軽量暗号アルゴリズム記述のための領域特化言語 Hermes の拡張

2018SE088　高見雄大
指導教員：横山哲郎

1　はじめに

近年，急速な IoT 化が進むにつれ多種多様なデバイスがインターネットに接続するようになった．一方で，インターネットに接続するデバイスへのサイバー攻撃も増加している．このサイバー攻撃から情報を守るために暗号技術が開発されている．また，暗号技術に対して可逆プログラミング言語 [1] が検討されている．

本研究は，軽量暗号アルゴリズムに対する領域特化可逆プログラミング言語 Hermes を拡張することである．

可逆プログラミング言語は uncall することで，全ての変数や配列の情報は消去する前に 0 にクリアされる．よって，暗号アルゴリズムの実装で記憶領域に情報を残すことなく実行することが出来る．したがって，記憶領域に対する攻撃から守られることが保証されている．しかし，サイドチャネル攻撃はメモリリークに関する対策だけでは完全な対策にはならない．従来の可逆言語 Janus では入力値に暗号処理時間が依存している．よって，サイドチャネル攻撃の一種であるタイミングベース攻撃について脆弱性がある．Hermes は Janus の良いところを残しつつ，入力値と実行時間を独立させることでタイミング攻撃から守られている．非可逆言語は暗号化プロシージャと復号プロシージャの 2 種類作成する必要性があるが，可逆なプログラミング言語は暗号化と復号を一つのプロシージャで実行することが出来る．よって，非可逆な言語の様に暗号化と復号それぞれプログラムを作る必要がない．Hermes で暗号アルゴリズムをコンパイルすることにより，全ての実行が高々一意に定まる可逆性が主張された C のプログラムが出力されるように設計されている．しかし，Herme は開発途中の言語であり，[2] にいくつかの暗号アルゴリズムに対して記述できることは示されている．また，Hermes では機密度型が定義されており，この機密度型が原因で脆弱性に関係のないコードでコードクローンが発生する可能性がある．実際に，軽量暗号アルゴリズム Speck128 についてコードクローンが生成された．本研究では，可逆プログラミング言語 Hermes を記述する際，脆弱性に関係のないコードについてコードクローンが生成されてしまう問題を解決することを目的とする．

2　準備

本研究を取り組むための背景を準備として記述する．

2.1　暗号の脆弱性

暗号は情報を第三者から守る手段である．暗号化した情報が第三者に何かしらの攻撃されることで情報が流出する可能性があることを脆弱性という．

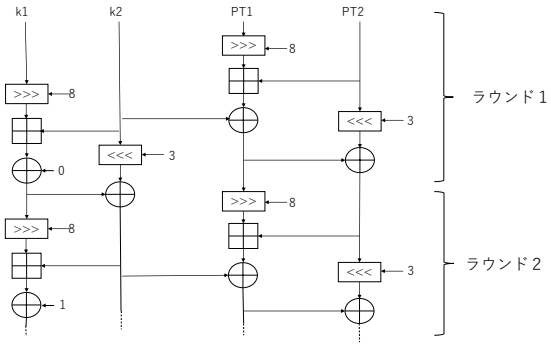


図 1　Speck128 の流れ

2.1.1　サイドチャネル攻撃

サイドチャネル攻撃は攻撃手法の一種である．サイドチャネル攻撃の特徴は安全性が計算量によって保証された暗号アルゴリズムとは関係のない物理的な情報を利用することにより，暗号鍵を特定する解読手法である．例えば，暗号化復号の処理時間を計測することや，消費される電気量を測定，コンピュータの暗号化が実装の際に発するノイズを測定するなどが上げられる．

2.1.2　タイミングベース攻撃

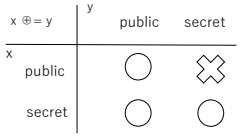
タイミングベース攻撃とはサイドチャネル攻撃の一種である．処理時間を測定することで暗号鍵を不正に特定する手法で，タイミングベース攻撃から守るためには暗号化復号する際，暗号鍵に関係なく処理時間を独立させることでタイミングベース攻撃から守ることができる．janus は暗号化復号の際，暗号鍵に処理時間が依存してしまう．その為，タイミングベース攻撃から守ることが出来なかった．だから，新しい可逆プログラミング言語 Hermes が作られた．

2.2　暗号アルゴリズム Speck128

暗号アルゴリズムで実装する際，基本的にラウンドと呼ばれる一連の流れをブロックのビット数に応じて繰り返すことで暗号化を行っている．Speck128 はブロック暗号方式を採用した軽量暗号アルゴリズムである．Speck 暗号の暗号化の一連の流れを 1 に示す．

2.3　可逆プログラミング言語 Hermes

可逆言語 Hermes は可逆言語 Janus を参考に作成された．



2.3.1 変数や定数についての型システム

Hermes の値は基本的に 64bit の符号なし整数と定められる．また，全ての式は機密度定数型が定められており public 型か secret 型の 2 種類のどちらかで指定される．スカラー変数や配列変数は 8,16,32,64bit の数値サイズで示される．定数の型は Const で示され，64bit と定められる．値の機密度定数の型を t と表記する．機密度定数は public⊑ secret を満たす機密度束 ({public，secret}，⊑) を仮定する．

2.3.2 可逆言語における機密度型

secret 型と public 型について説明する．x と y が public 型の場合，実行後の x の機密度型は public 型で問題ない．x と y が secret 型の場合も実行後の x の機密度型は secret 型で問題ない．実行後の x の値と実行前の x の値から secret 型の y の値を推測することは可能である．secret 型の情報が漏れてしまうと暗号文の脆弱性に繋がってしまう．なので，x の機密度型が public 型で y の機密度型が secret 型の場合は成り立たないことがわかる．x の機密度型が secret 型で y の機密度型が public 型の場合，y の値が public 型でも x の値を推測することは不可能である．なので，y の機密度型が public 型の場合は成り立つ．$x\oplus = y$ の計算において x の機密度型が public 型で y の機密度型が secret 型の時のみ成り立たないことがわかる．

3 本研究の目的

本研究の目的は Hermes で発生する可能性がある，脆弱性に関係のないコードについてコードクローンが生成される問題を解決することである．軽量アルゴリズム Speck128 にコードクローンが発生した．以下のソースコードは [2] を参照する．

Listing 1 speck128 のコードクローン

```
1 Rs(u64 x, u64 y, secret u64 k)
2 { x >>= 8; x += y; x ^= k; y <<= 3; y^= x
      ; }
3 Rp(u64 x, u64 y, public u64 k)
4 { x >>= 8; x += y; x ^= k; y <<= 3; y^= x
      ; }
```

4 Hermes の拡張方法

コードクローンの問題を解決するための拡張方法は 2 通りあると考えられる．1 つ目が機密度型のパラメータ化である．しかし，パラメータ化すると拡張が複雑になることが想定される．本研究では 2 つ目の新たな機密度型 N 型を拡張する．N 型は出力が secret 型で指定される時，入力でのみ N 型を指定することが出来る．Hermes を使用した Speck128 の記述の問題点は脆弱性に関係のないコードにコードクローンが生成されてしまう点である．N 型を提案することでクローンプログラムをなくすことが出来る．暗号アルゴリズムに使用された鍵情報を漏らさないために，secret 型が定義されている．$x\oplus = y$ の計算で考えると x が public 型で y が secret 型は成り立たない．言い換えると，計算後の型が public 型の時，入力が secret 型となることはできない．speck128 のクローンプログラムにおいて k だけが public 型を持っている．機密度型である public 型と secret 型には入力が public 型であっても出力が secret 型であるなら情報の漏洩はない特徴がある．つまり，N型が式の右辺でのみ使用できるように拡張すれば既存の機密度型に影響を及ばすことなくクローンプログラムを無くすことが出来るのではないかと考える．

Listing 2 N 型を使用した speck128

```
1 R(u64 x, u64 y, N u64 k)
2 { x >>= 8; x += y; x ^= k; y <<= 3; y^= x
      ; }
```

5 まとめ・今後の課題

本研究の成果として，本研究で題材となる可逆プログラミング言語 Hermes で脆弱性に関係のないコードにコードクローンが生成される問題は解決できた．実際に Speck128 のコードクローンの問題を Hermes を拡張することで解決できた．

今後の課題は，意味論を定義することや可逆性の保証をすること，提案した N 型の実装をすることである．また，他の軽量暗号アルゴリズムを記述する際に記述することが出来なかったり記述することは出来るが何かしらの問題が起こることが想定される．なので，それぞれのアルゴリズムにあった拡張を考え実装することが課題である．

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).