3F-4

入出力アドレスのランダム化による制御プログラム難読化方式の一提案

橋本 遼太† 勝田 喬雄† 三浦 昭浩† 古澤 康一†
三菱電機株式会社 情報技術総合研究所†

1. はじめに

近年、組込み機器に対するセキュリティ上の脅威が増加している。その一例として、組込み機器から不正に読み出された制御プログラムの閲覧／編集が行われ、ノウハウが漏えいしたり、不正コピーした制御プログラムを用いた組込み機器の模倣品が製造され市場に出回ったりする問題があり、早急な対策が求められている。

制御プログラムの不正な閲覧／編集を防ぐ方法の一つとして、制御プログラムの難読化がある[1]。難読化を適用した制御プログラムはそのままで実行可能だが、閲覧／編集は困難である。よって、難読化を適用した制御プログラムを実行する際に特別な処理が必要ない。そのため、ファームウェアの更新が難しい組込み機器に対しても、適用が容易である。しかし、難読化を適用した制御プログラムは不正コピーが正常に動作してしまうので、そのままでは模倣品の製造を防止できない。

制御プログラムは、組込み機器が制御する周辺機器の入出力に割り当てられた識別番号である入出力アドレスを用いて、周辺機器からの入力に基づく演算処理を実行し、その結果を出力して、周辺機器を制御している。このため、制御プログラムを閲覧／編集するためには、入出力アドレスを正しく把握する必要がある。

本稿では、制御プログラムの入出力アドレスの割り当てをランダム化し、その際に組込み機器の固有情報を利用して組込み機器と制御プログラムを関連付ける難読化方式を提案する。本方式により入出力アドレスを正しく把握できなくなるので、制御プログラムの不正な閲覧／編集を困難にすることができる。また、制御プログラムを他の組込み機器に不正コピーした場合は、固有情報が異なるので難読化を正しく解除できないようにし、正常動作できなくすることにより、模倣品の製造を防止することが可能となる。

2. 提案方式の概要

提案方式は、制御プログラム内の入出力アドレスをランダムに入れ替え、異なる入出力アドレスに変換することで、制御プログラムと周辺機器との入出力の関係を複雑にし、閲覧／編集を困難にする。入出力アドレスを変換する際は、変換テーブルを使用する。制御プログラムの実行時は、組込み機器のファームウェアと連携して難読化した制御プログラムを実行する。以下、変換テーブルの作成方法と入出力アドレスの変換方法及び難読化した制御プログラムの実行方法の詳細を述べる。

2.1. 変換テーブルの作成方法

図 1 に変換テーブルの作成方法の概要を示す。変換テーブルは組込み機器を一意に識別するための固有情報と、付加情報を使用して作成する。固有情報の例としては MAC アドレスなどが考えられる。固有情報だけを用いて変換テーブルを作成すると、常に同じ値の変換テーブルになってしまうため付加情報（例えばタイムスタンプ）と組み合わせることで、毎回異なる変換テーブルを作成する。変換テーブルの作成手順は以下の通りである。

(1) 組込み機器で利用可能な入出力アドレスをインデックスとして配列を作成する。
(2) 組込み機器から固有情報を取得する。
(3) 付加情報を作成する。
(4) 固有情報と付加情報から乱数を作成する。
(5) 作成した乱数を使用して変換後の数値に該当する配列をランダムソートする。

上記のように、変換後の入出力アドレスが難読化対象の組込み機器で実際に割り当てられている入出力アドレスになるように変換テーブルを作成する。これにより、難読化した制御プログラムの難読化を解除しなくても何らかの動作はするので、制御プログラムとして妥当なものになることから、既存の難読化方式と組み合わせて制御プログラムを保護することが可能となる。

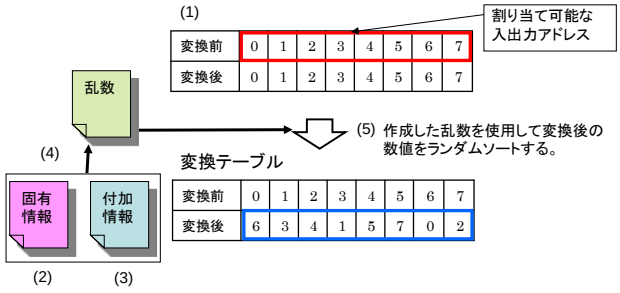

図 1 変換テーブル作成方法

“A proposal of obfuscation method by randomizing the I/O addresses of the control program”
† Information Technology R&D Center, Mitsubishi Electric Corporation

2.2. 入出力アドレスの変換方法

入出力アドレス変換の流れは、まず制御プログラムから入出力アドレスを一つ読み出し、変換テーブルを参照し、入出力アドレスを変換する。この入出力アドレスの変換処理を全ての入出力アドレスを変換し終えるまで繰り返す。

図2に入出力アドレスの変換例を示す。SWITCH_INPUT_0のアドレス0x00は変換テーブルを参照すると0x06になる。よって、変換後にSWITCH_INPUT_0を操作しようとするとアドレス0x06の入出力を操作することになるので、実際には、LED_OUTPUT_2を操作することになる。

次に、入出力アドレスの変換によるコードの変換例を、図3のサンプルコードを用いて説明する。入出力アドレス変換前のサンプルコードの動作はSWITCH_INPUT_0がONになるとLED_OUTPUT_0がONになり、SWITCH_INPUT_1がONになるとLED_OUTPUT_1がONになる。入出力アドレス変換後の制御プログラムの動作は前述の変換規則に従うと、実際の入出力の動作は、LED_OUTPUT_2がONになるとLED_OUTPUT_1がONになり、SWITCH_INPUT_3がONになるとLED_OUTPUT_3がONになる。このように、入出力アドレスの変換により実行結果が変化する。

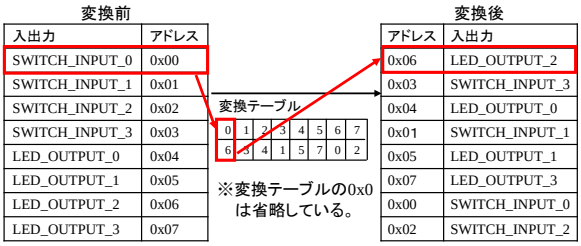

図 2 入出力アドレス変換の概要

変換前

```
void main(void){
  if( SWITCH_INPUT_0 == ON
    LED_OUTPUT_0 =  ON ;
  }
  if( SWITCH_INPUT_1 == ON ){
    LED_OUTPUT_1 =  ON ;
  }
  return ;
}
```

変換後の実際の動作

```
void main(void){
  if( LED_OUTPUT_2 == ON ){
    LED_OUTPUT_1 =  ON;
  }
  if( SWITCH_INPUT_3 == ON ){
    LED_OUTPUT_3 =  ON;
  }
  return ;
}
```

図 3 サンプルコード

2.3. 難読化した制御プログラムの実行方法

組込み機器は制御プログラムを実行する際、制御プログラムの難読化を解除してから実行する。難読化の解除には、固有情報と付加情報が必要になる。固有情報は、難読化時に組込み機器から読み出した情報なので、組込み機器内で読み出すことができる。しかし、付加情報は難読化時に作成した情報なので、難読化した制御プログラムを組込み機器に送信する際に、一緒に送信しておく必要がある。制御プログラム実行時の手順は以下の通りである。

(1) 固有情報と付加情報を読み出して、変換テーブルを作成する。
(2) 変換テーブルを使用して、難読化したときと逆の変換を行い、難読化を解除する。
(3) 難読化を解除した制御プログラムを実行する。

手順(1)で作成する変換テーブルは、難読化時と同じ情報を使用して作成することにより、難読化時に作成した変換テーブルと同一のものになるので、正しく難読化を解除可能である。

3. 提案方式の効果

提案方式の効果として、以下が挙げられる。

(1) 制御プログラムの閲覧／編集の困難化

制御プログラムと周辺機器との入出力の関係は制御プログラムの閲覧／編集を行うためには重要な情報となる。よって、入出力アドレスの入れ替えにより、攻撃者が制御プログラムを取得しても、その閲覧／編集は困難になる。

(2) 制御プログラムの模倣品製造の防止

固有情報の異なる組込み機器や、正規の組込み機器（難読化の解除ができるファームウェアを搭載した組込み機器）以外の組込み機器で制御プログラムを動作させた場合は、難読化を正しく解除できず、正常動作できないため、模倣品の製造を防止できる。

(3) 難読化方式の解析の困難化

タイムスタンプなどの付加情報を使用することで変換テーブルにランダム性を持たせており、同じ制御プログラムを同じ組込み機器と対応付けて難読化した際にも、異なる難読化結果を得ることができるので、難読化方式の解析が困難になる。

(4) 他の難読化方式との組み合わせ可能化

難読化した制御プログラムは制御プログラムとして妥当なものとして扱えるので、本方式を既存の難読化方式と組み合わせて使用して制御プログラムを保護することも可能である。本方式を既存の難読化方式と組み合わせて使用することで、模倣品の製造を防止しながら、本方式のみを適用した場合よりも不正な閲覧／編集を困難にすることが可能である。

4. 終わりに

本稿では、制御プログラムの不正な閲覧／編集を困難にし、模倣品の製造を防止する方式の一つとして、組込み機器の固有情報と関連付けて、入出力アドレスをランダム化する難読化方式について提案した。

今後は、提案方式の実装及び評価を進めていく。

参考文献

[1] 門田暁人, C.Thomborson , “ソフトウェアプロテクションの技術動向（前編）”, 情報処理, Vol46, No.4, pp.431-437, Apr. 2005.