軽量暗号化アルゴリズムに対する可逆プログラミング言語 Hermes の有効性の評価

2018SE076 佐々木龍之介
指導教員 横山哲郎

1 はじめに

近年，可逆プログラミング言語 Janus を軽量暗号に用いることが検討されている．しかし，Janus ではタイミング攻撃を防ぐことが困難であったため，新たな言語として Hermes が提案された．Hermes は軽量暗号アルゴリズムに焦点を当ててつくられた可逆プログラミング言語である．しかし，Hermes を用いて記述を行った軽量暗号アルゴリズムのプログラムの数が少ないという現状がある．Hermes が実際に軽量暗号アルゴリズムに利用されるようになるには，Hermes が軽量暗号アルゴリズムに対して実際に有用であるかを示す必要があると考えられる．それには，より多くの軽量暗号アルゴリズムを，Hermes で記述を行えるかを検証する必要があると考えられる．したがって，Hermes が軽量暗号アルゴリズムに対して実用的であるかを示すことを最終的な目標とし，Hermes の有効性の評価を行う．その一環として，文献 [1] に記載されている軽量暗号以外の軽量暗号を対象として，その軽量暗号の可逆プログラムを Hermes を用いて記述を行うことを目的とする．目的の達成にあたって，まず軽量暗号のプログラムを Hermes で記述が行えるように，Hermes の構文と意味論について理解を深める．また，文献 [1] に記載されている軽量暗号 TEA，RC5，Speck128 の Hermes プログラムから暗号化，復号の C プログラムが生成されることを確認し，それらについても理解を深める．他にも，これら以外の軽量暗号である AES についても調査を行い，AES が Hermes で記述可能であるかを検証する．

2 関連研究

2.1 暗号

暗号とは情報を伝達する際に第三者から情報を守るために使われている技術である．また，暗号化された文自体を暗号と呼ぶ場合もある．今回は，意味の混同を避けるため，暗号化された文を暗号文と呼ぶこととする．基本的には，平文を暗号化した文を他者へ伝達した後，受け取った相手がその暗号文を復号し平文へと戻すことで情報のやりとりを行う．平文とは第三者が読んでも理解できるような文のことである．暗号化とは平文を暗号文に変換することである．復号とは暗号文を平文に戻すことである．

代表的な暗号方式として，共通鍵暗号と公開鍵暗号が存在する．共通鍵暗号では事前に 1 つの秘密の暗号鍵を共有しておき，暗号化，復号のどちらの場合も同じ鍵（共通鍵）を利用する．公開鍵暗号では公開鍵と秘密鍵の 2 つの鍵を使用して暗号化と復号を行うので，それぞれ異なる鍵を利用する．

2.2 暗号に対する攻撃

暗号に対する攻撃とは暗号文を何かしらの方法を使用することによって不正に情報を入手しようとすることである．攻撃の例としてサイドチャネル攻撃などが挙げられる．サイドチャネル攻撃とは暗号化・復号の処理時間，消費電力量，パソコンのノイズなど，物理的な要因から不正に情報を手に入れようとする攻撃である．暗号化・復号する処理時間を計測しその時間から解析を行うことをタイミング攻撃，消費電力量を計測することで解析を試みることを電力解析攻撃，微弱な電磁波を測定することで解析しようとすることをテンペスト攻撃という．

2.3 軽量暗号

軽量暗号とは限られた条件の中でも実装可能な暗号である．どのようなものが軽量暗号なのかは厳密に定義されていないが，軽量暗号は ISO/IEC によって標準化が進んでいる．現在，軽量暗号として認められているものは主に共通鍵暗号方式である．軽量暗号の例として，AES,Camelia,SPECK などが挙げられる．軽量暗号で求められる条件として，以下のようなものが挙げられる．

1. 消費電力量についての制約
2. メモリサイズについての制約
3. 処理時間についての制約
4. 回路規模についての制約

2.4 可逆計算

可逆計算とは，任意の計算過程の直前の状態が，高々一つに定まるような計算である．可逆プログラミング言語とは，可逆なプログラムのみを記述することができる言語である．プログラムの任意の実行過程において，直前の状態が高々一つに定まるとき，そのプログラムは可逆であるという．可逆でないプログラムを記述できないように制約がかけられていることにより，その分安全性が高まっており，可逆であることが保証されている．また，モジュール性も高まっており，あるプログラムを別のプログラムで再利用しやすくなっている．

2.5 AES

AES とは，米国の標準化機関である NIST(National Intitute of Standards and Technology) によって選定さ

```
program   = {procedure} ;                                                    (program)
procedure = "id" , "(" , args , ")" , stat ;                                 (procedure)
args      = type , ( "id" | "id" , "[" , "]" ) , | args , "," , args ;      (args)
type      = ( "secret" | "public" ) , inttype ;                              (type)
stat      = [ lval , update , exp | [ "if" , "(" , exp , ")" ] , lval , "<->" , lval |
            "for" , "(" , "id" , "=" , exp , ";" , exp , ";" , stat |
            ( "call" | "uncall" ) , "id" , "(" , lval , { "," , lval } , ")" |
            { decls , stat } ] , ";" ;                                       (stat)
exp       = lval | "numConst" | "size" , "id" | ( exp , binop | unop ) , exp ;  (exp)
lval      = "id" ,["[" , exp , "]"] ;                                        (lval)
decls     = type , lval , ";" , decls | "const" , "id" , "=" , "numConst" , ";" , decls ;  (decls)
inttype   = "u8" | "u16" | "u32" | "u64" ;                                   (inttype)
unop      = "~" ;                                                            (unop)
binop     = "+" | "-" | "*" | "/" | "%" | "&" | "|" | "^" | "==" |
            "!=" | "<" | ">" | "<=" | ">=" | "<<" | ">>" ;                   (binop)
update    = "+=" | "-=" | "^=" | "<<=" | ">>=" ;                            (update)
```

図 1　Hermes の構文

れた共通鍵暗号方式のアルゴリズムである．AES は主に四種類の処理を行う．一つ目の処理は SubBytes といい，16 バイトの入力に対して 1 バイトごとに処理を行い，それぞれの値を 0〜255 のいずれかの値に置き換える．二つ目の処理は ShiftRows といい，4 バイト単位の行を，行ごとにそれぞれ異なるバイト数分だけ左へシフトするという処理を行う．三つ目の処理は MixColumns といい，ビット演算を用いて，ある 4 バイトを別の 4 バイトへと変換する処理を行う．四つ目の処理は AddRoundKey といい，MixColumns で得た出力とラウンド鍵の排他的論理和をとる処理を行う．これらを 1 ラウンドとして，10〜14 回繰り返す．

3　Hermes

可逆プログラミング言語 Hermes は Janus から着想を得て作られた，軽量暗号アルゴリズムに対するドメイン固有言語である．[1] ドメイン固有言語とはある特定の分野に利用する目的で作られた言語であり，それ以外の目的で利用されることを想定していない言語である．Janus とは可逆であるプログラムしか記述することができない命令型プログラミング言語である．軽量暗号に Janus を用いる利点として，暗号化，復号を 1 つのプログラムで実行できる点などが挙げられる．Janus は暗号化，復号の処理時間が暗号鍵と平文の値によって変化する制御構造を持っている，したがって，サイドチャネル攻撃の一種であるタイミング攻撃から守られていない．一方，Hermes は暗号化，復号の処理時間が暗号鍵と平文の値と独立している制御構造を持っていることによって，タイミング攻撃から守られている．[2]

- Hermes の構文
  Hermes の構文を EBNF で記述したものは図 1 のようになる．
- Hermes の意味論（判断，意味規則）
  Hermes の意味論の一部を表示的意味論で記述したものは図 2 のようになる．

$$\mathcal{L}[\![x]\!](\sigma,\eta) = \eta(x) \qquad \text{(Variable/Constant)}$$

$$\mathcal{L}[\![x[e]]\!](\sigma,\eta) = (z,\ ve[i]) \qquad \text{(ArrayElement)}$$
$$\text{where} \quad \eta(x) = (z,\lambda),\ \sigma(\lambda) = (vs, ve),\ \mathcal{E}[\![e]\!](\sigma,\eta) = i,\ i < vs$$

$$\mathcal{E}[\![n]\!](\sigma,\eta) = n \qquad \text{(Constant1)}$$

$$\mathcal{E}[\![x]\!](\sigma,\eta) = n \qquad \text{(Constant2)}$$
$$\text{where } \eta(x) = (n, \texttt{null})$$

$$\mathcal{E}[\![l]\!](\sigma,\eta) = \sigma(\lambda)\uparrow_z \qquad \text{(L-val)}$$
$$\text{where } \mathcal{L}[\![l]\!](\sigma,\eta) = (z,\lambda)$$

$$\mathcal{E}[\![\neg e]\!](\sigma,\eta) = I(\neg)(v) \qquad \text{(UnOp)}$$
$$\text{where } \mathcal{E}[\![e]\!](\sigma,\eta) = v$$

$$\mathcal{E}[\![e_1 \odot e_2]\!](\sigma,\eta) = I(\odot)(v_1, v_2) \qquad \text{(BinOp)}$$
$$\text{where } \mathcal{E}[\![e_1]\!](\sigma,\eta) = v_1,\ \mathcal{E}[\![e_2]\!](\sigma,\eta) = v_2$$

$$\mathcal{E}[\![\texttt{size}\ x]\!](\sigma,\eta) = vs \qquad \text{(Size)}$$
$$\text{where } \eta(x) = (z,\lambda),\ \sigma(\lambda) = (vs, ve)$$

図 2　Hermes の意味論の一部

4　結果と今後の課題

Hermes の構文を EBNF で記述を行うことにより，Hermes の構文についての理解を深めた．また，Hermes の意味論を表示的意味論で記述を行うことにより，Hermes の意味論についての理解を深めた．文献 [1] に記載されている軽量暗号 TEA，RC5，Speck128 の Hermes プログラムから暗号化，復号の C プログラムが生成されることを確認し，それらについても理解を深めた．また，AES についても調査を行い，AES のプログラムの一部である MixColumns などを Hermes を用いて記述を行った．しかし，AES のその他のプログラムについては，Hermes で記述が行えるかの検証を行えていないので，実際に検証を行う必要があると考えられる．今後は，AES のすべてのプログラムを Hermes で記述するために，AES と Hermes について更なる理解が必要だと考えられる．また，TEA，RC5，Speck128，AES 以外の軽量暗号についても調査し，それらが Hermes で記述可能であるかを検証する必要があると考えられる．

参考文献

[1] Mogensen, T.Æ.: Hermes: A Reversible Language for Writing Encryption Algorithms (Work in Progress), *Perspectives of System Informatics* (Bjørner, N., Virbitskaite, I. and Voronkov, A., Eds.), Cham, Springer International Publishing, pp.243–251 (2019).

[2] Mogensen, T.Æ.: Hermes: A Language for Light-Weight Encryption, *Reversible Computation* (Lanese, I. and Rawski, M., Eds.), Cham, Springer International Publishing, pp.93–110 (2020).