エッジコンピューティングにおけるデバイスセキュリティ対策

Device Security Measures in Edge Computing

佐 藤 嘉 直

要 約 AI処理の普及と組み込みハードウエアの高性能化に伴い，Pythonなどのスクリプト言語で実現したAI機能をIoTデバイスに実装することが容易になった．一方，IoTデバイスはその利用形態上，屋外など第三者が手で触れられる場所に設置される場合があり，盗難などのリスクがある．もし分解され，内部の情報にアクセスされれば，プログラムコードや学習モデルなどが解読され，知的財産の流出に直接つながってしまう．

こうしたリスクに対応するため，IoTデバイス内部の情報の暗号化とネットワーク分割に加え，復号鍵を分割保存することで鍵の漏洩リスクにも対応した鍵分割方式の考え方を基に検証用システムを作成し，正常に動作することを確認した．実運用時にはポートの無効化や人為ミスの防止などへも注意を要する．

Abstract With the popularization of AI processing and the high-performance improvement of embedded hardware, it has become easier to implement AI functions obtained in scripting languages such as Python into IoT devices. On the other hand, IoT devices are often placed outdoors where they are accessible to outsider, which poses risks such as theft. If they are disassembled and accessed internally, program codes and learning models can be deciphered, directly leading to the leakage of intellectual property.

To address these risks, a verification system was created based on the idea of a key splitting method that addresses the risk of key leaks by storing decryption keys separately, in addition to encrypting the information inside the IoT device and splitting the network, and it was confirmed that it worked properly. During actual operation, care must be taken to disable ports and prevent human error.

1. はじめに

インターネットの普及に伴い，2010年代後半から活用され始めたモノのインターネットであるIoTは，遠隔監視や遠隔制御の分野への応用が急速に広がっている．応用形態の広がりに伴い，機能を実現するためのシステム規模は拡大の一途をたどり，システム構成要素の役割分担を明確にしてクラウド上の処理を軽減する様々な取り組みが行われている．その一つがエッジコンピューティングである．

エッジコンピューティングは，センサデータから解析に用いるデータを抽出する処理を，クラウドではなく末端のコンピュータ（以下デバイス）で実行するものである．そのためデバイスには科学技術演算や画像処理など，データをリアルタイムで処理する高性能な処理能力が求められる．また，最近では組み込まれる処理に人工知能（AI：Artificial Intelligence）を利用したものが多く見受けられ，これらの処理の開発言語としてPythonが広く使われるようになった．現在は半導体技術の進歩により，これらの要求を満足するプロセッサやメモリが安価に手に入るので，エッジコンピューティングを実現するIoTデバイスの導入が容易になった．

IoTデバイスはその利用形態上，屋外や公共の場，それに類似した場所に設置されるケース

がある．こうした場所では関係者以外でもデバイスに手を触れられる可能性があり，セキュリティリスクへの配慮は，通信データのみならず幅広い視野で対策することが求められる．それには機器の盗難や動作中にカバーを開けて内部へアクセスするなどの物理的な手段への対策に加え，ネットワーク経由での論理的な侵入を想定した対策をとることが重要である．

本稿では，デバイス内部に格納されている学習モデルやプログラムコードを効果的に保護する手法を紹介するとともに，デバイスに適用できるセキュリティ対策とその実装，および評価について述べる．2章でエッジコンピューティングの注意点，3章と4章でIoTデバイスのセキュリティ脅威と対策を説明する．5章では検証用システムの動作概要と評価結果を述べる．

2. エッジコンピューティングにおいて守るべきもの

IoT推進コンソーシアムと総務省，経済産業省が共同で策定・公表した「IoTセキュリティガイドライン ver1.0」[1]では，「方針，分析，設計，構築・接続，運用・保守」の各段階でのセキュリティに関する指針を定めている．「分析」で守るべきものを定義し，どのような手法で守るかを「設計」で吟味する．それらの対策を「構築・接続」で実装し，「運用・保守」で安全安心な状態を維持することが提唱されている．

エッジコンピューティングでは収集したデータの処理を行うアプリケーションコードや学習モデルがIoTデバイスに組み込まれる．これらは知財[*1]そのものであり，従来のIoTデバイスが守る対象としていたデータに加え，今後はアプリケーションコードや学習モデルも「守るべきもの」の重要な対象となる．

学習モデルを使った推論の実装には言語としての利便性からPythonが広く利用されている．Pythonは，従来の組み込み言語であるC言語のようなデバイス内部のリソースへの細かな配慮が不要であるとともに，プログラムコードはスクリプト形式をインタープリタで直接実行できるため，コードの修正や変更が効率良く行える利点がある．さらに，機械学習をはじめとしたAI処理に利用するライブラリが豊富で，少ない記述量で目的の処理を実現できることから，生産性の向上と開発期間の短縮が図れる．これらが，Pythonが優位的に選択されている理由である．また，ハードウエアに関してはARM（Advanced RISC Machine）プロセッサにLinux OSを搭載した比較的安価な組み込み機器が流通していることから，これらの組み込み機器を利用してPythonを動作させれば，開発費の低減と開発期間の短縮の観点で優位である．しかし，Pythonは「スクリプト言語」であることから，組み込み機器に侵入されプログラムファイルを盗まれれば，その動作原理を即時に奪われることになる．また，学習モデルについても同様の結果となる．

3. IoTデバイスにおけるセキュリティ脅威と対策

経済産業省はIoT製品に対するセキュリティ適合性評価制度を構築し，IoT製品のセキュリティ確保を広く普及させ社会への浸透を図っている．2024年8月に「IoT製品に対するセキュリティ適合性評価制度構築方針」[2]を公開し，以下の3段階にレベル分けした適合基準を定めた．

レベル☆1：IoT製品に共通して求められる最低限の適合基準を定め，それを満たすことをIoT製品ベンダーが自ら宣言するもの

レベル☆2：IoT製品類型ごとの特徴を考慮し，☆1に追加すべき基本的な適合基準を定め，それを満たすことをIoT製品ベンダーが自ら宣言するもの

レベル☆3以上：政府機関や重要インフラ事業者，大企業等の重要なシステムでの利用を想定したIoT製品類型ごとの汎用的な適合基準を定め，それを満たすことを独立した第三者が評価し認証するもの

レベル☆1については，「別添 ☆1セキュリティ要件・適合基準」[2]にて，以下に示す詳細な内容を定めている．

・汎用のデフォルトパスワードを使用しない
・脆弱性の報告を管理するための手段を導入する
・ソフトウエアを最新の状態に保つ
・機密セキュリティパラメータをセキュアに保存する
・セキュアに通信する
・露出した攻撃面を最小化する
・停止に対してレジリエントなシステムにする
・ユーザが簡単にデータを消去できるようにする
・製品に関する情報提供を行う

IoTデバイスはその利用形態から第三者の手の届く場所に設置される場合があり，4G，5Gに代表される移動通信やWi-Fi®[*2]，Ethernet[*3]などのインターネット通信の他に，シリアル通信やUSB，Bluetooth[*4]，JTAG（Joint Test Action Group）[*5]ポートなどのインタフェースを使った攻撃を想定した対策が求められる．さらに機器そのものが盗まれ，ハードディスクやeMMC（Embedded Multi Media Card），およびFlash ROMなどの補助記憶部（以下，ストレージ）に格納されているファイルを抜き出され，プログラムコードや学習モデルなどの機密情報を含んだ知財が奪われることを想定し，セキュアに保存する対策が求められる．

4. 知財流出に備えたセキュリティ対策の検討

本章では，デバイス内部のプログラムや学習モデル，機器設定などの付随情報を盗み出す手口と，万一盗まれた場合の対策について述べる．

4.1 ファイルの暗号化とオーバレイファイルシステム

盗難の手口として，機器そのものが盗み出されることやデバイス内部のストレージが取り出されることを想定し，知財であるプログラムや学習モデルを暗号化する．これにより仮に知財がストレージに保存されていたとしても，第三者にはその解読はできない．その実現手法として以下の二つの対策を考案した．

1）プログラムや学習モデルの暗号化

プログラムや学習モデルの暗号化と復号については二つの案が考えられる．

a）暗号化してストレージに保存し，起動時RAM上で復号する．
b）暗号化して管理サーバに保存し，デバイスの起動後，該管理サーバからプログラムをRAM上にダウンロードして復号する．

暗号解読には復号を行う鍵を使うので，どちらの暗号化案も暗号・復号の鍵が悪意のある第三者に渡らないことが絶対条件となる．b)の案は，プログラムをダウンロードする際の通信に時間やコストが発生する．そのため，a)の案の方が制約は少なく現実的である．

2）オーバレイファイルシステムの利用

プログラムや学習モデルは，ストレージに暗号化された形式のデータのみが格納され，復号後のデータは一時的でも格納しないことが望ましく，その方策の一つとしてオーバレイファイルシステムの利用が有効である．オーバレイファイルシステムでは，ストレージ内のファイルは読み取り専用とし，書き込みが発生した際の更新データはRAM上で生成され，以降の該当データのアクセスはRAMに対して行われる．その結果，ストレージ上のファイルは変更されず，ファイルの復号はRAM上に展開・配置され，実行されることとなる．RAM上のプログラムは無防備なスクリプト形式であるが，RAMの特性上，デバイスの電源をオフにすれば，これらのプログラムや学習モデルは消滅する．機器の盗難に際して，盗み出す際に電源をオフにせざるを得ないことから，このようなオーバレイファイルシステムを利用したファイルの更新痕跡の消去は，セキュリティ向上の有効な手段となる．

4.2　セキュリティ強度を高める復号方法

データやプログラムのセキュリティ強度を高めるには暗号化は必須の技術であり，公開鍵方式や共通鍵方式のいずれでも復号鍵の管理はセキュリティの要となる．そこで本節では復号鍵の管理に焦点を当て，暗号・復号の方法と鍵の漏洩防止策について述べる．

復号鍵のセキュリティ強度を高める工夫として，復号鍵を二つに分割し，一方はデバイスに格納し，もう一方は管理サーバに保管する．この仕組みでは，最初にデバイスとのセキュアな通信でデバイスを認証し，分割した復号鍵の断片を管理サーバからデバイスへ送信する．復号鍵を分割するのは，仮にデバイスが悪意の第三者に盗まれてファイルが奪われても，復号できないようにするためである．

デバイスは管理サーバから受け取った鍵の断片と自身で保管している鍵断片を合わせて，分割前の復号鍵を復元し，暗号化されたファイルを復号する．復号鍵の復元とファイルの復号はオーバレイファイルシステム上で行い，復号処理の終了時，管理サーバから受信した鍵断片と復元した復号鍵は削除する．これにより復号鍵はデバイスの起動時にオーバレイファイルシステム上でのみ復元され，電源オフで復号鍵の痕跡は無くなる．

暗号化したファイルを復号するためのプログラムは，サイズが小さいことから分割鍵とともに管理サーバからダウンロードする方式とする．復号プログラムをセキュアな管理サーバ上に保存することで，分割鍵を用いた復号アルゴリズムの流出を防ぐことができ，セキュリティ強化の一要素となる．

4.3　通信セキュリティ

鍵情報をやりとりする管理サーバとデバイス，および，これらをセッティングする際に使用するキッティング機間の通信セキュリティについては，1）セキュアな通信ができること，2）互いに相手の認証が行えること，3）デバイスを認識する固有の情報が容易に第三者に悟られないこと，の3点が主な要件となる．

各機器の設置場所は離れているので，通信には専用回線かインターネットを利用することになる．インターネット回線の場合，セキュリティを確保するためにネットワークは論理的に分離し，通常はVPNで接続する．ただし，昨今のランサムウエア被害ではVPNの脆弱性を突いた攻撃が増加しており，VPN機器のセキュリティ対策やE2Eの認証が求められる．

4.4　盗難について

デバイス内部のプログラムや学習モデル，機器設定などの付随情報を盗み出す方法として，1）ネットワークや通信インタフェースからファイルを転送する，2）機器を盗み出し，ストレージを取り出してファイルをコピーする，3）JTAG などのハードウエアデバッグポートからアクセスしてストレージやメモリ上のデータを読み取る，の 3 点が考えられる．

1）のネットワークを経由して侵入する手口に対しては，脆弱性によるセキュリティホールへの対応など一般的な IT セキュリティ対策を施すことに加え，組み込み機器に搭載したシリアル通信などの通信インタフェースからログインされることを阻止する対策も用意する．

また，2）の機器そのものが盗まれるケースにおいては，盗難の検知や防止，盗難された場合の対策を考える．盗難の防止については，デバイスの固定方法など物理的な対策が主となり，万一の盗難に備えて，デバイスが本来存在すべき場所から移動したことを検知する機能があればなおよい．その実現には GPS を利用した位置検知や，設備の電力線を利用した PLC 通信などによって設備機器と紐づけるなどの工夫を要する．しかし，これらの対策は製品のコスト増の要因になるとともに，運用上設置場所を移動する際の管理の煩雑さを招く．そのため，低価格のデバイスに適用するにはハードルが高い．

一方，盗難そのものを検知するのではなく，デバイス内部の知財を保護する観点に絞れば，内部に物理的にアクセスしても情報に手が届かない仕掛けを組み込むことで，目的は達成できる．機器あるいはストレージが盗難され，ファイルがコピーされるケースには，4.1 節で述べたようにストレージ内のファイルを暗号化して格納する手法が有効である．

3）の JTAG などハードウエアデバッグポートからの侵入は，動作中のデバイスに対する物理的アクセスの具体例である．このポートはデバイス筐体の内部にあり，デバイスのカバーを外せば接続できるインタフェースである．そこから CPU のレジスタを介してメモリ上のデータにアクセスできるので，悪意のあるデータの読み出しが可能である．しかし，このインタフェースが筐体内部にあることを利用し，カバーが開けられたことを検知したらストレージ上のデータを消去する仕組みを取り入れれば，知財保護の目的は果たせる．なお，カバー開の検知は，永久磁石と磁力に反応して作動するリードスイッチを用いれば，容易に実現できる．

5.　動作検証および評価

4.4 節で述べた JTAG から侵入して情報を盗み出す手法は技術レベルがかなり高く，一方でその対策であるカバー開検知やストレージ上の情報消去の手順は単純であり組み込む難易度は低いことから，そういった物理的盗難による知財流出のリスクは極めて低いと考える．本章では比較的リスクの高いデバイス内部の情報の論理的抜き取りに対応する仕組みの動作検証について述べる．具体的には，暗号化鍵を分割して鍵断片を管理サーバにて管理し，デバイス起動時にこのサーバから鍵断片と復号ロジックをダウンロードして，暗号化したプログラムコードや学習モデルなどの知財データを復号する仕組みについて，検証用システムを作成し，動作検証と評価を行った．

5.1　動作概要

検証用システムの機器構成は，図 1 に示すデバイス，管理サーバ，キッティング機の 3 機である．デバイスは，プログラムの復号とアプリケーションの起動を行う．Python で作られた

アプリケーションを扱うことから，Linux OS 搭載の機器とする．また，アプリケーション実行時に使用するオーバレイファイルシステムが動作できる十分な容量の RAM を搭載する．管理サーバは，暗号化と復号鍵の管理，デバイスとのアプリケーション通信を行う．デバイス同様 Linux 搭載機を利用し，デバイスとキッティング機以外の通信は遮断する．

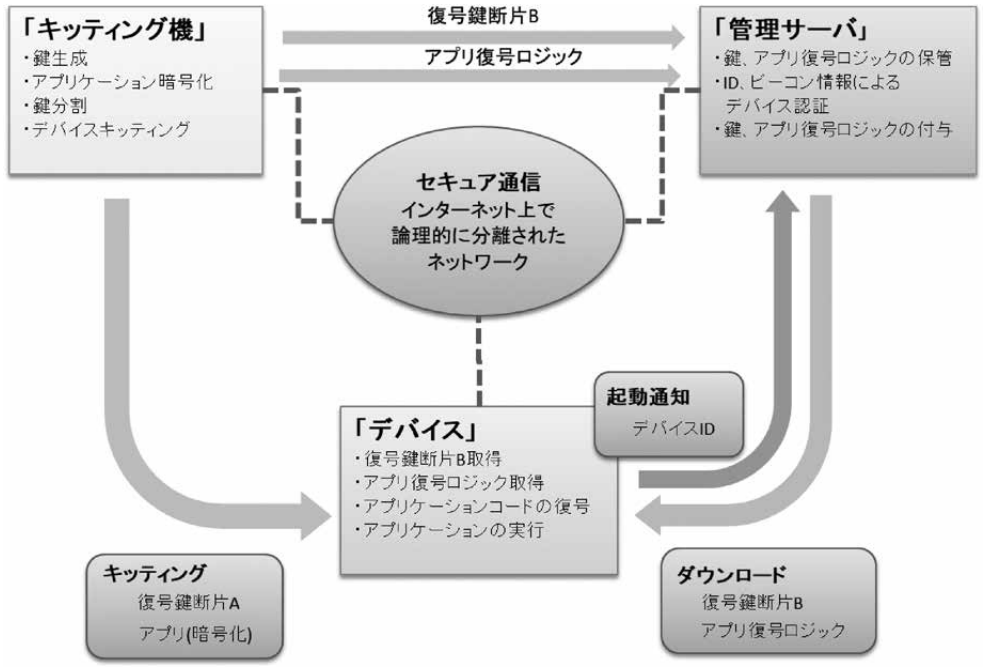

図 1　検証用システム構成

5. 2　各機器間の通信方式

IoT デバイスはその役割上，遠隔地に設置される場合が多いため，鍵のダウンロードに使う通信経路からの侵入を防ぐ対策を施すとともに，インターネット上の論理的な通信路は分離されていることが望ましい．この要件を満足する方法として，管理サーバとデバイス，キッティング機の間のセキュア通信は，CPU への負担が少なくセキュリティ的に堅牢な SYNCHRO 社の KATABAMI[3] を利用した．KATABAMI は，IPv6 ベースのメッシュネットワークで暗号化通信を行うオープンソースの Yggdrasil を利用し，公開鍵からハッシュ関数により生成された独自の IPv6 アドレスによって，IPv6 通信と互いのデバイス認証を行う．それによりインターネット上で閉域網通信を実現している．KATABAMI を用いて，暗号・復号に使う鍵の交換を行うことで，通信セキュリティを担保する．

5. 3　デバイスのキッティング

キッティング機で実行する，デバイスへのキッティング手順を以下と図 2 に示す．

手順①：復号鍵（共通鍵）を作成，鍵を分割する

手順②：開発したアプリケーションと起動シェルを共通鍵で暗号化する

手順③：デバイスに Linux と Python をインストールする

手順④：デバイスをインターネットに接続して KATABAMI をインストールする
手順⑤：KATABAMI のインストールで生成した IPv6 アドレスを管理サーバに登録する
手順⑥：鍵の断片を管理サーバに転送する
手順⑦：暗号化したファイルともう一方の鍵の断片をデバイスに転送する
手順⑧：キッティング機から共通鍵と分割した鍵を消去する

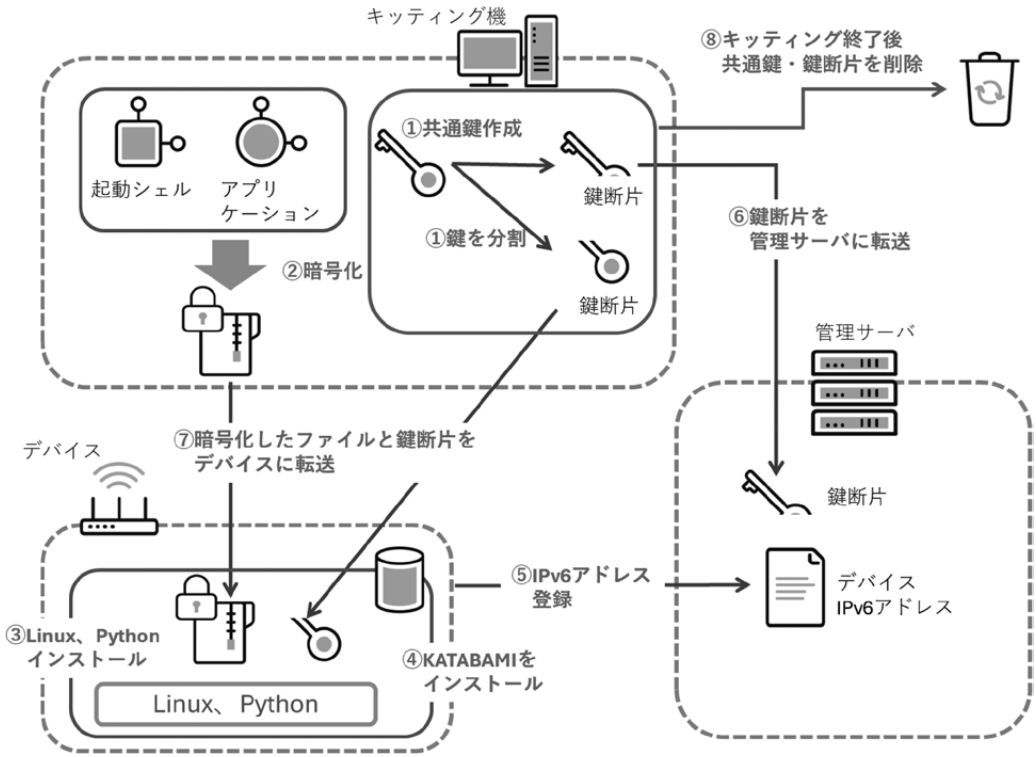

図 2　デバイスのキッティング

5. 4　デバイスの起動

デバイスの起動シーケンスを以下に述べ，図 3 に示す.

手順①：デバイスを電源オンする
Linux が起動し，オーバレイファイルシステムが利用可能となる
以降，ファイルの更新はオーバレイファイルシステム上で行われる
手順②：KATABAMI 通信によるデバイス認証により，デバイスは管理サーバと接続する
手順③：管理サーバから共通鍵の断片と復号プログラムをダウンロードする
手順④：復号プログラムを起動し，デバイスが持っている鍵の断片と管理サーバから取得した鍵の断片を合わせて共通鍵を復元する
手順⑤：復元した共通鍵で暗号化されたアプリケーションプログラムと起動シェルを復号する
手順⑥：復号後，共通鍵を廃棄する
手順⑦：AP 起動プログラムを開始して，アプリケーションを起動する

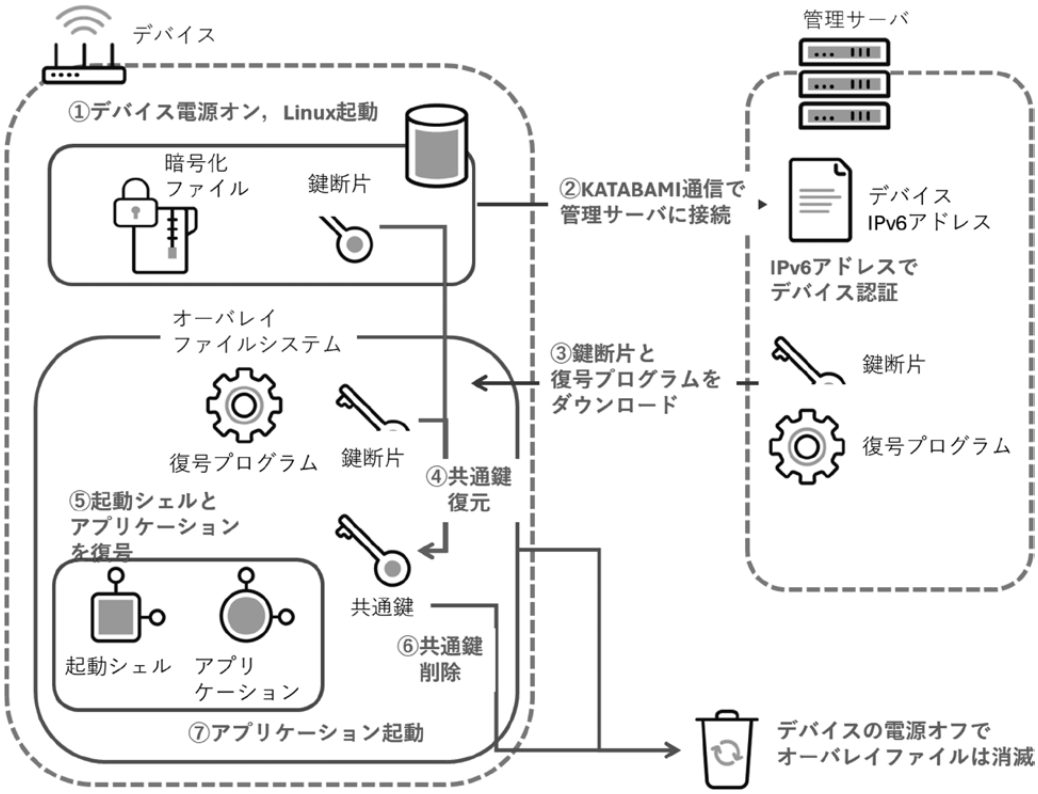

図3　デバイスの起動，コードの復号

5.5　評価

アプリケーションを構成するプログラムや学習モデル，起動プログラムは図3のストレージ上の「暗号化ファイル」として示す位置に，一つのファイルとして圧縮，暗号化し格納している．このファイルをデバイス起動後にオーバレイファイルシステム上で復号し，解凍したのちにアプリケーションが起動すること，正常に動作することを確認した．また，管理サーバから取得した共通鍵の断片と復元した共通鍵がオーバレイファイルシステム上で消去されていることを確認した．これらにより，プログラムコードや学習モデルを暗号化し安全にデバイスへ組み込めることが分かった．

それぞれの機器間の通信のセキュリティも，KATABAMI を使うことで担保されている．さらに，実運用時にはそれぞれの機器で以下の対策を施す．

- デバイス
  - ・リモート保守用機器との通信も KATABAMI ネットワーク上で行う．その他アプリケーションサーバに接続する際に使うポートはデバイスからの接続のみとし INPUT は受け付けない．それ以外のポートは閉じる
  - ・USB IF を搭載している場合は USB デバイスでの自動起動は行わない
  - ・シリアル通信 インタフェースは無効とする

- 管理サーバ
  - ・KATABAMI 以外の通信は受け付けない
  - ・接続するデバイスの IPv6 アドレスをホワイトリストに登録し，それ以外の機器とは接続しない
- キッティング機
  - ・コードの開発でインターネットに接続するため，ファイアウォールやマルウエア対策に加えて，メールソフトの削除など，人の操作によるセキュリティ事故への対策に十分注意する

6. お わ り に

本稿で紹介した技術により，デバイス内部に組み込まれた高度な状態監視や画像解析のアルゴリズムを守ることができる．今後，この仕組みにアプリケーションの更新機能を追加するとともに，盗難検知や盗難時のストレージ内情報の消去など，デバイスの具体的な実装開発を進める．この技術が，あらゆるものがインターネットにつながる時代のセキュリティ手法の一つとして活用され，IoT 技術の発展に寄与できれば幸いである．

＊ 1　知財（知的財産）とは，一般的には音楽，映画，絵画などの著作物や，発明・実用新案，デザイン，商標などを指す．本稿ではプログラムコードや営業秘密などの無体物を指すものとして用いる．
＊ 2　IEEE 802.11 準拠のデバイス間接続の登録商標．
＊ 3　IEEE 802.3 準拠の通信規格．
＊ 4　IEEE 802.15.1 準拠の近距離無線通信規格．
＊ 5　IEEE 1149.1 準拠の，半導体チップ（IC チップ）の検査用端子の仕様や検査手法の標準規格．

参考文献　［1］ IoT セキュリティガイドライン ver1.0，IoT 推進コンソーシアム，総務省，経済産業省，2016 年 7 月．https://www.soumu.go.jp/main_content/000428393.pdf
［2］ IoT 製品に対するセキュリティ適合性評価制度構築方針，経済産業省 商務情報政策局サイバーセキュリティ課，2024 年 8 月．
https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_cybersecurity/iot_security/pdf/20240823_1.pdf（本編）
https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_cybersecurity/iot_security/pdf/20240823_2.pdf（別添 ☆1 セキュリティ要件・適合基準）
［3］ KATABAMI Series 概要説明 技術説明，株式会社 SYNCHRO
https://www.udc-synchro.co.jp/service/katabami/

※　上記参考文献に含まれる URL のリンク先は，2024 年 12 月 5 日時点での存在を確認．

執筆者紹介　佐　藤　嘉　直（Yoshinao Sato）
1985 年日本ユニシス(株)入社．2200 シリーズ向け通信制御装置や磁気ディスク制御装置の開発に従事し，2004 年，ユニアデックス(株)転籍，日本語プリンタ，特殊端末の開発を経て IoT を始めとする組込みシステムに関連する機器の開発に従事．