/WoT/
Securing Smart Cities	Securing Smart Cities は非営利の国際的活動であり、世界中の企業、政府、メディア、他の非営利団体および個人との協力を通じて、スマートシティの現状および将来のサイバーセキュリティ問題を解決することを目指している。
UL2900	UL のサイバーセキュリティ保証プログラム（CAP）は、安心を提供する。CAP 認証は、意図しないあるいは不正なアクセス、変更、あるいは破壊を引き起こすかもしれない脅威に対して、製品が合理的な保護レベルを提供することを検証する。 http://industries.ul.com/software-and-security/product-security-services/product-testing-and-validation http://www.ul.com/cybersecurity
NIST CPS Public Working Group (CPS PWG)	サイバーフィジカルシステムパブリックワーキンググループ（CPS PWG）は専門家の集まりで、CPS の主要な要素について定義とアウトラインを提供し、米国経済の様々なセクタにおける CPS の開発と実装を促進することに貢献すると期待される。 https://www.nist.gov/news-events/news/2015/09/nist-releases-draft-framework-help-cyber-physical-systems-developers （翻訳者注：リンク切れのため、リンクをアップデート。さらに CPS フレームワーク v1.0 を策定済み）

Appendix D - その他のガイダンス文書

IoT 製品の設計者や開発者が検討を行うのに役立つ他の多くのガイダンス文書がある。

- PRPL Foundation は、2015 年に「**組み込み型コンピューティングの重要分野のためのセキュリティガイダンス**」を発表した。このガイダンスでは、セキュリティ分離アプローチ（security separation approach）の使用、および、組み込みシステムの安全な開発とテストの実施のような概念について詳しく説明している。

- アメリカ国防情報システム局（Defense Information Systems Agency, DISA）は、開発チームが自社製品のセキュリティを確保するために使用するセキュリティ技術導入ガイド（Security Technical Implementation Guides, STIGs）を長年にわたり提供してきた。とりわけ、IoT 製品開発者にとって有用と思われるものは、**アプリケーションセキュリティと開発 STIG** である。 この STIG は、開発、設計、テスト、保守、構成管理、およびトレーニングに関するガイダンスを提供する。

- マイクロソフトでは、**セキュリティ開発ライフサイクル（SDL）**の定義に長年を費やしてきた。 マイクロソフト SDL は、品質ゲート（quality gates）とバググラフ（bug bars）の作成、設計要件の確立、攻撃対象面（attack surface）のレビューの実施など、一連の実施対象項目を示している。

- OWASP IoT プロジェクトは、IoT に対する脅威を定義する上で大きな貢献をしている。 また、**IoT デバイスのテストガイド**を公開して、安全でないモバイルインターフェイス、安全でないソフトウェア/ファームウェア、物理的なセキュリティの不備などの事項に対するテスト方法について詳しく説明している。