組込機器でリアルタイム実行可能なセキュリティ攻撃検知・分類アルゴリズムの実装

小河原 徹，山本 泰生，曾 珍，廣部 直樹

IoTの普及に伴い、インターネットには組込機器も接続されるようになり、セキュリティ攻撃の危機にさらされている。

セキュリティ攻撃によって組込機器の制御が停止させられると、大きな損失が予想される。従って、攻撃を受けても制御を停止させないような対策が求められている。

セキュリティ攻撃を受けた際、即座にその攻撃の種類を特定できれば、攻撃に合った対応を自動的に実施できる。結果、制御を停止させず損失を最小限に留めることが可能になる。そのため我々は、軽量な攻撃検知・分類アルゴリズムを開発・実装し、組込機器上でリアルタイムに実行できる性能であることを確認した。

Implementation of Real-time Security Attack Detection and Classification Algorithms for Embedded Devices

KOGAWARA Toru, YAMAMOTO Taisei, ZENG Zhen and HIROBE Naoki

With the spread of IoT, embedded devices are also connected to the Internet, which is in danger of security attacks.

If a security attack stopped control of embedded devices, a large damage is expected. Therefore, a measure is demanded which does not stop the control even if the device is attacked.

If the device can immediately classify the type of attack when it receives a security attack, it can automatically respond to the attack. As a result, it is possible not to stop the control, and to minimize the damage. That is why we developed and implemented a lightweight attack detection and classification algorithm, and confirmed that it was possible to be run in real-time on embedded devices.

1. まえがき

多様な分野でのIoTの普及に伴い、インターネットにはパソコンやサーバだけでなく、様々な組込機器が接続されるようになってきた。その結果、従来のパソコンやサーバと同様、組込機器もまたセキュリティ攻撃の危機にさらされている。

中でも、自動車の車両制御装置や工場の機械の制御装置などは、物理的に動作する機械を制御しているため、セキュリティ攻撃によって人的・物的に大きな損害を受ける可能性がある。

例えば、自動車のハッキングに成功し、携帯電話回線経由で自動車のハンドルやエンジン・ブレーキまでを遠隔操作できたという報告がある[1)]。

また、工場や発電所などの制御機器を対象としたウィルス等の不正ソフトウェアも多数報告されており、機械の破損や停電などの大きな損失を受けている[2)]。

セキュリティ攻撃への対策を考えるにあたって、NIST（米国立標準技術研究所）発行の「サイバーセキュリティフレームワーク」[3)]を用いるのが一般的である。これはセキュリティ対策を大きく5つのフェーズ「特定」「防御」「検知」「対応」「復旧」に分けて取り組むためのものである。図1に、サイバーセキュリティフレームワークの概要を示す。

Contact : *KOGAWARA Toru* toru.kogawara@omron.com

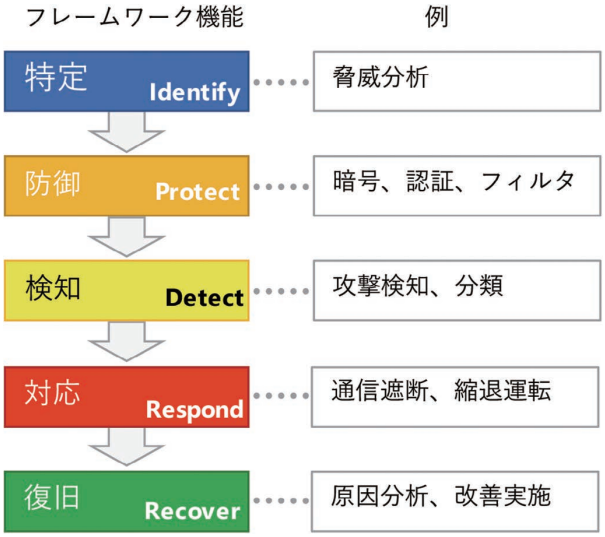

図 1　NIST サイバーセキュリティフレームワークの概要

「特定」により対象システムのセキュリティリスクを分析し、「防御」によってセキュリティ攻撃から対象システムを保護するのが、セキュリティ対策の基本である。しかし、セキュリティ攻撃は日々進化しており、「防御」をすり抜ける攻撃がいずれ出現する。

そこで、「防御」を突破された場合でも、そのセキュリティ攻撃を「検知」すること、また検知した攻撃に対して適切な「対応」を行うことで、被害を最小化することが必要となる。

本稿では、低リソースな組込機器においても「対応」によって被害を最小化することを目的とした、組込機器上でリアルタイム実行可能な「検知」技術の開発および実装について述べる。

2. 課題

「検知」した攻撃に対して適切な「対応」を行うには、どのような攻撃であったか特定する必要がある。受けた攻撃の種類に応じて、適切な対応策を選択するためである。攻撃の検知と合わせて分類を行うため、本稿ではこれらを合わせて「検知・分類」と呼ぶ。

また、物理的に動作する機械を制御している機器においては、何らかの問題が発生した際には、周囲の人員や機械そのものに損害を与えないよう速やかに対処しなければならない。セキュリティ攻撃への対処も同様であるため、「検知・分類」はリアルタイムに実行できる必要がある。

従って本稿では、組込機器上でリアルタイムに攻撃検知・分類を行うことを課題とする。

3. 従来の攻撃検知技術

組込機器レベルにおけるセキュリティ対策はまだ始まったばかりであり、特に攻撃検知・分類機能をリアルタイムに実行できる製品は存在しない。

一方で、パソコンやサーバ等、IT 系のセキュリティ対策ソフトウェアでは、攻撃検知・分類機能をリアルタイムに実行できるものも存在する。

IT 系の攻撃検知・分類技術では、例えば検知したウィルスの亜種やバージョン名など、非常に詳細な内容まで特定する。また、未知の攻撃手口を機械学習＋AI で検出するものもある。そのため、大量のデータと、それを検索するための大量の計算量を必要とする。

結果、IT 系の攻撃検知・分類技術は非常に高速な CPU と、大きなメモリ・ストレージ容量とを必要とするため、組込機器にそのまま移植することはできない。

ここでは一例として、Blue Planet works 社の AppGuard[4] を挙げる。

AppGuard は Windows 系 OS 上で動作するソフトウェアであり、非常に軽量であることをセールスポイントとしている。それでも、プログラム本体（防御エンジン）のサイズが 1 MB、ディスク容量は 500 MB、CPU の動作周波数は 1.8 GHz 以上 を要求する。Windows 用アプリケーションとしては非常に軽量な部類ではあるが、それでも組込機器のスペックでは満たすことができない。

例えば今回の開発で使用した車載マイコンのスペックは、Flash ROM（パソコンのディスクに相当）の容量 1 MB、メモリ容量 128 KB、CPU 動作周波数 約 100 MHz であり、同技術をそのまま流用できるような環境ではない。

4. 攻撃検知・分類アルゴリズムの開発

ここでは、我々が開発した既に発表済みの攻撃検知・分類技術[5] について説明する。これは、制御を行う組込機器を対象としたものであり、組込機器でリアルタイムに実行することを目的としている。

組込機器がセキュリティ攻撃を受けた場合、攻撃は機器の外部から行われるため、外部からの入力に正常でない入力が含まれている。正常でない入力を「異常」と定義し、異常を複数検出できれば、それは攻撃を受けていると判断できる。これを攻撃検知の基本的なアイディアとする。

攻撃を構成する異常の組み合わせは、攻撃の種類に依存する。従って、ある特定の異常の組み合わせを検出することは、ある特定の攻撃を検知することを意味する。

このような特定の攻撃に対する検知を、検知したい攻撃の種類それぞれについて実行することで、結果として攻撃検知と分類を同時に実施することができる。

攻撃検知・分類アルゴリズムの概念図を図 2 に示す。

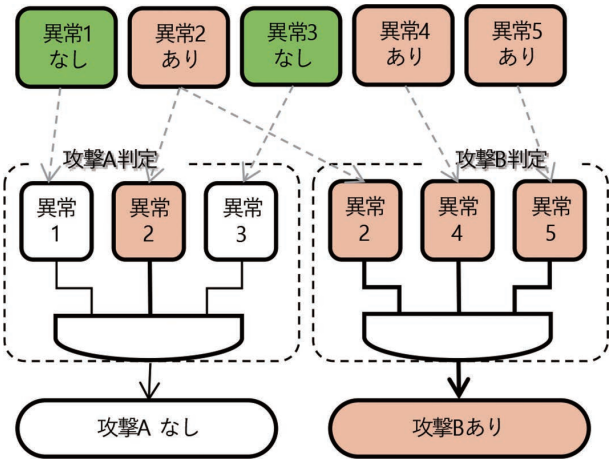

図 2　攻撃検知・分類アルゴリズムの概念図

単純な異常のあり・なしの組み合わせを判定するだけであり、組込機器に要求される「高速に処理できる」「必要なデータ容量が小さい」の両方を満たせると考えられる。

また、分類する攻撃の種類は、表 1 に示す 6 種類とした。一般的な脅威分析モデルの 1 つである STRIDE[6] に沿ったものであり、S, T, R, I, D, E を頭文字とする 6 種類の脅威に対応している。

表 1　分類対象となる攻撃の種類

攻撃の種類	具体例
なりすまし Spoofing	正規の機器になりすまし、偽の制御コマンドを送り込む
改ざん Tampering	ファームウェアや設定データを改ざんする
否認 Repudiation	侵入の証拠を消すため、ログを消去・上書きする
情報漏えい Information disclosure	特殊な通信コマンドや設定データ等、非公開の情報を取得する
サービス拒否 (DoS 攻撃) Denial of service	機器に大量の通信メッセージを送信し、サービスを停止させる
特権昇格 Elevation of privilege	不正に特権を取得する

さらに、複数の攻撃を組み合わせた複合的な攻撃を受けた場合、どの攻撃に対して優先的に対応すべきかを判断する必要がある。そのため、攻撃の進行を不可逆な状態遷移としてモデル化し、最も進行した攻撃について対応を行う、という手法を提案した[7]。

攻撃の進行をモデル化した状態遷移図（攻撃遷移図と呼ぶ）を図 3 に示す。

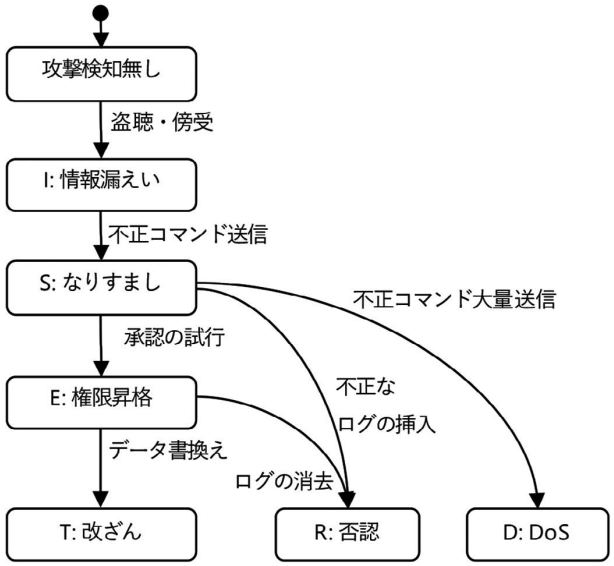

図 3　攻撃遷移図の例

5. 攻撃分類の細分化

5.1 攻撃分類の粒度における課題

4 章で紹介した技術では、攻撃を 6 種類に分類した。しかし、受けた攻撃の種類に応じて適切な対応を行うという目的に対しては、この 6 種類の分類ではまだ十分ではない。

例えばサービス拒否攻撃においても、単に通信バスが占有されてサービスが成立しない場合と、大量のメッセージの処理によって CPU の処理時間が間に合わなくなって機器が機能停止する場合とでは、とるべき対応は異なると考えられる。

従って、より攻撃を細分化して検知・分類する必要がある。

5.2 内部状態異常の統合による細分化

4 章で紹介した攻撃検知・分類アルゴリズムでは、異常の検出対象を外部入力としていた。

一方で、5.1 節で例を挙げた通り、適切な対応を行うためには、外部入力だけでなく組込機器の内部状態も併せて判断する必要がある。

機器の内部状態についても、正常でない状態や挙動を「異常」と定義する。さらに、内部状態による異常が発生したときに検知される攻撃を、新たな種類の攻撃であるとして細分化する。これを図 3 の攻撃遷移図に追加することで、外部入力異常と内部状態異常の両方を統合して攻撃検知を行うことができるようにした。

細分化した攻撃遷移図の例を図 4 に示す。

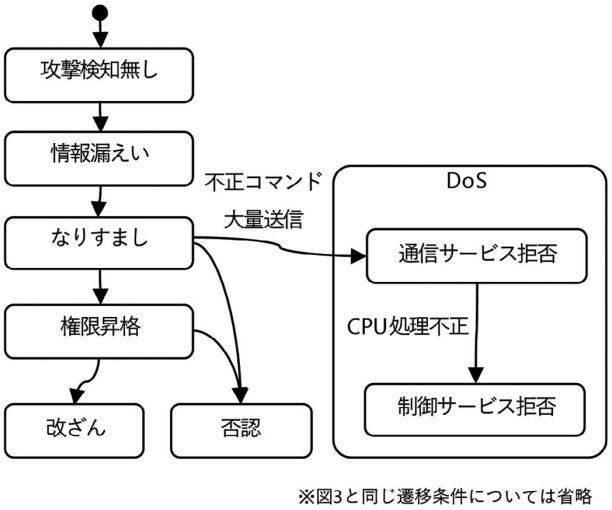

図 4　内部状態異常を統合して細分化した攻撃遷移図の例

6. アルゴリズムの実装

6.1 車載 ECU 想定モデル

実装対象として、自動車に搭載される電子制御装置（以下、車載 ECU）を想定したモデルを構築した。車載 ECU は組込機器の中でも低リソースであり、また 1 章で述べたようにセキュリティ攻撃のリスクにさらされていることから、本技術の適用先として適切である。

表 2 にハードウェア仕様を示す。このハードウェアは車載用ワンチップマイコンを CPU とし、標準的な車載ネットワークである CAN を複数チャネル持つ。また、ECU を想定した機能として、指定されたメッセージを受信・送信する。

表 2　車載 ECU 想定モデル ハードウェア仕様

項目	仕様
CPU 動作周波数	96 MHz
Flash ROM	1 MB
RAM	128 KB
通信プロトコル	CAN
通信速度	500 kbps
制御周期	1 ms

6.2 アルゴリズムの実装方法

今回開発した攻撃分類アルゴリズムは、攻撃の種類ごとに独立して判定するため、ビット演算によって高速処理を行うことが可能である。

さらに攻撃の分類について、4 章で説明した通り、不可逆な状態遷移であるとモデル化した。そのため、特定の攻撃 A は「A まで遷移してくる遷移条件をすべて満たす」（IN 条件）と「A から次へ遷移する条件を一つも満たさない」（OUT 条件）の両方を満たした場合に検知したと判定できる。図 5 に概念図を示す。

IN 条件は全て AND で成立するため、ビット列で表現できる。ビット列において、IN 条件を構成する異常に対応するビットの値を 1、それ以外のビットの値を 0 としたものを、IN 条件のビットマスクと呼ぶ。OUT 条件も同様にビットマスクで表現できる。

検出した異常についてもビット列を用意し、異常を検出したビットの値を 1、そうでないビットの値を 0 としておけば、IN 条件と OUT 条件の判定は、それぞれのビットマスクとの AND 演算 1 回で実現できる。つまり、1 つの攻撃検知を判定するにはビット演算 2 回で済む。

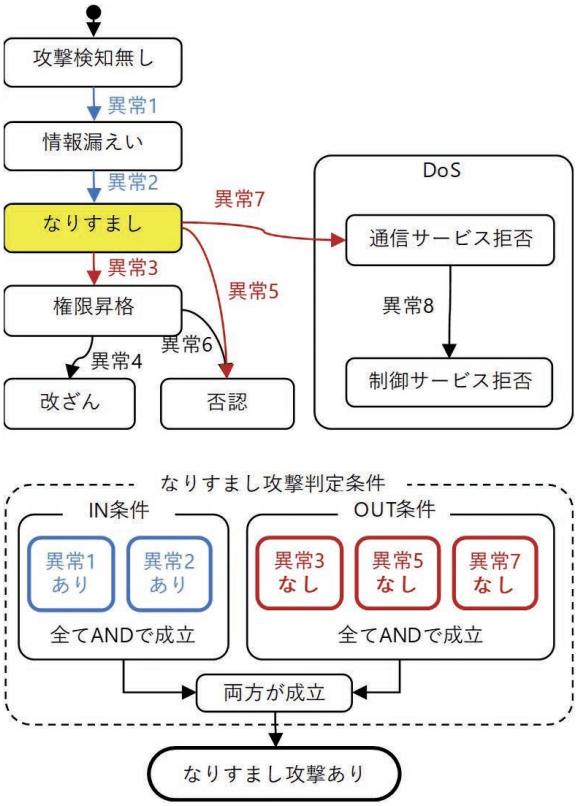

図 5　攻撃遷移図を利用した判定条件

このように実装した場合の、CPU の演算量とデータ容量を試算する。

検知する攻撃の種類を n 種類、攻撃検知のために検出する異常の種類を m 種類とした場合の、各攻撃の IN 条件と OUT 条件のために必要なビットマスクを図 6 に示す。

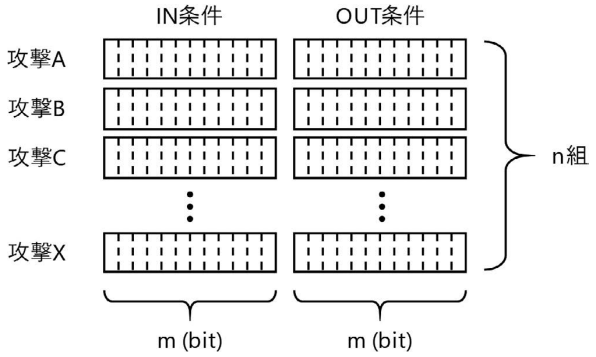

図 6　攻撃検知に必要なビットマスク

CPU 演算量について試算する。車載 ECU で一般的な 32 bit CPU を想定すると、一度に 32 bit の演算が可能なので、演算回数は $2\times\lceil m/32\rceil\times n$ 回となる。

同様に、必要データ容量について試算する。データはバイト単位で保持されるため（1 バイト＝8 ビット）、必要なデータ容量は $2\times\lceil m/8\rceil\times n$ バイトとなる。

6.3 攻撃検知・分類機能仕様

車載 ECU 想定モデルに対して、セキュリティ対策を設計し、分類する攻撃と、攻撃検知のための異常検出について抽出した。その結果、攻撃分類をより細分化し、11 種類とした。

設計した攻撃検知・分類機能の仕様を表 3 と図 7 に示す。

表 3　攻撃検知・分類機能仕様

項目	仕様
検知・分類可能な攻撃の種類	11
検知・分類に使用する異常の種類	45

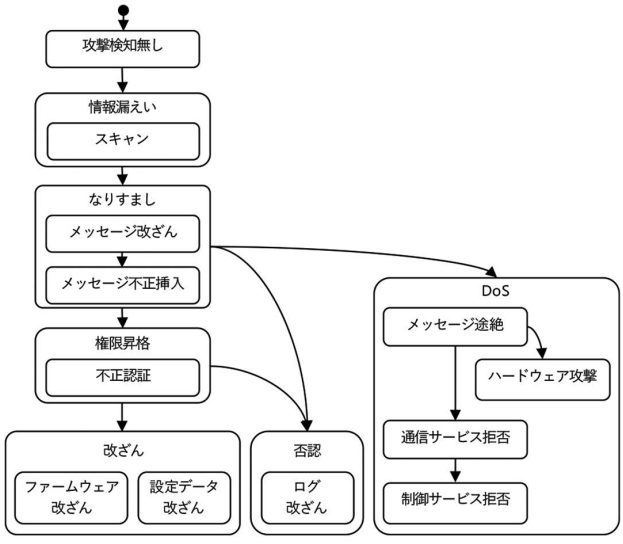

図 7　設計した攻撃遷移図

この機能仕様と 6.1 節のハードウェア仕様を、6.2 節の試算結果に当てはめて、実際の処理時間とデータ容量を見積もる。6.2 節のパラメータ n, m については、それぞれ攻撃の種類 n=11, 異常の種類 m=45 となる。

CPU 演算量の試算結果はビット演算 44 回となる。ビット演算が CPU クロック 1 周期で完了すると仮定すると、演算にかかる処理時間はおよそ $44/(96\times10^6)\fallingdotseq0.5\,\mu$s であり、制御周期 1 ms の 1/2000 程度となる。

また、データ容量の試算結果は 132 バイトとなるので、Flash ROM 容量 1 MB に比して 1/8000 程度となる。

以上より本アルゴリズムは、車載 ECU のような組込機器では、全く問題なく実装できる演算量とデータ容量であると見積もることができる。

7. 実装したモデルの評価

7.1 要件

攻撃検知・分類機能の要件として、以下の 2 点であると定義する。

・正しく機能していること
　定義した全ての攻撃の種類について、検知して正しく分類できる。
・組込機器本来の機能を阻害せず、リアルタイムに実行できる性能であること
　メモリ使用量が十分に小さい。（Flash ROM, RAM それぞれの容量の 10%未満）
　また、最も負荷が重い状態でも、全機能の処理時間が制御周期（1 ms）を超えない。

7.2 機能評価

定義した全ての攻撃の種類それぞれについて、攻撃方法を設定した。表 4 に示す。

表 4　攻撃検知・分類機能テストケース

No.	攻撃の種類	攻撃内容
1	スキャン	攻撃機器から通信経路に、CAN ID 総当たりのメッセージを注入
2	メッセージ改ざん	中間機器を通信経路に挟み、正規メッセージのデータを書き換えて転送
3	メッセージ不正挿入	攻撃機器から通信経路に、正規メッセージと同じ ID の不正メッセージを注入
4	不正認証	不正な暗号鍵を用いて、攻撃機器から認証シーケンスを開始

5	メッセージ途絶	中間機器を通信経路に挟み、特定 ID のメッセージのみを転送せず遮断
6	ハードウェア攻撃	特定 ID のメッセージが送信されている間だけ、攻撃機器から不正信号を送り、H/W エラーを起こす
7	通信サービス拒否	攻撃機器から通信経路へ受信対象でないメッセージを大量注入し、通信を飽和させる
8	制御サービス拒否	攻撃機器から通信経路へ受信対象のメッセージを大量注入し、制御処理を飽和させる
9	セキュリティログ改ざん	不揮発メモリ上のセキュリティログを改ざんしてモデル起動
10	ファームウェア改ざん	不揮発メモリ上のファームウェアを改ざんしてモデル起動
11	設定データ改ざん	不揮発メモリ上の設定データを改ざんしてモデル起動

テスト実施時のシステム構成の模式図を図 8 に示す。

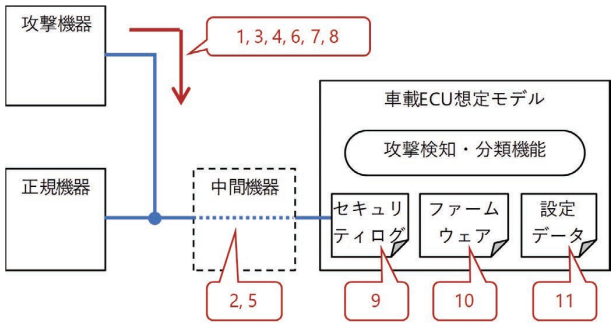

図 8　システム構成の模式図

表 4 のテストケース番号に対応する攻撃箇所を、図中の吹き出し内の番号で示している。

図 8 のシステム構成にて、表 4 の各テストケースを実施した結果を、表 5 に示す。

表 5　攻撃検知・分類機能テスト結果

No.	経過時間 (ms)	攻撃検知・分類結果	判定
1	1	スキャン	OK
2	1	メッセージ改ざん	OK
3	1	メッセージ不正挿入	OK
4	1	不正認証	OK
5	200	メッセージ途絶	OK
6	200	メッセージ途絶	OK
	226	ハードウェア攻撃	
7	1	スキャン	OK
	38	メッセージ途絶	
	1419	通信サービス拒否	
8	1	メッセージ不正挿入	OK
	27	メッセージ途絶	
	376	制御サービス拒否	
9	—	セキュリティログ改ざん	OK
10	—	ファームウェア改ざん	OK
11	—	設定データ改ざん	OK

No. 1〜11 の各テストケースにおいて、検知・分類した攻撃と、攻撃開始から検知・分類までの経過時間を記載している。攻撃の種類によっては、まず他の攻撃として検知される場合もあるが、最終的には期待する攻撃として検知・分類されているため、問題ない。

なお No. 9〜11 のケースについては、ECU 想定モデル起動時の初期化処理で検知されるため、経過時間は無効としている。

この結果から、実装した攻撃検知・分類機能が正しく機能していることを確認できた。

7.3 性能評価

まずメモリ使用量について確認結果を示す。

攻撃検知・分類機能が使用する Flash ROM 容量は、アルゴリズムのコード分と、6.2 節で示したビットマスクの分である。使用する RAM 容量は、実行時の状態を保存するための領域であり、静的に確保している。

メモリ使用量の測定結果を表 6 に示す。

表 6　攻撃検知・分類機能 メモリ使用量測定結果

項目	メモリ使用量	
	要件	性能
Flash ROM	100 KB 未満	6.2 KB
RAM	12.8 KB 未満	8.4 KB

Flash ROM 使用量、RAM 使用量いずれも、表 2 で示した Flash ROM、RAM それぞれの総容量の 10%未満であり、要件を満たしている。

次に、処理時間の確認結果を示す。

処理時間の測定は、内部処理を切り替えるごとにデジタル信号を出力するようソフトウェアを改造し、デジタル信号の波形を観測することで実施した。なお、この改造による処理時間への影響は無視できる程度であった。

処理時間の測定結果を図 9 に示す。

処理時間が最大となった場合＝通信負荷が 100%であった場合でも、要件である 1 ms 以内を余裕をもって満たしている。

この結果から、実装した攻撃検知・分類機能が、組込機器本来の機能を阻害せずにリアルタイムに実行できる性能であることが確認できた。

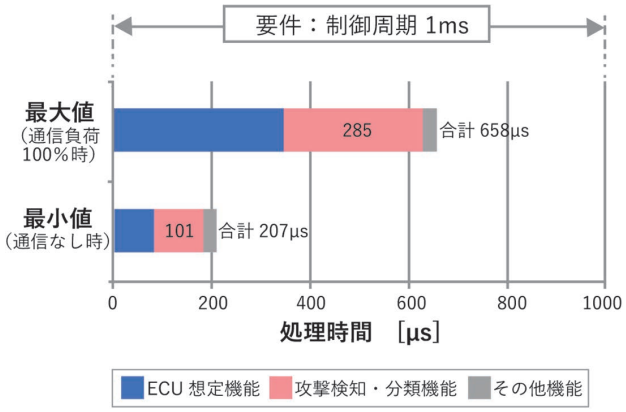

図 9　車載 ECU 想定モデルにおける処理時間

以上から、今回実装したアルゴリズムは、低リソースな組込機器においても本来の機能を損なわずに攻撃検知・分類できる、実用可能なレベルの技術であるといえる。

8. むすび

組込機器上でリアルタイムに攻撃検知・分類を行うという課題に対し、組込機器上でも実行可能である高速・軽量な攻撃検知・分類アルゴリズムを開発した。また、攻撃をより細かく分類し、適切な対策が選択できるように攻撃検知・分類アルゴリズムを改良した。

さらに、開発したアルゴリズムを実際の組込機器ハードウェア上に実装して機能と性能を確認した。結果、細分化した分類を実現しつつ、リアルタイムに実行できる性能を持っていることを確認できた。

本技術を組込機器に搭載することで、セキュリティ攻撃を受けてから緊急対応までを即座に実行できる。さらに、攻撃分類を適切な粒度まで細分化できるため、受けた攻撃の種類に応じた緊急対応を選択し、自動的に実行させることも可能となる。

今後、攻撃の種類に応じた緊急対応を適切に設定することができれば、組込機器へのセキュリティ攻撃による被害を最小化できるようになる。

また今回は車載 ECU 上に実装するため、異常の検出方法やスケジューリング等、車載 ECU に特化した開発を行った。今後は、車載 ECU 以外の領域・分野の組込機器に向けても、本技術を展開していきたい。

参考文献

1) Miller, C.; Valasek, C. “Remote Exploitation of an Unaltered Passenger Vehicle”. http://ilmatics.com/Remote Car Hacking.pdf, (参照 2019-11-27)
2) 阿部 慎吾. “制御システムに関する最近の脅威と JPCERT/CC の取り組み”. 第 5 回 制御部門マルチシンポジウム. http://mscs2018.sice-ctrl.jp/program/_tutorials/mscs2018tutorial_abe.pdf
3) 米国国立標準技術研究所. “Framework for Improving Critical Infrastructure Cybersecurity”. https://www.nist.gov/cyberframework/framework, (参照 2019-11-27)
4) Blue Planet-works. 製品紹介 AppGuard. https://www.blueplanet-works.com/solution/appguard.html
5) 廣部 直樹. “汎用組込機器向け攻撃検知・分類技術の紹介”. 第 4 回 IoT セキュリティフォーラム.
6) マイクロソフト. “The STRIDE Threat Model”. https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878 (v=cs.20), (参照 2020-01-08)
7) 小河原 徹. “攻撃の遷移に着目した組込機器向け攻撃検知・分類技術の提案”. SCIS2020.

執筆者紹介

小河原　徹　KOGAWARA Toru
技術・知財本部
研究開発センタ
専門：ソフトウェア科学

山本　泰生　YAMAMOTO Taisei
技術・知財本部
研究開発センタ
専門：電気電子情報工学

曾　珍　ZENG Zhen
技術・知財本部
研究開発センタ
専門：ソフトウェア工学

廣部　直樹　HIROBE Naoki
技術・知財本部
研究開発センタ
専門：ソフトウェア工学