スリープ状態をもつ組込みシステムを対象とした電力解析にもとづく異常動作検知とその実証的評価

長谷川 健人[1,a)]　近松 聖[2]　戸川 望[1]

概要：低コストで高機能な IoT デバイスが普及する昨今，IoT デバイスを対象としたサイバー攻撃が懸念される．IoT デバイスなどの組込み機器では，ハードウェアの設計・製造段階で挿入されるハードウェアトロイや，ファームウェアアップデートによるマルウェア感染の危険性が指摘されており，組込みシステムにおける脅威を早期に検出する技術の開発は急務である．本稿では，スリープ状態をもつ組込みシステムを対象とした電力解析にもとづく異常動作検知手法を提案し，実デバイスを用いて提案手法を実証的に評価する．まず，マイクロコントローラを搭載した組込みシステムに対し消費電力を測定し，平常時におけるスリープ状態とアクティブ状態の区間を教師なし学習で識別する．次に，アクティブ状態の区間に着目してその継続時間とその間の消費エネルギーを算出し，特徴量を抽出する．抽出した特徴量にもとづき，外れ値検知アルゴリズムを適用して異常動作を検知する．複数のマイクロコントローラを用いた実験により，提案手法の有効性を確認した．

1. はじめに

近年，我々の日常では数多くの Internet of Thing (IoT) デバイスが利用されている．第五世代移動通信システム (5G) や Society 5.0，Industry 4.0 が広く関心を持たれる中で，IoT 技術は今後もさらなる急速な発展が見込まれる．こうした状況のもと，IoT 技術における脅威が指摘されている．近年ではハードウェア設計・製造において海外企業への外部委託や Intellectual Property (IP) の利用により，ハードウェア製品を安価に容易に製造することが可能となっている．外部委託の再委託や製造コストを抑えるために導入した安価な IP などにより，設計・製造工程の一部に信頼されない工程が混入する危険性が指摘されている [1]．ここでは，ハードウェアデバイスにおけるセキュリティ課題に着目し，議論する．

IoT デバイスにおいてその動作の信頼性を担保するために，そのデバイスの主たる通信経路を常時監視する方法が用いられる．しかし，総務省の調査によれば 2020 年までに 400 億以上の IoT デバイスが普及すると予想されており [2]，すべての主たる通信路を直接監視することは難しい．対して，クラウドレベルに対し比較的末端に近いエッジレベルでは，個々の IoT デバイスを監視する．一度エッジレベルにおいて IoT デバイスにおける異常動作を監視する仕組みを構築できれば，莫大な量のトラフィックを監視するための負荷を軽減することができる．

それぞれの IoT デバイスではしばしば固有のマイクロコントローラやオペレーティングシステム (OS)，組込みハードウェア・ソフトウェアが利用されており，これらに適応したエッジレベルの監視システムを構築することは難しい．この課題を解決するため，個々の IoT デバイスに特化した監視の仕組みを構築するのではなく，電圧や電流，消費電力と言ったサイドチャネル情報を利用する方法が挙げられる．サイドチャネル情報を効果的に解析することができれば，IoT デバイスなどの小規模なプラットフォーム上でも監視アルゴリズムを構築することができる．例えば消費電力に着目した監視システムを考えたとき，消費電力の推移から異常な動作を検知することができれば，それは IoT デバイスにおけるマルウェア検知の重要な手がかりとなり得る．

文献 [3] では，サイドチャネル情報にもとづくハードウェアトロイ (ハードウェアに挿入された悪意のある機能) 検知手法が提案されている．この手法ではサイドチャネル情報にもとづくハードウェアトロイ検知を可能とする一方で，ハードウェアトロイが挿入されていないことが保証された ‘ゴールデン回路’ を必要とする．サプライチェーンが複雑化した昨今においてゴールデン回路を確保することは非常に困難であり，現実的でない．そのため，本稿では，

[1] 早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻
Dept. of Computer Science and Communications Engineering, Waseda University
[2] キーサイト・テクノロジー・インターナショナル合同会社
Keysight Japan G.K.
[a)] kento.hasegawa@togawa.cs.waseda.ac.jp

© 2019 Information Processing Society of Japan

ゴールデン回路を必要としない，サイドチャネル情報にもとづく異常動作検知手法を提案する．文献 [4] では，ハードウェアに挿入された悪意のある機能はごく稀な条件下でのみ動作すると仮定するため，異常動作の検知を目的とする．この条件下で対象とする IoT デバイスの消費電力を計測して通常の状態を学習し，異常動作を検知する手法を提案している．この文献では限られた条件で実験がなされているのみであるため，本稿ではより高機能なハードウェアを対象とした異常検知を実験的に確認する．

本稿では，スリープ状態をもつ組込みシステムを対象とした，電力解析にもとづく異常動作検知手法を提案する．提案手法では，消費電力から特徴量を抽出し，この特徴量にもとづき通常状態といくつかの異常動作を予め学習する．次に，学習した特徴量にもとづき，異常動作を識別する．

本稿の貢献を以下に示す．

(1) 精細な電力解析にもとづく，組込みシステム，とりわけマイクロコントローラを対象とした異常動作検知手法を提案する．提案手法では，対象とするハードウェアの通常の動作を教師なし学習で学習する．
(2) 実験により，複数の組込みシステムに対し提案手法を用いて異常動作の検知することで，悪意のある機能の動作の検知に成功した．

本稿は以下のように構成される．2 章で，本稿で対象とする脅威モデルと，組込みシステムを明らかにする．3 章で，スリープ状態をもつ組込みシステムを対象とした，電力解析にもとづく異常動作検知手法を提案する．4 章で，複数の組込みシステムを対象に実験する．5 章で，本稿をまとめる．

2. 対象とする脅威モデル

本章では，本稿で対象とする脅威モデルを明らかにする．

ハードウェアにおける悪意のある動作の検知のため，本稿では 1) ゴールデン回路を利用しないことと，2) 事前に詳細なプロファイル作成を必要としないことに着目する．

2.1 マイクロコントローラに挿入された悪意のある機能

ハードウェアデバイスは，抵抗器や容量，IC チップなどから構成される．特に，ASIC や FPGA，マイクロコントローラは数値計算やデータ管理などアプリケーションにおける主要な処理を実現する．そのため，攻撃者にとって Application Specific Integrated Circuit (ASIC) や Field-Programmable Gate Array (FPGA)，マイクロコントローラは都合の良い攻撃対象となり得る．これらに挿入された悪意のある機能は，一般にハードウェアトロイと呼ばれる．いかにハードウェアトロイを検出するかは近年のハードウェアセキュリティにおいて大きな課題となっている．ASIC や FPGA に関して，これらは物理的な回路を構成してシステムの機能を実現するため，それらの特徴に合わせたハードウェアトロイ検知手法が提案されている．一方で，マイクロコントローラでは中央演算処理装置 (CPU) やメモリを利用してメモリ上に格納された命令を実行するため，ASIC や FPGA を対象としたハードウェアトロイ検知手法をそのまま適用できない．一般にマイクロコントローラで実行されるソフトウェアはアセンブラや C 言語，C++が利用されており，特に近年では組込みシステム向けのスクリプト言語である Micro Python も利用される．マイクロコントローラはソフトウェアで機能が実装されるという点で，ソフトウェアを書き換えるだけで機能を更新できるため，開発者だけでなく攻撃者にとっても取り扱いが比較的容易である．加えて近年では IoT デバイスのファームウェアアップデートを自身で自動的に適用する仕組みも開発されており，IoT デバイス自身で最新のファームウェアを取得，更新することが可能となっている．もしこの仕組みが悪用されれば，システムに重大な不具合を引き起こす原因となり得る．このようにハードウェアデバイスを利用するときに，マイクロコントローラを利用するシステムを考慮した悪意のある機能の検知手法を開発する必要がある．

一般に，悪意のある機能は普段その存在を隠すため，悪意のある機能を起動するためのトリガ条件が設定される．このトリガ条件が満たされた場合にだけ，悪意のある機能が起動する．トリガ条件としては，特定の入力を受け取った場合や，特定の時間が経過した場合などが考えられる．トリガ条件をもつ悪意のある機能はハードウェア出荷時のテスト工程や通常の運用で見つけることが困難である．本稿では，こうしたトリガ条件をもつ悪意のある機能を対象として，電力解析により異常動作を検知する手法を提案する．

2.2 マイクロコントローラの動作状態

本稿では，マイクロコントローラを利用した小規模な IoT デバイスを対象とする．小規模な IoT デバイスでは電力の消費量を抑えるため，省電力化のための機能をもつ．マイクロコントローラにはこれを実現するため，システムが主要な動作を必要としない場合に消費電力を抑える機能がある．本稿では，これを ‘スリープ状態’ と呼ぶ．マイクロコントローラにおけるスリープ状態では，タイマやクロックなど再起動するために必要最小限な機能以外は停止する．スリープ状態では，mW 以下程度の電力を消費する．スリープ状態に対して，マイクロコントローラがシステムの機能を実行する状態を ‘アクティブ状態’ と呼ぶ．小規模な IoT デバイスの例としてセンサロガーを挙げれば，システムは一定時間ごとにセンサ情報を取得し外部に送信する場合，ごくわずかな時間アクティブ状態になった後にスリープ状態となる動作を一定期間ごとに繰り返す．アクティブ状態では，mW 程度の電力を消費する．それぞれの

© 2019 Information Processing Society of Japan

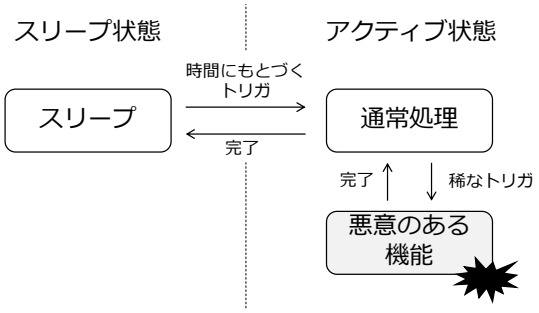

図 1 マイクロコントローラにおける脅威モデルの概要.

動作状態においてマイクロコントローラの処理に応じて消費電力はわずかに変化するが，アクティブ状態とスリープ状態では大きく変化する．本稿ではこの電力の違いを利用する．

次に，攻撃者によってマイクロコントローラに悪意のある機能が挿入された場合を考える．図 1 に，マイクロコントローラの動作状態の概要を示す．スリープ状態では，マイクロコントローラは最小限の動作しかしない．アクティブ状態では，システムに必要な機能が動作する．ここで，攻撃者によって悪意のある機能が挿入された場合を考えると，マイクロコントローラが動作しなければ悪意のある機能は動作できない．したがって，悪意のある機能があるトリガ条件によって動作するとしても，アクティブ状態中に動作する．

本稿では，スリープ状態をもつ組込みシステムを対象として，電力解析にもとづき悪意のある機能の動作を検知する．文献 [4] によれば，マイクロコントローラを対象とした異常動作検知手法が提案されている．本稿ではより精細な電力解析にもとづき，複数のマイクロコントローラを対象とした異常動作検知手法を提案する．

3. 電力解析にもとづく悪意のある機能の発現検知

本章では，消費電力解析にもとづく，マイクロコントローラに挿入された悪意のある機能の発現検知手法を提案する．提案手法は，以下の 5 つの処理から構成される．

(1) **電力測定**: 測定機器を用いて対象となる IoT デバイスの消費電力を計測する．
(2) **波形整形**: 取得した消費電力の測定波形を平滑化する．
(3) **アクティブ/スリープ状態識別**: 教師なし学習を用いて，消費電力波形をアクティブ状態の区間とスリープ状態の区間に識別する．
(4) **特徴量抽出**: アクティブ状態と識別された区間に対し，その継続時間とその間の消費エネルギーを算出し，特徴量として抽出する．
(5) **外れ値検知**: 抽出された特徴量に対し外れ値検知アルゴリズムを適用し，異常な区間を検出する．検出された区間では，悪意のある機能が発現したと識別される．

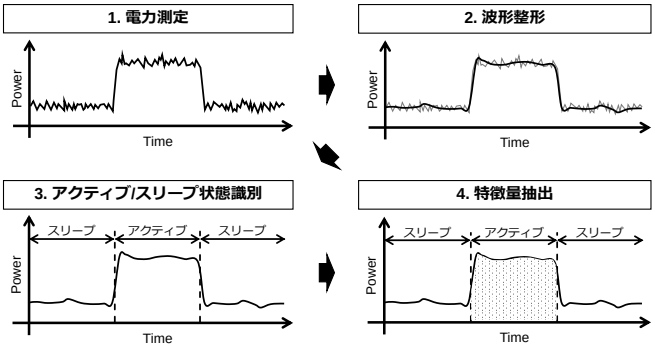

図 2 電力解析にもとづく特徴量抽出までの流れ [4].

3.1 Step 1: 電力測定

まず，対象の IoT デバイスから消費電力を時系列的に計測する．消費電力 P は電圧 V と電流 I を同時計測し，$P = V \times I$ により得る．一般に，電圧の計測には電圧プローブが，電流の計測にはクランプ型の電流プローブが利用される．しかし，電流については IoT デバイスの場合アクティブ・スリープ間の電流変動比が大きく，上記電流プローブではダイナミックレンジや感度の点で不十分な場合もあり必ずしも適用できるものではない．加えて，近年動作周波数が 100MHz を超えるデバイスもあるため，広帯域かつ高精細に電流を計測可能な測定技術が必要となる．そのため，本稿では高精細測定に向いたシャント抵抗センシングをベースに発展させた電流センサ (4.1 節参照) を利用している．

3.2 Step 2: 波形平滑化

Step 2 では，移動平均を用いて波形を平滑化する．時刻 n における時系列データ $x[n]$ に対し，N 次の移動平均を取る場合，移動平均により平滑化されたデータ $y[n]$ は，$y[n] = \frac{1}{N}\sum_{i=0}^{N-1} x[n-i]$ で示される．本稿では $N = 5$ として 5 次の移動平均をとる．

3.3 Step 3: アクティブ/スリープ状態識別

Step 2 で整形した波形に対し，アクティブ状態の区間かスリープ状態の区間かを識別する．ここでは教師なし学習によるクラスタリングアルゴリズムの一種である k 平均法 [5] を利用する．本稿では，取得した消費電力をもとに，アクティブ状態とスリープ状態の二つに識別する．k 平均法ではパラメータとして分割するクラスタ数だけが必要である．ここではアクティブ状態とスリープ状態の二つに分割すれば良いため，他のパラメータ設定は不要である．

3.4 Step 4: 特徴量抽出

Step 4 では Step 3 で識別されたアクティブ状態区間の消費電力波形から継続時間とその間の累積消費電力，すなわち消費エネルギーを特徴量として抽出する．2 章で議論

© 2019 Information Processing Society of Japan

したように，悪意のある機能はアクティブ状態の区間においてトリガ条件を満たした場合にのみ動作する．悪意のある機能として Denial-of-Service (DoS) 攻撃や内部情報を流出する機能が動作する場合，アクティブ状態の継続時間や消費電力はアクティブの状態と比較して明らかに変化するはずである．このような点から，本稿では異常動作検出のための特徴量としてアクティブ状態の継続時間とその間の消費エネルギーを特徴量として抽出する．

3.5 Step 5: 外れ値検知

Step 4 で抽出した特徴量にもとづき，外れ値検知アルゴリズムを適用して悪意のある機能が発現したアクティブ状態の区間を検出する．本稿では，外れ値検知アルゴリズムとして局所外れ値因子 (LOF) 法 [6] を利用する．LOF 法では，周囲と比較して密度が極端に異なるサンプルを外れ値として識別する．本稿で抽出されるアクティブ状態区間における継続時間と消費エネルギーは，区間ごとにわずかに異なる．しかし，もし悪意のある機能が動作すればこれらの特徴量が他の区間よりも変化するはずである．悪意のある機能はごく稀な条件でのみ動作するので，これらの特徴量が分布する空間は密度が低いはずである．LOF 法を利用することで，これらを効果的な検出が期待される．

LOF 法を適用する前に，まず取得した特徴量を標準化する．特徴量のうち継続時間に着目して，i 番目のアクティブ状態区間の継続時間を t_i と置く．N 個分のアクティブ状態区間における継続時間の平均値と分散値をそれぞれ μ_N, σ_N とする．このとき，標準化された i 番目の継続時間 t'_i は，$t'_i = (t_i - \mu_N)/\sigma_N$ によって得られる．消費エネルギーも同様に標準化する．このように標準化された値に LOF 法を適用する．LOF 法では，それぞれのサンプルに対し LOF 値として外れ度合いを示す値が得られる．この値にもとづき，その区間が異常か否かを識別する．異常であると判定されれば，その区間で悪意のある機能が発現したものとして検出する．

4. 実験結果

本章では，3 章で提案した手法にもとづき，実際の装置を用いて実験した結果を示す．

4.1 セットアップ

図 3 に実験で使用した機器の概要を示す．図 4 に実験装置の様子を示す．電源装置として Keithley 2280S-32-6 [7] を使用し，この電源装置から対象となるマイクロコントローラに電源を供給する．実験では電源装置の出力電圧を 5V とし，最大電流を 0.4A と設定した．電流と電圧の測定には Keysight CX3324A [8] を使用した．電圧はパッシブプローブを使用し，電流は電流センサ Keysight CX1101A を使用して測定した．

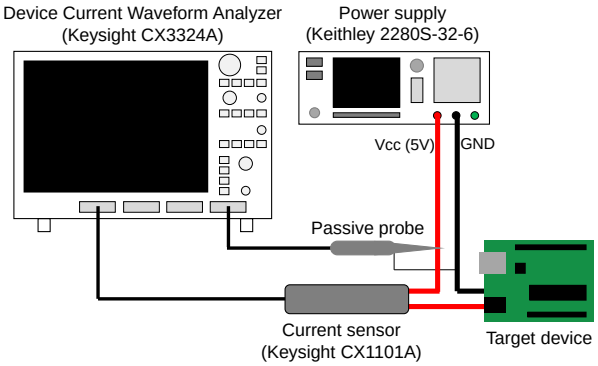

図 3 実験機器の概要．

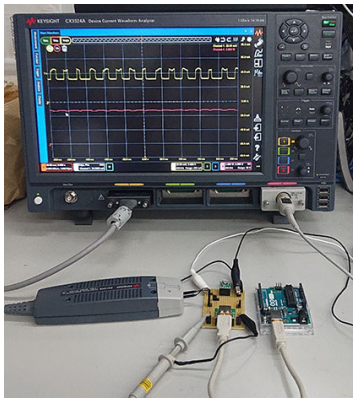

図 4 実験装置の様子．

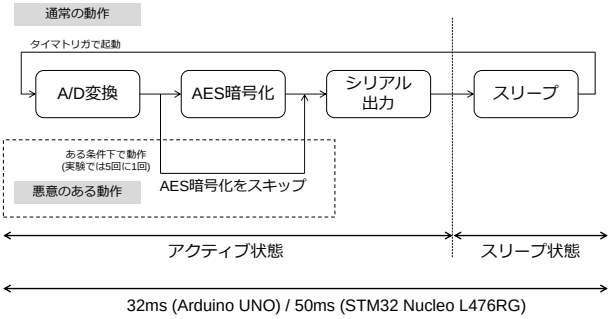

図 5 マイクロコントローラで動作するアプリケーションの概要．

文献 [9] に示されるように，クランプ型の電流プローブの最小測定電流は 1–3mA 程度である．これに対し，本実験で使用した電流センサは 100MHz 帯域で，0.41Ω のシャント抵抗により最大 1A から最小 3μA まで測定できる [8]．このセンサではシャント抵抗方式と電流トランス方式の 2 つの方式を組み合わせる新しい電流センス方式を用いたことで広帯域と感度を両立している [10]．デバイスの電源回路に挿入されるシャント抵抗値が 0.41Ω と小さいため急激な電流変化に対してもデバイスの電源電圧降下を抑えることができる．

上記の実験環境で電流を測定する際，測定結果をコンピュータに転送する．解析には Python 3 と機械学習のライブラリ scikit-learn [11] を利用して実装されたプログラムを利用した．

© 2019 Information Processing Society of Japan

4.2 実験で使用するマイクロコントローラとアプリケーション

実験では，センサロガーを模したアプリケーションを実装した．本稿では Arduino UNO と Nucleo L476RG，2 つのマイクロコントローラを利用する．アプリケーションの動作の概要を図 5 に示す．まず，マイクロコントローラは A/D 変換機能を利用して，マイクロコントローラのピンに印加された電圧を取得する．次に，取得した電圧を AES 暗号化 [12] する．AES 暗号化されたデータはシリアル通信を利用して外部に送信される．上記がアクティブモードの一連の流れであり，これが完了するとマイクロコントローラはスリープモードに移行する．本稿では，上記の一連の動作を Arduino UNO で 32ms ごと，Nucleo L476RG で 50ms ごととして実装した．

ここで，悪意のある機能を上記のアプリケーションに実装する．本稿では AES 暗号化に対して 5 回に 1 回無効化するような機能を実装した．悪意のある機能が有効になると，AES 暗号化処理が実行されずに A/D 変換した結果が平文のまま出力される．

4.3 実験結果

Arduino UNO

Arduino UNO を対象とした実験結果を図 6，図 7 と表 1 に示す．図 6 では，Step 1 で取得された消費電力波形を示しており，縦軸は消費電力，横軸は時間を示す．図 7 では，横軸はアクティブモードの継続時間を，縦軸はその間の消費エネルギーを示す．背景は LOF 値を示しており，色が濃いほど LOF 値が低く，異常度が高いことを示す．

図 6 より，ところどころアクティブ状態となっている部分の継続時間が短くなっており，視覚的に異常な状態を検出することができる．実際，特徴量を抽出した図 7 では，はっきりと右上と左下の二か所に特徴量が分布する．右上の特徴量はアクティブモードの継続時間が長く消費エネルギーも大きくなっており，LOF 値は大きく異常度が小さくなっている．一方で左下の特徴量は右上の特徴量よりも LOF 値が小さく，異常度が大きくなっている．表 1 では，それぞれの特徴量 (継続時間と消費エネルギー) における LOF 値を示す．LOF 値に下線が引かれたものが，図 7 において左下に分布した特徴量である．図や表から明らかなように，Arduino UNO を対象とした実験では LOF 値において-100 を閾値として設定することで明確に異常な振る舞いを検出できる．Arduino UNO は動作周波数が 16MHz となっており，比較的低速なため CMOS のスイッチングノイズも小さい．そのため，本実験では悪意のある機能の動作を明確に検出することに成功した．

Nucleo L476RG

Nucleo L476RG を対象とした実験結果を図 8，図 9 と表 2 に示す．図 8 では，Step 1 で取得された消費電力波形を

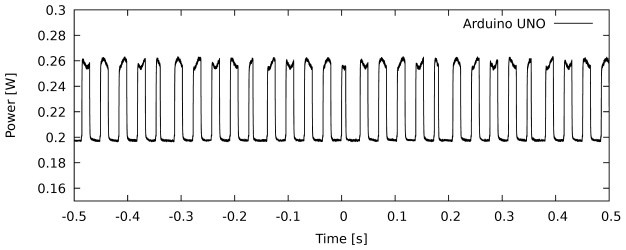

図 6　Arduino UNO における消費電力波形.

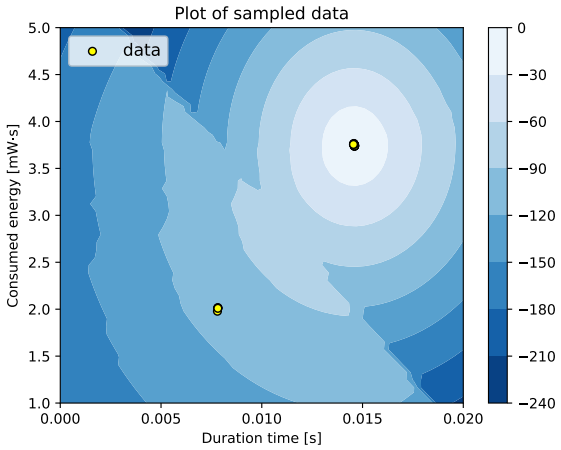

図 7　Arduino UNO における実験結果の特徴量分布.

表 1　Arduino UNO における実験結果.

区間	継続時間 [s]	消費エネルギー [mW · s]	LOF 値
1	0.01461	3.74	-1.01
2	0.01457	3.76	-1.02
3	0.01457	3.76	-1.00
4	0.01459	3.74	-1.10
5	0.00782	2.02	-109.57
6	0.01457	3.77	-1.07
7	0.01458	3.75	-0.94
8	0.01461	3.74	-1.09
9	0.01458	3.75	-0.94
10	0.00782	2.01	-109.67
11	0.01459	3.75	-0.98
12	0.01461	3.74	-1.06
13	0.01458	3.75	-0.94
14	0.01457	3.76	-1.01
15	0.00780	1.98	-110.45
16	0.01460	3.74	-0.99
17	0.01457	3.76	-1.02
18	0.01457	3.76	-0.99
19	0.01460	3.74	-1.10
20	0.00782	2.02	-109.57
21	0.01456	3.76	-1.02
22	0.01459	3.75	-0.90
23	0.01460	3.74	-1.14
24	0.01459	3.76	-0.94
25	0.00782	2.01	-109.69
26	0.01459	3.75	-0.98
27	0.01460	3.74	-1.09
28	0.01458	3.76	-0.99
29	0.01455	3.76	-1.01

示しており，縦軸は消費電力，横軸は時間を示す．図 9 では，横軸，縦軸，背景はそれぞれ図 7 と同様である．表 2 では，LOF 値に下線を引かれたものが悪意のある動作が発現した時の結果である．

© 2019 Information Processing Society of Japan

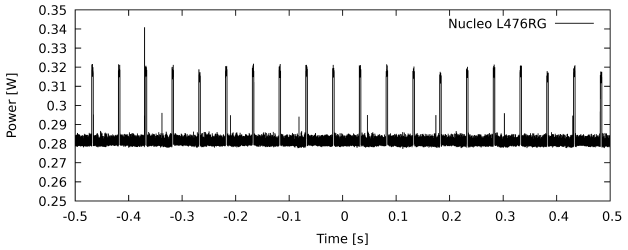

図 8　Nucleo L476RG における消費電力波形.

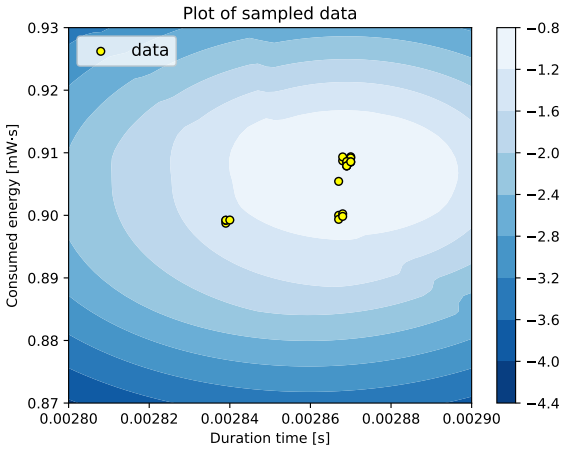

図 9　Nucleo L476RG における実験結果の特徴量分布.

表 2　Nucleo L476RG における実験結果.

区間	継続時間 [s]	消費エネルギー [mW · s]	LOF 値
1	0.002868	0.9087	-0.99
2	0.002839	0.8987	-1.35
3	0.002870	0.9094	-0.99
4	0.002869	0.9086	-0.99
5	0.002867	0.9000	-0.99
6	0.002868	0.9093	-0.99
7	0.002839	0.8992	-1.34
8	0.002869	0.9080	-0.99
9	0.002870	0.9092	-0.99
10	0.002869	0.9086	-0.99
11	0.002870	0.9085	-0.99
12	0.002839	0.8992	-1.34
13	0.002867	0.9054	-0.99
14	0.002867	0.8994	-0.99
15	0.002869	0.9079	-0.99
16	0.002869	0.9079	-0.99
17	0.002840	0.8993	-1.31
18	0.002868	0.9002	-0.99
19	0.002870	0.9086	-0.99
20	0.002868	0.8998	-0.99

図 8 に示されるように，Nucleo L476RG では Arduino UNO と比較してアクティブ状態の継続時間が非常に短く，この波形だけから視覚的に異常を検出するのは困難である．そこで提案手法を利用することで，表 2 に示されるように，LOF 値に対し閾値を-1 として設定することで悪意のある動作を異常動作として検出することに成功した．図 9 では，異常な動作は左下のクラスタに当たり，他のクラスタよりも LOF 値が低いことが確認できる．以上に示されるように，Nucleo L476RG での正常な動作と異常な動作との差はごくわずかである．これは Nucleo L476RG が 80MHz の動作周波数であることから Arduino UNO と比較し高速であり，消費エネルギーだけを利用した正常な動作と異常な動作の判別が難しいことが一因である．したがって，提案手法のように精細な電力解析にもとづく，アクティブモードの継続時間と消費エネルギーの両方を考慮した解析が必要となる．

5. まとめ

本稿では，スリープ状態をもつ組込みシステムを対象として電力解析にもとづく異常動作検知手法を提案し，それを実証的に評価した．実験ではセンサロガーを模した二種類の IoT デバイスを実装し，精細な電力解析を利用して提案手法を適用した．実験の結果，最大 80MHz で動作するマイクロコントローラに対しても，提案手法により異常な動作を検知することに成功した．

今後の課題として，より高機能なマイクロコントローラや複雑な動作をする IoT デバイスなど，電力解析がより困難なアプリケーションを対象とすることが挙げられる．

参考文献

[1] J. Francq and F. Frick, “Introduction to hardware Trojan detection methods,” in *Proc. Design, Automation and Test in Europe (DATE)*, 2015, pp. 770–775.
[2] 総務省, “情報通信統計データベース.” http://www.soumu.go.jp/johotsusintokei/index.html
[3] R. Shende and D. D. Ambawade, “A side channel based power analysis technique for hardware trojan detection using statistical learning approach,” in *Proc. International Conference on Wireless and Optical Communications Networks (WOCN)*, 2016, pp. 1–4.
[4] K. Hasegawa, M. Yanagisawa, and N. Togawa, “Detecting the existence of malfunctions in microcontrollers utilizing power analysis,” in *Proc. IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2018, pp. 97–102.
[5] J. Macqueen, “Some methods for classification and analysis of multivariate observations,” in *Proc. Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14, 1967, pp. 281–297.
[6] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LOF: identifying density-based local outliers,” in *Proc. ACM SIGMOD International Conference on Management of Data*, 2000, pp. 93–104.
[7] Tektronix, “Keithley 2280 Precision Measurement DC Power Supplies.” https://www.tek.com/tektronix-and-keithley-dc-power-supplies/keithley-2280s-series
[8] Keysight Technologies, “CX3300 Series Device Current Waveform Analyzer Datasheet.” https://literature.cdn.keysight.com/litweb/pdf/5992-1430EN.pdf?id=2727780
[9] Keysight Technologies, “Evaluating current probe technologies for low-power measurements.” http://literature.cdn.keysight.com/litweb/pdf/5991-4375EN.pdf
[10] K. Chikamatsu, “Current sensing circuit,” U.S. Patent 9 689 900, 2017.
[11] F. Pedregosa *et al.*, “Scikit-learn: machine learning in python,” *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[12] “AESLib.” https://github.com/DavyLandman/AESLib

© 2019 Information Processing Society of Japan