7J-03

IoT機器向けデータ処理基盤におけるセキュアタスク生成機能の検討

伊藤 大智† 金城 聖† 毛利 公一†
†立命館大学情報理工学部

1 はじめに

IoT 機器は非力な CPU，少ないメモリなどハードウェアの制限がある．加えて，MMU を搭載していない IoT 機器も存在し，各タスクのメモリ空間の分離やメモリ保護がおこなわれていない場合がある．また，汎用的な PC で使用されている OS と比べて IoT 機器で使用されている OS のセキュリティ機能はスタックオーバフロー検知などの最低限のものにとどまっている．これらのことから OS やタスクの脆弱性を利用し，IoT 機器が保有するセンシティブなデータの改ざん，漏洩の恐れがある．

この問題に対して我々は，IoT 機器向けにセキュアな隔離空間でセンシングや署名など改ざんされてはいけないタスクの実行を可能とし，センシティブな情報を安全に処理するデータ処理基盤の研究をおこなう．

このデータ処理基盤では，センシングデータに署名をすることでデータの改ざんを防止したいような IoT 機器を想定する．センシングや署名などの処理をセキュアタスク (Secure Task/ST) として隔離空間にロードし，隔離空間が使用するメモリやレジスタを通常空間の OS やタスクから秘匿した状態で ST を実行させる．ST を隔離空間上で実行させることで ST が扱うデータの改ざんなどの不正な操作を防ぐことができる．隔離空間を実現する方法として Trusted Execution Environment(TEE) 実装の一つである Arm TrustZone[1] を利用する．

本論文では，このデータ処理基盤における Secure Task 生成機能について検討した．

2 TrustZone

2.1 Cortex-M 向け TrustZone

Armv8-M アーキテクチャを採用している Arm Cortex-M シリーズでは，セキュリティ拡張機能として TrustZone を搭載している．Cortex-M 向け TrustZone の概要図を図 1 に示す．TrustZone は TEE の実装の 1 つで，通常の実行空間とは別にセンシティブな情報を扱うための隔離実行空間を提供する．TrustZone では通常空間のことを Non-Secure World(NSW)，隔離実行空間のことを Secure World(SW) と呼ぶ．Trust-

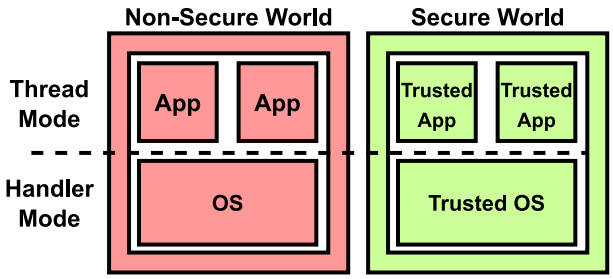

図 1 Cortex-M 向け TrustZone の構成図

Zone は起動時に SW と NSW の 2 つの実行空間を作成する．NSW は SW の情報に直接アクセスできない．それぞれの空間は従来の実行空間と同じように Thread Mode と Handler Mode が存在し，メモリや例外は各ワールドのどちらかに属する．

3 データ処理基盤

3.1 想定しているシステム

想定システムの概要図を図 2 に示す．SW でセンシングや署名のような改ざんされてはならない処理をおこない，NSW で外部サーバにデータを送信するような IoT 機器を想定している．

3.2 データ処理基盤の概要

設計コンセプトは NSW を主体とし必要なときに SW でタスクの隔離実行が可能なシステムである．設計の概要を図 3 に示す．NSW のタスクである Non-Secure Task(NST)，SW のタスクである ST，NSW から呼び出される ST の Secure Function(SF) がある．そして，ST，SF を SW で動作させるために必要なソフトウェアの集まりの Runtime が存在する．NW と SW の切り替えのために NSW に SW として動作する NST が存在し，NW の Scheduler によってディスパッチされることによって SW に遷移する．つまり，NSW のタスクの 1 つとして SW に処理時間が与えられ各ワールドが動作している．また，NSW 側の OS および Runtime は FreeRTOS[2] をベースとしている．

4 Secure Task 生成機能

ST，SF のうち本論文では ST の生成機能について述べる．

4.1 Secure Task 生成機能の特徴

ST の生成機能は以下の特徴がある．

Study of secure task generation function in data processing platform for IoT devices
Daichi Ito†, Akira Kanashiro†, and Koichi Mouri†
†College of Information Science and Engineering, Ritsumeikan University

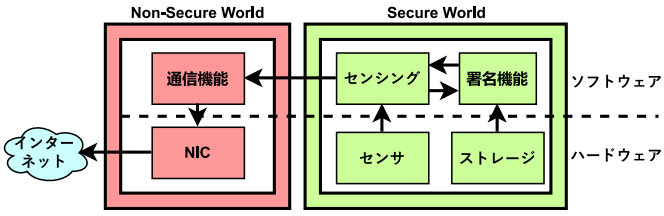

図 2　想定システム

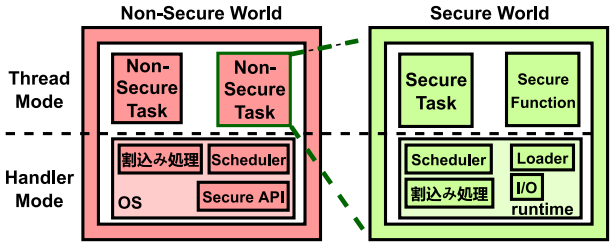

図 3　データ処理基盤の概要

- NSW が必要なときに SW を利用し，タスクを隔離実行させることができる
- ST を動的に生成できるため，システムの更新やバグ解決が容易にできる
- NSW に悪意のあるプログラムが侵入しても SW の資源は守られる

4.2　Secure Task 生成機能の処理手順

ST 生成のために検討した手順を以下に示す．

(1) NSW が SW に対して ST を生成要求する

(2) Secure 属性のメモリ領域に ST のバイナリを配置する

(3) SW の Scheduler に (2) で配置したタスクを登録する

(1)(2) を実現するために TrustZone のメモリ分離を利用する．TrustZone ではワールドによってアクセスできるメモリ領域を設定することで，各ワールドがアクセスできるメモリ領域を分離している．このメモリ領域の設定は Implementation Defined Attribution Unit(IDAU) および Secure Attribution Unit(SAU) で管理され，Secure 属性，Non-Secure Callable(NSC) 属性，Non-Secure 属性 3 つのメモリ属性を付与できる．

(1) の手順では両ワールドからアクセス可能な NSC 属性のメモリ領域を経由し，NSW から SF を呼び出すことで実現する．

(2) の手順では 2 つの実現方法が考えられる．1 つ目に，Non-Secure 属性のメモリ領域にある ST のバイナリを Secure 属性のメモリ領域にコピーする実現方法がある．2 つ目に，メモリ属性を管理している IDAU と SAU のうち，動的にメモリ属性を変更できる SAU を用いて，ST のバイナリが配置されている Non-Secure 属性のメモリを Secure 属性に変更する実現方法がある．

(3) の手順では FreeRTOS のタスク生成関数である xTaskCreate 関数を用いて SW の Scheduler にタスクを登録する．

```
[NW]Secure Function Call
[SW]Hello Secure World
[NW]Secure Function Return
```

図 4　出力コンソール

4.3　Secure Function の作成と実行

4.2 節で述べた手順 (1)(2)(3) のうち，(1) で作成する SF の作成と実行の確認をおこなった．SF を作成するためには，ST に cmse_nonsecure_entry 属性を付与することによって作成できる．この属性を付与することで NW から SF を呼び出すための命令がコンパイル時にコンパイラによって追加される．

今回は引数なし，戻り値なしの “Hello Secure World” と出力する SF を作成した．この SF を NSW から呼び出し実行させたところ動作を確認した．出力のコンソールを図 4 に示す．左端に [NW] と出力されているのが NSW から出力した文字で [SW] と出力されているのが SW から出力した文字である．1 行目の “Secure Function Call” の直後に NSW から SF が呼び出され，SW で “Hello Secure World” と出力したあと，NSW に帰還し “Secure Function Return” を出力している．

5　おわりに

本論文では Secure Task 生成機能の検討をおこなった．この機能を利用し，動的に ST を生成し SW で実行させることで不正な操作を防ぐことができ，IoT 機器のセンシティブなデータが保護できる．今後はこの検討に加えて他の TEE における Task 生成の実装の調査をもとに NW から SW へ渡す引数や動作 mode，特権レベルなどの細かな設計をおこない，実装を進める．

参考文献

[1] Arm Limited: Arm TrustZone Technology for the Armv8-M Architecture, (入手先)https://developer.arm.com/documentation/100690/0201/?lang=en(参照 2022-11-22).

[2] Amazon Web Services: The FreeRTOS Reference Manual API Functions and Configuration Options, (入手先)https://www.freertos.org/fr-content-src/uploads/2018/07/FreeRTOS_Reference_Manual_V10.0.0.pdf(参照 2022-11-22).