修士論文

センサネットワークに向けた小型暗号回路

A Tiny Cipher Circuit for Sensor Networks

平成 17 年 1 月 31 日 提出

指導教官
藤島　実　助教授

東京大学大学院 新領域創成科学研究科
基盤情報学専攻

学籍番号：47-36307

金子　秀彦

要旨

近年、相互通信能力や演算処理能力を持った無数の超小型集積回路がセンサネットワークを構成する、ユビキタス社会の到来が期待されている。センサネットワークにおける重要課題の 1 つとして、ノード間通信の安全性がある。センサネットワークで用いられる集積回路で求められるのは、超小型である事と、バッテリ交換が困難である事から低消費電力であることの 2 点が挙げられる。しかし、RSA に代表される非対称暗号や、AES に代表される共通鍵暗号などの、現在用いられている暗号方式は、消費電力や回路規模の制限からセンサネットワークに適用することは出来ない。そこで、超小型暗号回路を実現すべく、3 つの方式を提案し、その安全性について検証した。その結果、暗号化を共通鍵の共有とメッセージ送信の 2 つの操作に分離する手法は、従来手法と同程度の安全性を保ちながら、超小型で低消費電力を実現できる暗号方式であることを示した。

第 1 章　序論

近年、相互通信能力や演算処理能力を持った無数の超小型集積回路がセンサネットワークを構成する、ユビキタス社会の到来が期待されている。MEMS 技術や RF 集積回路技術の発展により、センサネットワークで用いられる集積回路が安価に作成できることも、ユビキタス社会の到来の後押しをしている[1]。センサネットワークは、天候や動物の生態などのモニタリング、警備のための監視など様々な分野での利用が期待されている。

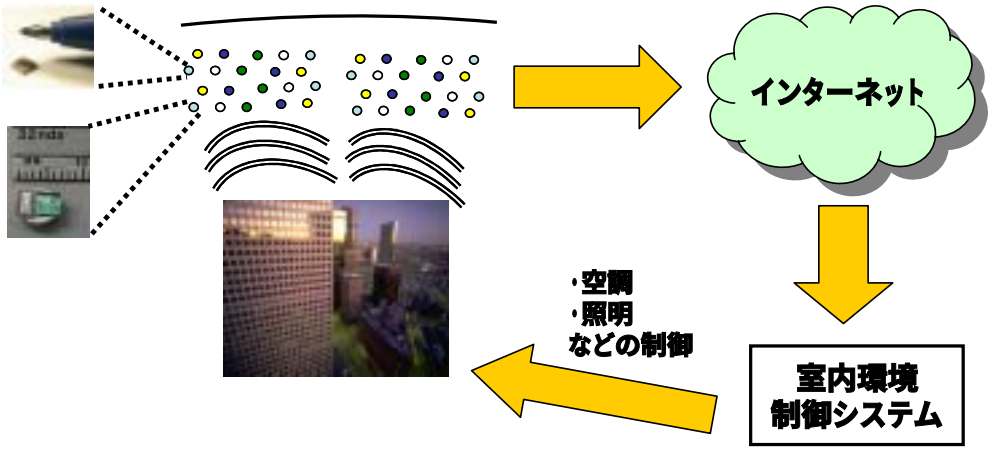

図 1.1. センサネットワークの応用例

しかし多くの場合、センサネットワークはアクティブに周囲の環境をモニタリングしているため、盗聴者が望む情報を入力として与えることが容易に行える[2]。また、ノード間通信が無線で行われる事により、盗聴者は容易に盗聴ができ、盗聴を容易に行うために盗聴者が望むパケットをセンサノードの送信することも可能になる。以上の 2 点から、ノード間通信の安全性は、センサネットワークにおいて、最も重要な課題の 1 つと言う事ができる。

そこで、次に暗号技術について述べる。現在、暗号方式は大きく分けて秘密鍵暗号と公開鍵暗号の 2 つの種類に分類される。秘密鍵暗号では暗号アルゴリズムは公開するが、暗号化の鍵と復号化の鍵が同一でありその値は秘密である。公開鍵暗号では暗号化の鍵と復号化の鍵が異なっており、暗号アルゴリズムに加え、暗号化の鍵も公開される。片方を公開しても他方が求まらないように設計されているので、このような扱いが可能となる。

本研究では、小型暗号回路を実現できるアルゴリズムを提案しているが、秘密鍵暗号方式を用いている。そこで、秘密鍵暗号方式についての技術動向について述べる。秘密鍵暗号方式は大きく分けてストリーム暗号とブロック暗号の 2 種類に分類される。

ストリーム暗号では、鍵として乱数系列を用いる。完全な乱数を生成することは難しく、

また長い乱数列を送信者と受信者の間で共有することは効率面でも問題が大きい。以上のような問題から、ブロック暗号と比較して研究はあまりされていない。

ブロック暗号に関しては、1977 年に DES 暗号[3]が米国連邦政府標準暗号に制定された。それ以来、暗号に関連した研究は急速に発展し、多くの研究分野が新たに開拓された。DES 暗号の安全性についても数多くの研究が行われ、現在でも DES 暗号に対する最も現実的な解読法は鍵の全数探索法となっている。しかし、コンピュータの普及に伴って世の中のコンピュータの台数が増加し、それらがネットワークにつながり特定の目的のために利用可能となり、かつコンピュータ 1 台当たりの処理能力が向上し、さらに暗号アルゴリズム実装技術の研究が発展して 1 秒当たりに調べることができる鍵の個数が増加した現状では、DES 暗号の安全性は徐々に低下して来ていると言わざるを得ない。

以上のような背景のもとで、2001 年に公募の上、NIST(National Institute of Standards and Technology)[4]により DES 暗号に代わる秘密鍵標準ブロック暗号として Rijndael が採択された[5]。このプロジェクトは AES(Advanced Encryption Standard)と呼ばれており、Rijndael は AES 暗号と呼ばれる場合が多い。AES 暗号の制定以来、多くの研究者が AES 暗号の集積回路への実装に取り組み、現在では 0.6μm プロセスで 10,799[GE]での実装が報告されている[6]。

しかしながら、それでも消費電力や回路面積の制限から上述したセンサネットワークや RF タグなどの超小型集積回路おけるセキュリティ面の問題の解決策として AES 暗号を用いる事はできない。そこで、本論文ではブロック暗号の概念を取り入れたストリーム暗号を用いることで、暗号回路の小型化が実現できることを示した。

本論文の構成を示す。第 2 章では秘密鍵暗号について述べる。第 3 章では双方向通信を用いた暗号化手法を述べ、その効果と欠点について述べる。第 4 章では、第 3 章における欠点を踏まえた暗号手法について述べ、その安全性を証明した。また、集積回路に実装した場合に必要となるゲート数を見積もり、小型化が実現できることを示した。最後に第 5 章で結論を述べる。

第 2 章　従来の秘密鍵暗号

本章では、ブロック暗号と本研究における暗号方式であるストリーム暗号とその動作モードについて述べる。

2.1. ブロック暗号

本節では、ブロック暗号の概念と、現在標準暗号として用いられている AES について述べる。また、AES の集積回路への実装についての論文は数多く発表されているが、その回路規模について述べ、センサネットワークなどで用いられる超小型集積回路への搭載について検討する。

2.1.1. ブロック暗号概要

ブロック長と呼ばれる長さ n ビットの平文 M をまとめて暗号化する共通鍵暗号をブロック暗号と呼ぶ。図 2.1 に示すように、ブロック暗号の暗号化アルゴリズム E_K は、鍵長 k ビットの秘密鍵 K と、n ビットの平文 M を入力とし、n ビットの暗号文 $c = E_K(m)$を出力する。平文 M は次式によって計算する。ただし、D_K を秘密鍵が K であるときの復号アルゴリズムであるとする。

$$D_K(c) = D_K(E_K(m)) = m \tag{2.1}$$

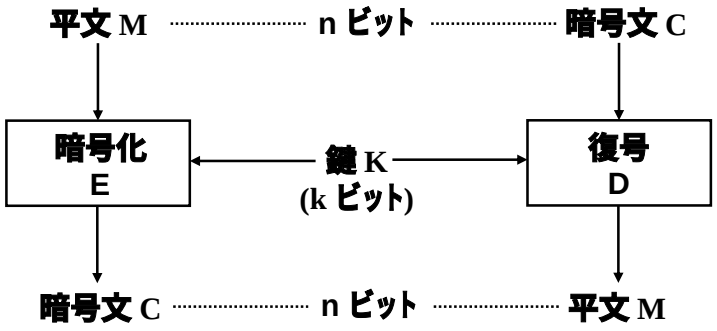

図 2.1. ブロック暗号

2.2.2. 標準暗号

暗号技術が最も進んでいる米国において、2001 年に公募の上、AES 暗号が標準ブロック暗号として制定された。現在では、AES は最も用いられている暗号方式となっている。以下では、AES の暗号方式を述べる。

概要

[1] 鍵長とラウンド数

AES 暗号の鍵長は 3 通り(128, 196, 256 ビット)から選ぶことができる。また、4 バイトを 1 ワードとし、鍵 K のワード長を N_k で表す。従って、N_k = 4, 6, 8 となる。また、暗号化におけるラウンド数を N_r で表す。それぞれの鍵長 N_k に対し、ラウンド数 N_r は、表 2.1 のように規定されている。

鍵のビット長[ビット]	128	196	256
鍵のワード長 N_k[ワード]	4	6	8
ラウンド数	10	12	14

表 2.1. AES における鍵のビット長とラウンド数の関係

[2] ブロック長

AES 暗号のブロック長は 128 ビットとなっている。128 ビットは、各要素が 1 バイトである 4 行 4 列の配列で表現できるので、暗号化、及び復号の途中段階の状態は、図 2.2 で示されるような 4 行 4 列の状態配列で表される。

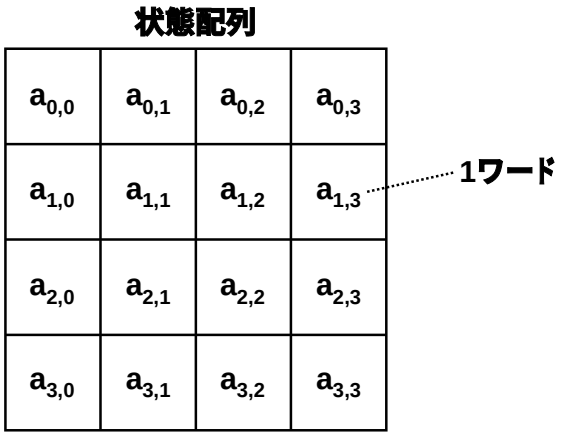

図 2.2. 1 ブロックを表す状態配列

[3] 暗号化の概要

暗号化アルゴリズムは、暗号処理部と鍵拡張部からなる。

(1) 暗号化処理部

暗号化処理部は、図 2.3 の様に、初期処理、第 1 ラウンド処理、第 2 ラウンド処理、… 、第 N_r ラウンド処理からなる。

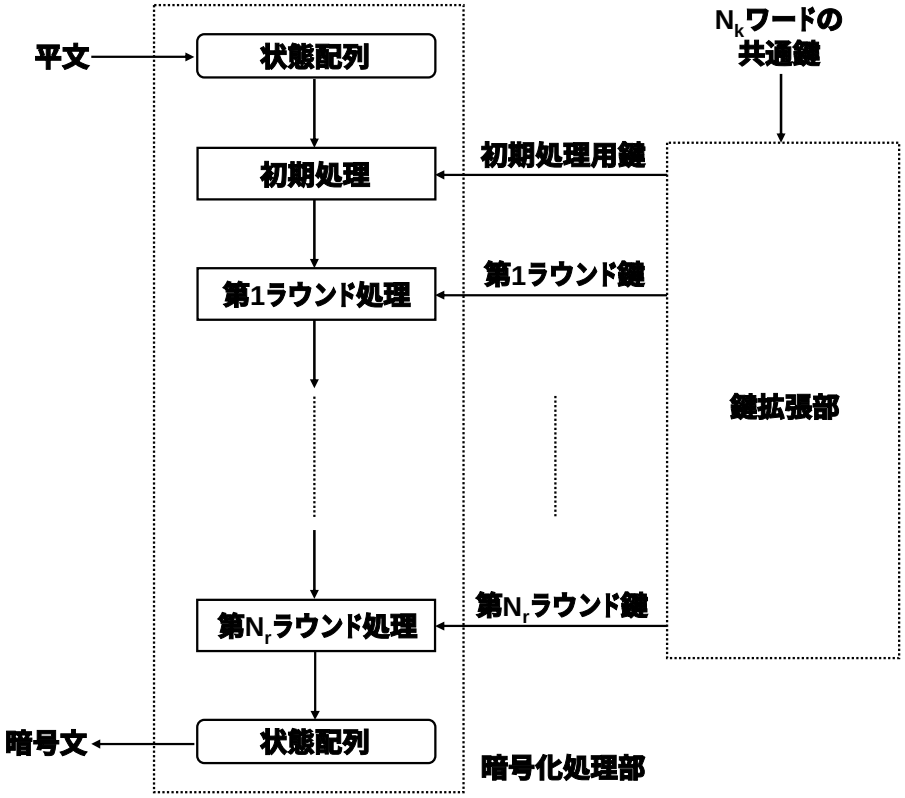

図 2.3. 暗号化の流れ

初期処理と各ラウンド処理では、ラウンド鍵と呼ばれる 4 ワードの鍵が用いられる。従って、全部で $N_r + 1$ 個の 4 ワードのラウンド鍵が使用される。

各ラウンド鍵は 4 ワード = 16 バイトなので、4 行 4 列の配列で表現される。

(2) 鍵拡張部

鍵拡張部では、N_k ワードの鍵 K を $4(N_r + 1)$ワード $w_0,\cdots,w_{4Nr+3}$ に引き伸ばす。図 2.4 に示すように、(w_0, w_1, w_2, w_3)を初期処理用として、$(w_{4r}, w_{4r+1}, w_{4r+2}, w_{4r+3})$を第 r ラウンドのラウンド鍵として用いる。

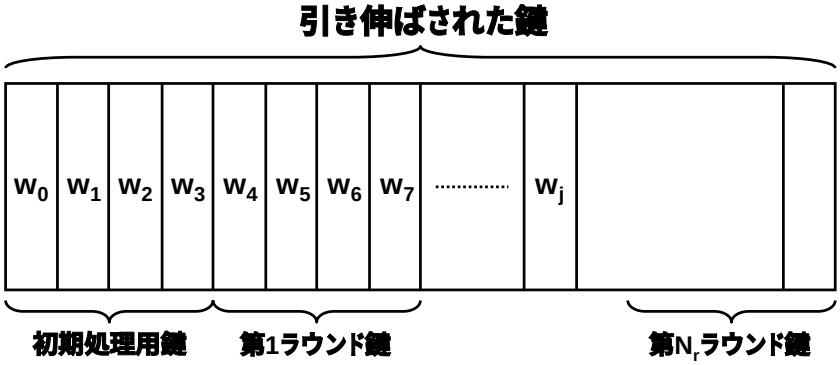

図 2.4. 引き伸ばされた鍵とラウンド鍵

アルゴリズム

[1] 初期処理

初期処理では、図 2.5 のように、平文を表す状態配列と、初期処理用の鍵配列の排他的論理和が取られる。

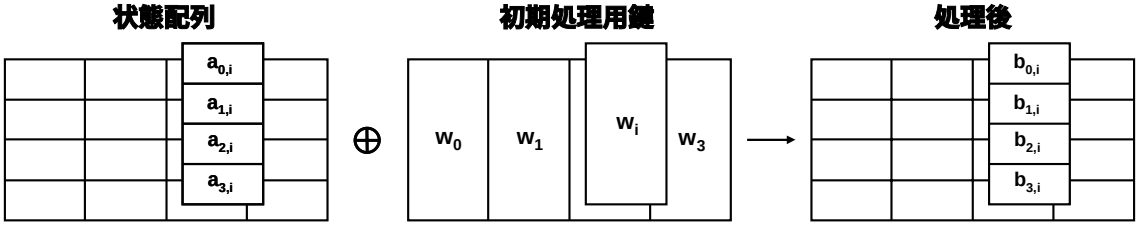

図 2.5. 初期処理

[2] 各ラウンド処理

各ラウンドでは、状態配列に対して、(1)バイト置換、(2)行シフト、(3)列内ミックス、(4)ラウンド鍵加算の 4 つの操作が順に行われ、(4)のあとの状態配列が、次のラウンドの処理へ渡される。第 N_r ラウンド目の処理は、(1),(2),(4)の処理のみ行われ、(3)の列内ミックスは行われない。

(1) バイト置換

図 2.6 に示すように、状態配列の各バイトを $a_{i,j}$ とすると、$a_{i,j}$ に S ボックスと呼ばれる置換を適用したものを変換後の状態配列の各バイト $b_{i,j}$ とする。

S ボックス置換は次のように定義される。入力された 1 バイト a を、拡大体 GF(2^8)の元とみなし、逆元を求める。

$$x = a^{-1}$$

ただし、GF(2^8)の法多項式は $P(X) = X^8 + X^4 + X^3 + X + 1$ とする。

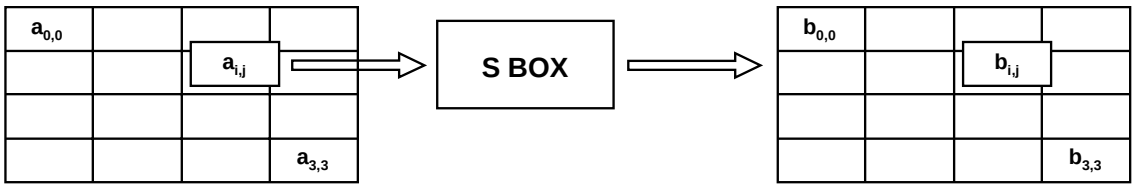

図 2.6. バイト置換

次に、x を GF(2)上の八次元ベクトル($x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7$)と考えて次式により GF(2)上の八次元ベクトル $y = (y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7)$を求める。

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

得られた 8 ビット y を、S ボックス置換の出力とする。

(2) 行シフト

状態配列が図 2.7(a)の状態であるとすると、変換後の状態配列は図 2.7(b)の様になる。すなわち、2 行目は左へ 1、3 行目は左へ 2、4 行目は左へ 3 シフトされる。

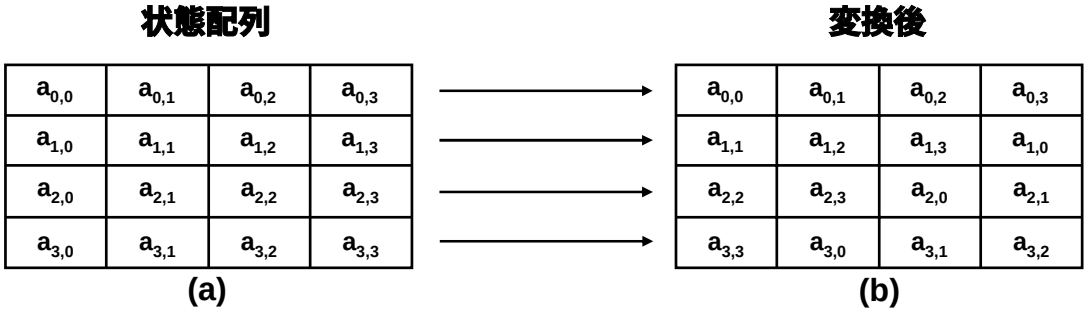

図 2.7. 行シフト

(3) 列内ミックス

この操作は、状態配列の 4 つの列に対して、同じ操作がそれぞれ行われる。第 j 列の 4 バイト($a_{0,j}$, $a_{1,j}$, $a_{2,j}$, $a_{3,j}$)の各要素を、それぞれ GF(2^8)の要素とみなし、それらを係数とする GF(2^8)上の多項式を A(X)とする。次に

$$B(X) = A(X) \times (3X^3 + X^2 + X + 2) \bmod (X^4 + 1)$$

を計算する。ここで、多項式の係数の和や積の計算は GF(2^8)上で行う。B(X)は GF(2^8)上の 3 次多項式となるので、B(X)は４つの係数($b_{0,j}$, $b_{1,j}$, $b_{2,j}$, $b_{3,j}$)を持つ。そして、これらを変換後の状態配列の第 j 列の 4 バイトとする。以上の変換を、図 2.8 に示す。また、以上の変換は、以下の式と等価である。

$$\begin{pmatrix} \mathbf{b}_{0,j} \\ \mathbf{b}_{1,j} \\ \mathbf{b}_{2,j} \\ \mathbf{b}_{3,j} \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \begin{pmatrix} \mathbf{a}_{0,j} \\ \mathbf{a}_{1,j} \\ \mathbf{a}_{2,j} \\ \mathbf{a}_{3,j} \end{pmatrix}$$

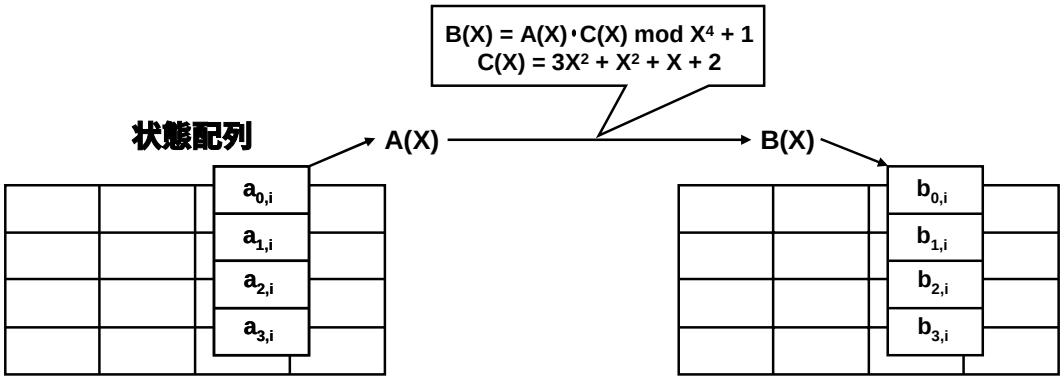

図 2.8. 列内ミックス

(4) ラウンド鍵加算

初期処理と同様に、状態配列とラウンド鍵を表す鍵配列の排他的論理和を取る。

[3] 復号

暗号文に対して、(4)ラウンド鍵加算→(3)列内ミックス→(2)行シフト→(1)バイト置換のように逆処理を行えば、もとの平文が得られる。

[4] 鍵拡張部

鍵拡張部は、N_k ワードの共通鍵 K から、$4(N_r + 1)$ワードの鍵 $w_0, \cdots, w_{4Nr+3}$ を作成する。ここでは、N_k = 4,6 の場合のみについて説明する。

まず、初めの N_k ワードである $w_0, \cdots, w_{Nk-1}$ は、鍵 K をそのまま使う。以降の鍵 w_i は、図 2.15 のように計算される。ただし、S(w)は、w の各バイトに S ボックス置換を適用した結果得られる 4 バイト(1 ワード)を表す。00 は、全てビット 0 からなる 1 ワードを表す。

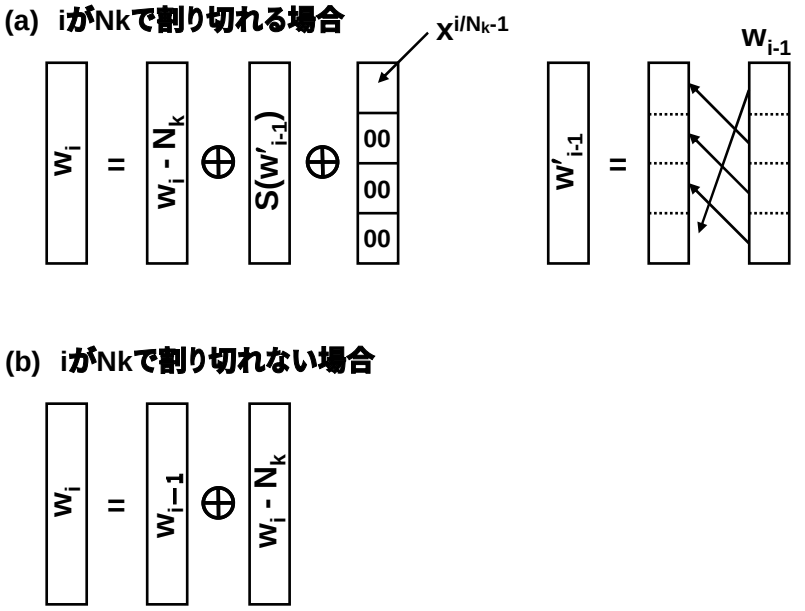

図 2.9. 鍵拡張

2.2. ストリーム暗号

本節では、本研究の暗号方式であるストリーム暗号とその動作モードについて述べる。ストリーム暗号とは、図 2.10 に示すように、平文系列に鍵系列をビット単位でマスクすることにより、暗号化を行う方式である。

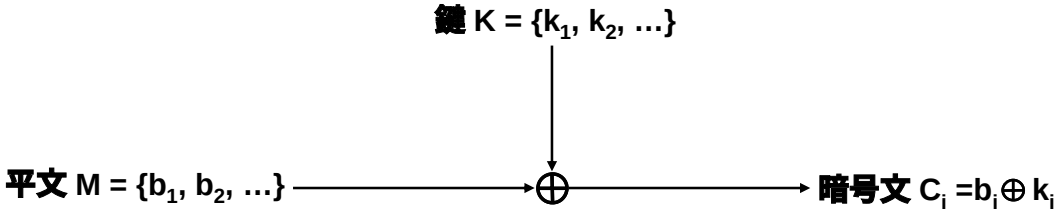

図 2.10 ストリーム暗号

鍵系列 K が真にランダムな場合は、無限大の能力を有する敵に対しても安全である。しかし、鍵系列 K を真にランダムにするためには、平文ごとに K を新しく選び直さなければならず、効率が悪い。そこで、線形シフトレジスタなどを利用し、短い鍵から長い擬似ランダムな鍵系列を生成する方法が提案されている。

2.2.1. 同期式ストリーム暗号

同期式ストリーム暗号の一般的なブロック図を図 2.11 に示す。送信側と受信側の擬似乱数生成器の同期は、何らかの外部的な機構により取れるものとする。通常、擬似乱数としては、線形フィードバックレジスタ(LFSR)を用いたものや、ブロック暗号を利用したものが用いられる。安全となるように設計されたブロック暗号では非線形演算が組み込まれているため、ブロック暗号を用いた擬似乱数生成器は非線形な乱数生成器となる。

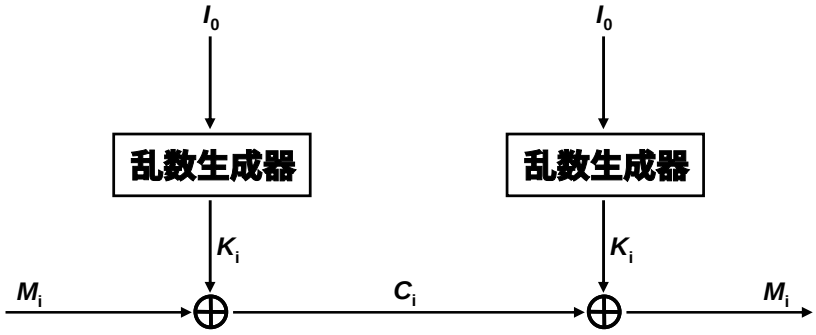

図 2.11. 同期式ストリーム暗号

ブロック暗号を利用した同期式ストリーム暗号を図 2.12 に示す。長さ n_B ビットのレジスタの内容 R_i をブロック暗号で暗号化し、その出力 S_i の一部から鍵 K_i を作る。非線形演算を含むブロック暗号の符号化は、LFSR による乱数生成に比べて、通常時間がかかる。そのため、一度のブロック暗号化で $n(1 \le n \le n_B)$ ビットの鍵を作ることにより、高速化が計られる。S_i から鍵 K_i を作る一方で、S_i から n_F ビットを取り出して F_i を作り、n_F 回左シフトしてその値をレジスタに戻す。なお、回路を簡単化するために、$F_i=K_i$ あるいは $F_i=S_i$ とすることもできる。

後者の場合は、n_B 本並列にフィードバックできるように回路を作れば、一度のシフト操作で F_i のフィードバック操作が完了し、高速化できる。このようなブロック暗号の使用法を出力フィードバック(OFB)モードと呼ぶ。

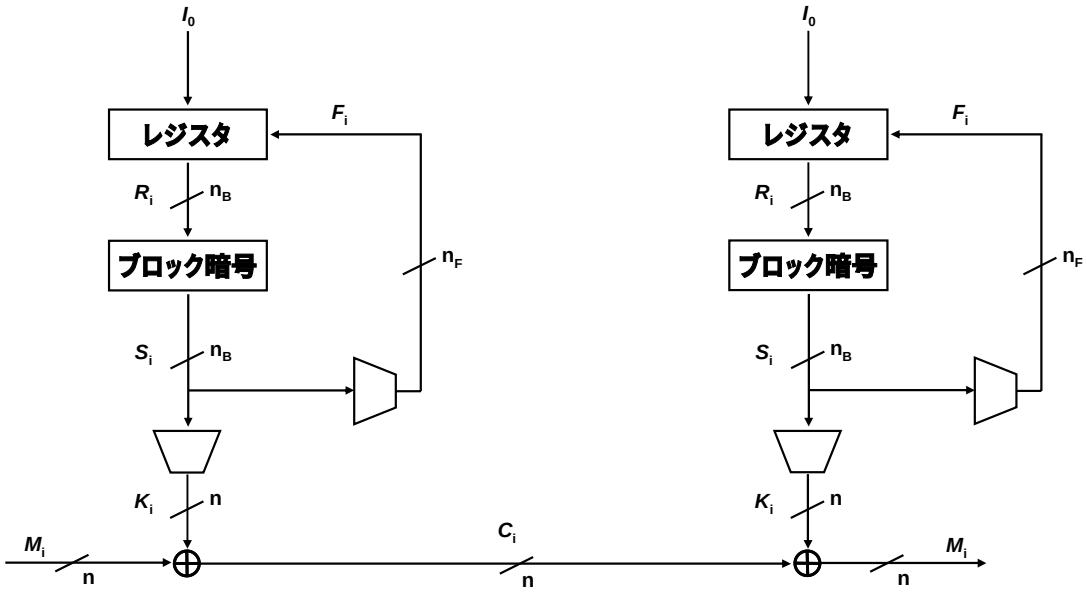

図 2.12. ブロック暗号の OFB モード

図 2.12 では、F_i をフィードバックすることにより、レジスタの内容を更新している。しかし、n_F=0 として暗号器出力 S_i からは何もフィードバックしないで、代わりにカウンタを用意し、そのカウンタ出力でレジスタの内容を更新していく手法もある。

OFB モードでは、時刻 i に置ける鍵の値 K_i を知るためには、始めから鍵生成操作を繰り返さなければ求める事ができない。上記で述べたカウンタを用いる手法では、カウンタの値を i に設定すれば、K_i が求められる。従って、任意の時刻 i の暗号文 C_i から対応するメッセージ M_i を容易に復号できる。また、ある平文 M_i の値を M_i'に変更したいとき、全平文系列を再暗号化しなくても、対応する C_i を $C_i' = M_i' \oplus K_i$ に書き直すだけで処理が終わる。

同期式ストリーム暗号では、通信路の雑音によるビットの欠落などで、一度同期が外れると、暗号文 C_i と鍵 K_i の対応がずれるため、その後の暗号文系列は正しく復号できなくなる。つまり、欠落誤りに対して誤り伝播が生じる。従って、同期式ストリーム暗号ではこのような同期ずれを防ぐために、暗号文系列をある一定長ごとでフレーム化し、フレーム単位で同期をチェックして、同期ずれを回復する。

2.2.2. 自己同期式ストリーム暗号

同期式ストリーム暗号では、送信側と受信側で擬似乱数の同期を取るための機構を、ストリーム暗号外に組み込む必要がある。自己同期式ストリーム暗号方式は、図 2.3 のように構成され、ストリーム暗号自身で同期を自動的に回復する。図 2.2 の同期式ストリーム暗号では、シフトレジスタへのフィードバック F_i を S_i から作っていたものを、暗号文 C_i を用いて、$F_i = C_i$ とした場合に対応している。

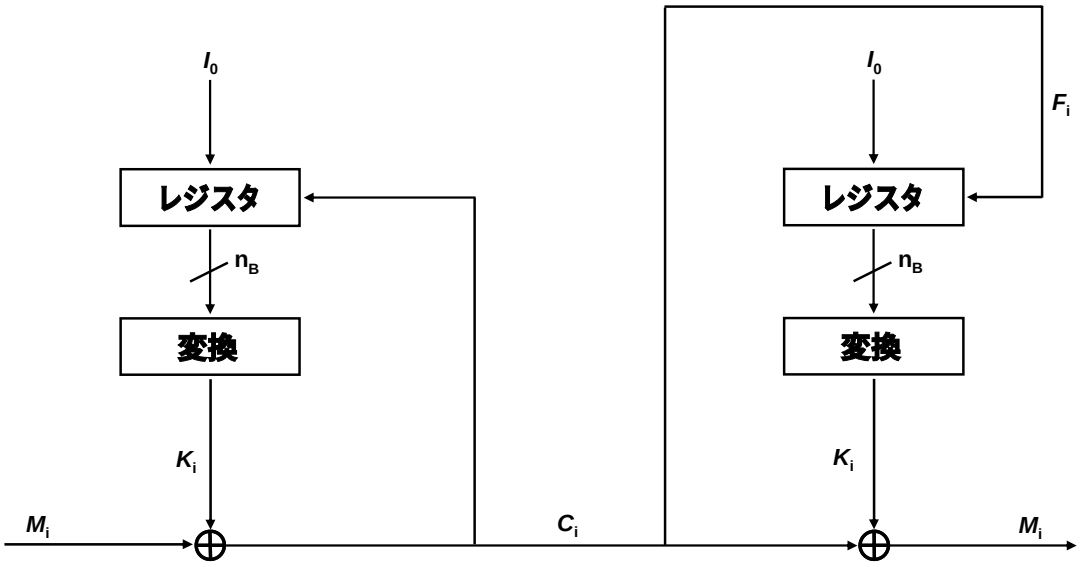

図 2.13. 自己同期式ストリーム暗号

図 2.13 のシステムでは、送信者と受信者で同期がずれても、レジスタ長 n_B ビット分の C_i が誤りなく伝送されれば、その後は両側のレジスタが同じになり、擬似乱数の同期が回復する。ただし、自己同期ストリーム暗号では、C_i のビット誤りやビット欠損が生じるとその誤った C_i が受信側のレジスタにある間は、送信側と受信側で異なる乱数が生成されるため、レジスタ長分の誤り伝播が生じる。

図 2.13 の変換部分には、強度の点から非線形演算を含むブロック暗号を使用することができる。図 2.14 に示すようなブロック暗号の使用法を暗号文フィードバック(CFB)モードという。なお、n=1 の時には、図 2.13 の自己同期式ストリーム暗号と等価になる。

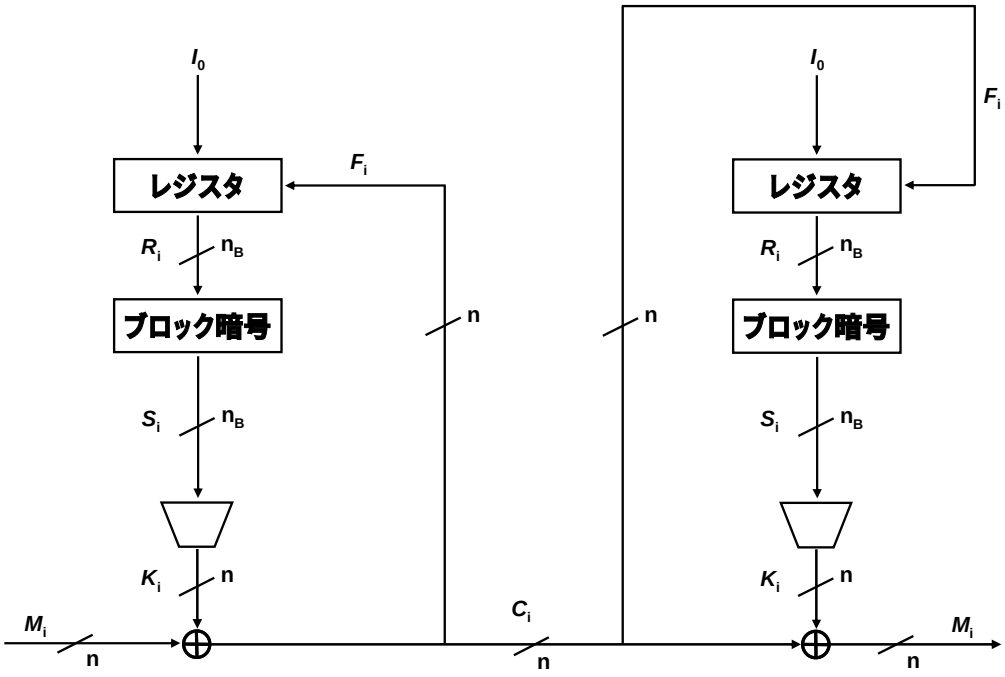

図 2.14. ブロック暗号の CFB モード

2.2.3. ブロック暗号の使用モード

ストリーム暗号では、同一の平文 M_i が続けて入力されても、鍵 K_i が次々に変化していくため、暗号文 C_i は異なる値になる。しかし、ブロック暗号では、ストリーム暗号と異なり、通常ある一定の期間同一の鍵が使用される。そのため、図 2.15 に示す ECB(electric codebook)モードと呼ばれる基本的な使用法を用いると、同一の平文がいつも同じ暗号文に暗号化されてしまう。しかし、ブロック暗号を OFB, CFB モードで用いると、実質的にはストリーム暗号として働くため、この問題点は解決される。

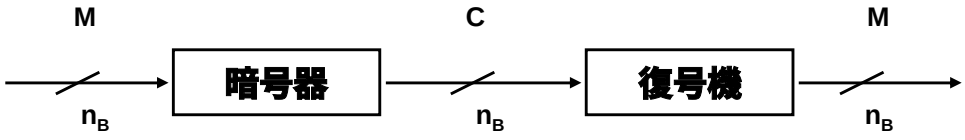

図 2.15. ブロック暗号の ECB モード

OFB, CFB モード以外には同一の平文を入力しても異なる暗号文が出力されるように工夫したブロック暗号の使用法として、図 2.16 に示す暗号文ブロック連鎖(CBC)モードがある。

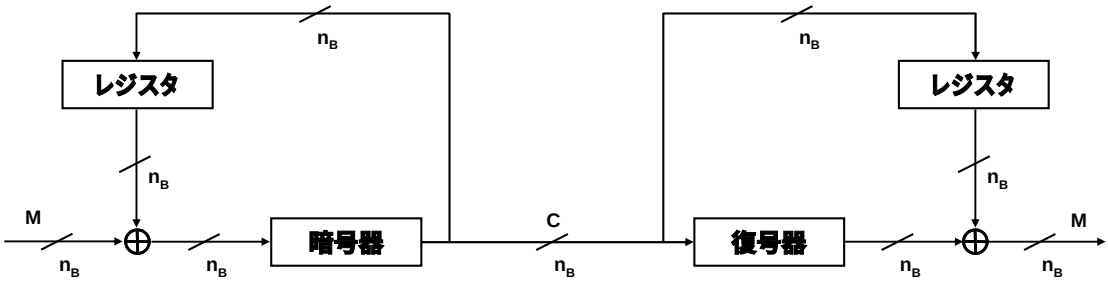

図 2.16. ブロック暗号の CBC モード

CBC モードでは、ブロック長 n_B 単位で暗号化されるが、レジスタに記憶されている 1 つ前の暗号文と次の平文が加算されるため、同一の平文が入力されても異なる暗号文となる。なお、暗号文はレジスタを経由して次々に連鎖を繰り返して行くが、暗号文はレジスタに 1 回記憶されるだけなので、ビット誤りが生じた場合に、その影響が及ぶのはブロック長 n_B の暗号文 2 回分の復号である、誤り伝播が長く続くことはない。

2.3. 情報理論に基づく安全性

本節では、秘密鍵暗号方式に対する安全性を情報理論的に述べる。2.3.1 では情報量を表す尺度であるエントロピーについて述べる。次に、2.3.2 ではエントロピーを用いて、秘密鍵暗号アルゴリズムが安全性を保つためには、どのような条件を満たすべきかについて述べる。2.3.3 では完全秘匿を実現することができるストリーム暗号である、バーナム暗号について述べ、2.3.4 ではストリーム暗号で用いられる乱数生成器について述べる。2.3.5 では、秘密鍵暗号系に対して、盗聴者がどのような攻撃を行うかについて述べる。

2.3.1. 情報量

まず、以下の議論で用いる表記の説明をする。確率変数を X, Y の様に英大文字で表す。また、確率変数が取り得る値の集合(アルファベット)を $\mathcal{X}, \mathcal{Y}$ の様に斜体で表し、$\|X\|$ で $\mathcal{X}$ に含まれる元の数(アルファベットサイズ)を表す。$a \in \mathcal{X}$ に対して、X が値 a を取る確率を $P_X(a) = \Pr[X=a]$ と表記する。この時、情報量を考える際に用いるエントロピーは以下の様に定義される。

[定義 2.1]

確率分布 $P_X(x)$に従う確率変数 X のエントロピーH(X)は次式で定義される。

$$H(X) = -\sum_{a \in X} P_X(a) \log P_X(a) \qquad (2.1)$$

ただし、0log0 = 0 とする。

対数の底が 2 の時、エントロピーの単位はビットとなる。以下では、対数の底として 2 を取るものとする。

(2.1)に示されたエントロピーの定量的な意味について述べる。長さ N の系列 $X^N = (X_1, X_2, X_3, \cdots, X_N)$の各 X_lが同じ確率分布 $P_X(x)$に従って生起している場合を考える(つまり、X_lは無記憶であると仮定する)。各 $a \in X$ が X^N の中で N_a 回出現しているものとすると、大数の法則から、N が大きくなるにつれて、N_a は $NP_X(a)$に近づく。つまり代表系列がほぼ確率 1 で生起し、非代表系列が出現する確率は 0 に近づく。

代表系列 $x^N = (x_1, x_2, \cdots, x_N)$の生起確率を考えると、各 $a \in X$ が確率 $P_X(a)$で $N_a \approx NP_X(a)$改生起していることから、以下の様に $Pr[X^N = x^N]$を導くことができる。

$$\begin{aligned} Pr[X^N = x^N] &= \prod_{a \in X} P_X(a)^{Na} \\ &\approx \prod_{a \in X} P_X(a)^{NPx(a)} \\ &= \prod_{a \in X} [2^{\log Px(a)}]^{NPx(a)} \\ &= \prod_{a \in X} 2^{NPx(a)\log Px(a)} \\ &= 2^N \sum_{a \in X} P_X(a)\log P_X(a) \\ &= 2^{-NH(X)} \end{aligned} \qquad (2.2)$$

(2.2)から、代表系列はその系列 x^N によらず、生起確率がほぼ $2^{-NH(X)}$の一定値となる。また、全ての代表系列の生起確率の和はほぼ 1 であることより、代表系列の個数は $1/2^{-NH(X)} = 2^{NH(X)}$個となる。従って N が一定の場合、代表系列数はエントロピーが大きいほど多くなり、事前にどの代表系列が出現するかを予測することは困難となる。以上から、エントロピーH(X)はデータ系列$\{X_l\}$の事前の曖昧さを定量的に表していることがわかる。

式(3.1)は、1 つの確率変数 X に対するエントロピーの定義だが、結合確率分布 $P_{X,Y}(a, b)$, $a \in X, b \in Y,$ で定まる 2 つの確率変数(X, Y)に対しても、同様にエントロピーH(X, Y)を定義できる。さらに、X が既知の場合の Y の曖昧さを表す情報量として、条件付きエントロピー

H(Y|X)を定義できる。

[定義 2.2]

結合エントロピーは以下の様に定義される。

$$H(X, Y) = -\sum_{a \in X} \sum_{b \in Y} P_{X,Y}(a, b) \log P_{X,Y}(a, b) \quad (2.3)$$

条件付きエントロピーは以下の様に定義される。

$$H(Y \mid X) = -\sum_{a \in X} \sum_{b \in Y} P_{X,Y}(a, b) \log P_{Y|X}(b \mid a) \quad (2.4)$$

また、結合エントロピーおよび条件付きエントロピーについては以下の定理[2.1]が成り立つ。

[定理 3.1]

$$H(Y|X) \leq H(Y) \quad (2.5)$$

$$H(X, Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) \quad (2.6)$$

$$\leq H(X) + H(Y) \quad (2.7)$$

$$H(X, Y) \geq \max \{H(X), H(Y) \quad (2.8)$$

さらに 3 個以上の確率変数に対しても、同様に結合エントロピーや条件付きエントロピーを定義できる。(2.7)により、N 変数の $X^N = (X_1, X_2, \ldots, X_N)$に対して一般に $H(X^N) \leq \sum_{l=1}^{N} H(X_l)$となるが、各 X_l1 が同一の確率分布 $P_X(a)$に従って毎回独立に生起している場合、つまり無記憶の場合、(2.9)が成り立つ。

$$H(X^N) = NH(X) \quad (2.9)$$

2.3.2. 情報理論に基づく安全性

秘密鍵暗号方式を図 2.17 に示す。暗号には平文長が n のブロック暗号を用いるものとし、この暗号の情報理論的安全性を以下に述べる。

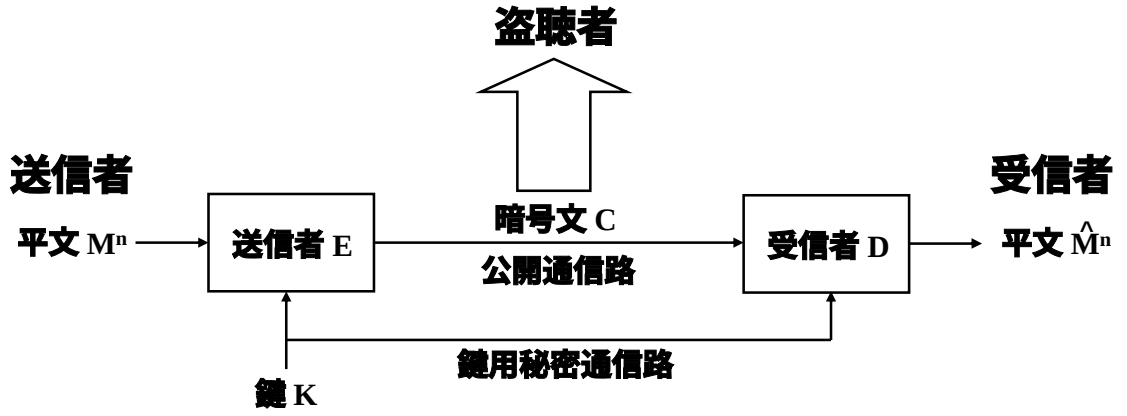

図 2.17. 秘密鍵暗号

暗号化器 E は平文 M^n を鍵 K を用いて暗号化し、暗号文 C を公開通信路を通して復号器 D に伝送する。このとき暗号文 C は盗聴者により、盗聴される可能性がある。一方鍵 K は、盗聴者から完全に秘密に保たれた特別の通信路を通じて伝送される。復号器では、暗号文 C と鍵 K から復号系列 $\hat{\mathbf{M}}^n$ を復号する。

ここで、M は無記憶な系列と仮定し、暗号文 C と鍵 K はそれぞれ L_C 個と L_K 個の異なる値を取るものと仮定する。このとき暗号文のレート R_C(平文 1 シンボル当たりの暗号文のビット長)と鍵のレート R_K(平文 1 シンボル当たりの鍵のビット長)は、以下の様に定義される。

$$R_C = \frac{1}{n} \log L_C \tag{2.10}$$

$$R_K = \frac{1}{n} \log L_K \tag{2.11}$$

通信効率を考えれば、(3.3)と(3.4)は小さい方が望ましい。しかし、平文を正規の受信者に正しく送り、かつ盗聴者から安全に守るためには、ある値以上小さくすることが出来ない。

暗号の安全性は、盗聴者が暗号文 C を知った時の平文 M^n に対する曖昧さの程度を示す条件付きエントロピー$H(M^n|C)$で評価できる。また、正規の受信者には誤りなく平文を復号できる必要がある。このとき、暗号の安全性は以下の様に定義される。また、定義 2.2 は定理 2.1 のように言い換える事ができる。

[定義 2.2]

任意のε>0 と十分大きな n で

$$\frac{1}{n} H(M^n|C) \geq h - \varepsilon \leq h \leq H(M) \quad (2.12)$$

$$Pr[M^n \neq M'^n] \leq \varepsilon \quad (2.13)$$

を満たすような暗号E,Dが存在するとき、レート対$R_C R_K$は安全基準 h が達成可能である。

[定理 2.1]

安全基準 h を達成する暗号(E, D)が存在するための必要十分条件は、レートが次式を満たす事である。

$$R_c \geq H(M) \quad (2.14)$$

$$R_K \geq h \quad (2.15)$$

式(3.5)と式(3.9)から、h=H(M)の場合は、

$$H(M) - \varepsilon \leq \frac{1}{n} H(M^n|C) \leq \frac{1}{n} H(M^n) = H(M) \quad (2.16)$$

が成立する。式(3.7)が任意のε>0 で成立することから、M^n と C は $H(M^n|C) = H(M^n)$を満たす。このことは、定理より、確率的に M^n と C が独立であることを示しており、盗聴者が暗号文 C を得ても、平文 M^n のエントロピーは全く減らず、M^n に関する情報は何も得られない。つまり、安全基準 h = H(M)を達成できる暗号(E,D)は完全秘匿を実現できる。次節では、簡単に完全秘匿通信を実現できるバーナム暗号について述べる。

2.4. バーナム暗号

2.3 では完全秘匿通信を実現するためには、暗号文と鍵のレートがともに平文のエントロピーH(M)以上必要な事を示した。この完全秘匿通信は図 2.18 に示すバーナム暗号により実現できる。

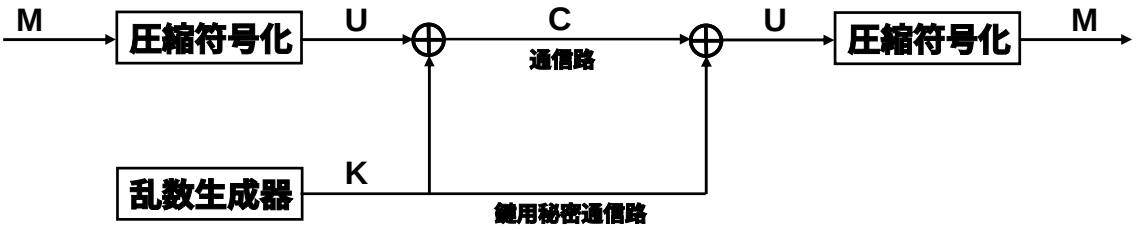

図 2.18. バーナム暗号

平文 M が{0, 1}の 2 値系列からなる場合、真の 2 値系列からなる場合、真の 2 値系列 K (Pr[K = 0] = Pr[K = 1] = 0.5)を鍵として使用し、ビットごとに C = M ⊕ K と排他的論理和を取って暗号文 C を作れば、C と M は互いに独立となり、完全秘匿が達成できる。

平文 M が H(M) < 1 で冗長度を持っていたり、あるいは M が 2 値でない場合は、単に 2 値乱数 K を加算しただけでは暗号文と鍵のレートを H(M)まで下げることはできない。しかし、図 2.18 に示すように、算術符号などを用いて平文 M^n を圧縮符号化すれば、n シンボルの平文 M^n をほぼ nH(M)ビットに圧縮でき、暗号文および鍵のレートともほぼ H(M)ビットで、完全秘匿通信が可能となる。

次に安全基準 h が H(M)より小さい場合に、バーナム暗号を用いて(2.12)-(2.15)を達成する方法を示す。

p = Pr($\hat{\mathbf{X}}$ = 1)の生起確率を持つ長さ $\tilde{\mathbf{n}}$ の 2 値系列 $\hat{\mathbf{X}}^{\tilde{\mathbf{n}}}$ を、算術符号を用いて圧縮符号化すると、Pr[X = 1] = 0.5 の長さがほぼ $\tilde{\mathbf{n}}\,H_2(p)$ である系列 $\mathbf{X}^{\tilde{\mathbf{n}}\mathbf{H_2(p)}}$ が得られる。ただし、$H_2(p) = -p \log p - (1-p) \log (1-p)$を表す。また、復号により逆の変換が行える。そこで、Pr[K = 1] = 0.5 である長さ nh の系列 K^{nh} に対して、確率 p 用の算術符号の復号処理を行うと、p = Pr[$\hat{\mathbf{K}}$ = 1]を持つ長さ $nh/H_2(p)$ の 2 値系列 $\hat{\mathbf{K}}^{\mathbf{nh/H_2(p)}}$ が得られる。一方、長さ n の暗号文 M^n は圧縮符号化により、長さ nH(M)の 2 値系列 $U^{nH(M)}$に符号化できる。従って、$nh/H_2 = nH(M)$を満たすように p の値を設定すれば、図 2.19 の構成で$\{U_i\}$と$\{\hat{\mathbf{K}}_\mathbf{i}\}$を 1 対 1 に対応付けて $U_i \oplus \hat{\mathbf{K}}_\mathbf{i}$ と暗号化できる。このとき、nh ビットの鍵は、長さ n の平文にほぼ一様に影響を及ぼし、(2.12)が満たされる。また、長さ n の平文 M^n1 つ当たり、暗号文と鍵に nH(M)ビットと nh ビットを使用しているので、暗号文と鍵のレートはそれぞれ H(M), h となり(2.14), (2.15)が満たされる。

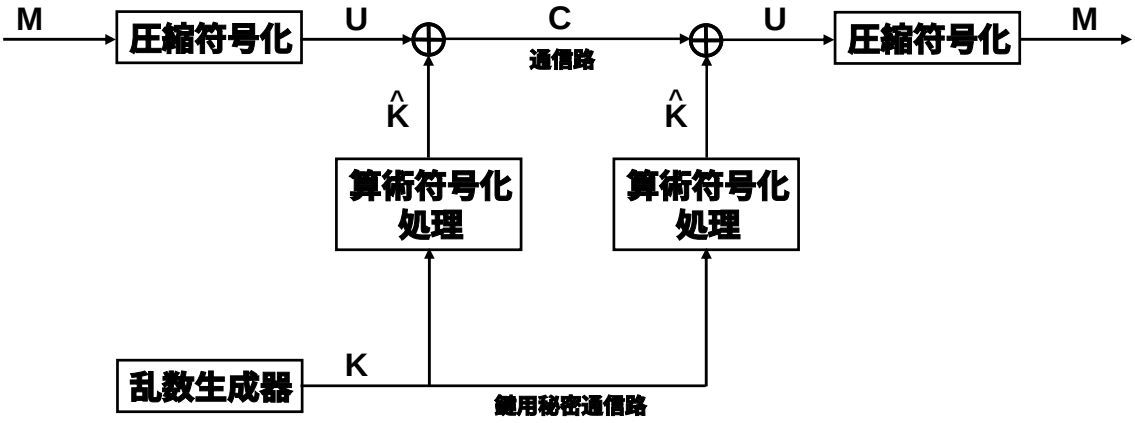

図 2.19. 改良バーナム暗号

2.5. 擬似乱数

2.4 で述べたバーナム暗号では、送信者と受信者が乱数を共有しなければならない。安全性を保つためには全くランダムな数値である乱数を用いることが望ましい。しかしながら、真の乱数は共有することが難しく、多くの場合は擬似乱数が真の乱数の代わりに用いられている。

2.5.1. 擬似乱数としての条件

擬似乱数は見かけ上ランダムな系列であるが、適当な長さの種から決定的なアルゴリズムによって生成される。従って、種を送信者と受信者の間で共有できれば、長い擬似乱数を共有できる。しかし、決定的なアルゴリズムで生成されているため、擬似乱数の一部から他の部分が推測される恐れがある。そのような危険が生じないように、暗号で使用する擬似乱数は、次のような性質を満たす必要がある。

1. 統計的一様性
2. 無相関性
3. 長周期性
4. 非線形性

暗号の分野では、次に述べる線形フィードバックシフトレジスタ(linear feedback shift register, LFSR)を用いると、長い周期の 2 値乱数系列を簡単な装置で高速に生成できる。しかし、その線形性のため、LFSR で生成される系列も予測不可能さの観点から望ましくない。予測を困難にするために、LFSR に非線形変換を組み合わせた乱数生成装置が用いられることが多い。

2.5.2. 線形フィードバックシフトレジスタ

図 2.20 に 2 値の線形フィードバックシフトレジスタ(LFSR)を示す。図中の四角がシフトレジスタを表し、1 クロックごとに→の向きにシフトする。丸の中に書かれている a_i は、LFSR の結線状態を表しており、$a_i = 0$ の時は切断されていることを示す。従って、この LFSR により生成される系列$\{r_i\}$は(2.17)を満たす。

$$r_i = a_1 r_{i-1} + a_2 r_{i-2} + \ldots + a_L r_{i-L} \qquad (2.17)$$

ただし加算は排他的論理和を表す。

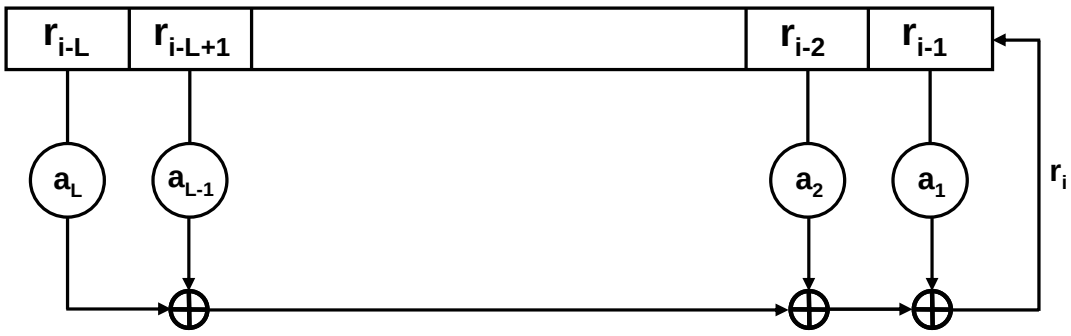

図 2.20. 線形フィードバックシフトレジスタ

L 個のレジスタからなる LFSR は、そのレジスタに記憶している値により 2^L 個の状態が存在する。しかし、全レジスタの値が 0 である状態(ゼロ状態)は、a_i の値に関わらず、その次の時刻でもやはりゼロ状態となり、LFSR の出力は常に $r_i = 0$ となってしまう。従って、ゼロ状態を除いた $2^L - 1$ 個の状態が、時刻の経過につれて全て現れるとき、LFSR の出力系列$\{r_i\}$は最も長い周期 $2^L - 1$ を持つ。このような最長周期を持つ系列を M 系列と呼ぶ。

LFSR 出力系列の特性は、配線 a_i の値により決まるが、特性多項式(2.18)が L 次の原始多項式の時に、系列$\{r_i\}$は M 系列となる。

$$a(x) = a_0 + a_1 x + a_2 x_2 + \ldots + a_L x_L,\ a0 = 1 \qquad (2.18)$$

M 系列では、ゼロ状態を除いて他の状態が 1 周期の間に全て現れることより、1 周期当たりの等頻度性や無相関性が優れている。しかし、LFSR は線形操作のみで構成されているため、過去の系列から未来の系列が容易に求められる。実際 L 段の LFSR では、(2.17)を用いて、2L - 1 個の a_i を未知数とした L - 1 個の方程式が得られ、a_i が求められる。その結果、その後の r_i が完全に分かってしまう。

2.6. 盗聴者による攻撃

暗号化アルゴリズムを暗号の設計者以外にいつまでも秘密に保つことは一般に困難であり、また暗号の使用者は暗号の設計者からも情報を秘密に保ちたいという要求がある。そのため、暗号は、その暗号アルゴリズムが攻撃者に知られているという仮定の下でも安全であることが要求される。言い換えると、暗号は、暗号アルゴリズムが公開されており、その安全性は秘密鍵を敵から守ることにより達成できるものでなければならない。

暗号の攻撃法は、攻撃者が利用できる平文および暗号文の種類により次の 4 通りに分類できる。

1. 暗号文攻撃

暗号文だけが利用できる場合(図 2.21)。

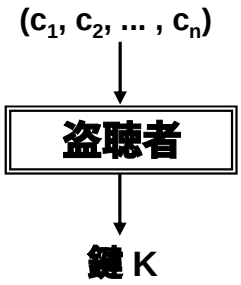

図 2.21. 暗号文攻撃

2. 既知平文攻撃

平文と対応する暗号文が利用できる場合(図 2.22)。

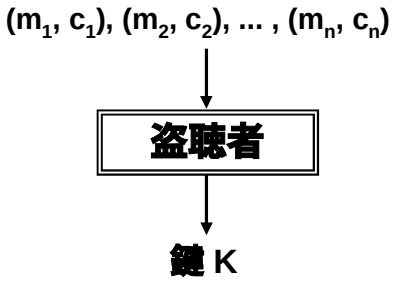

図 2.22. 既知平文攻撃

3. 選択平文攻撃

盗聴者が任意に選択した平文とそれに対応する暗号文が利用できる場合(図 2.23)。

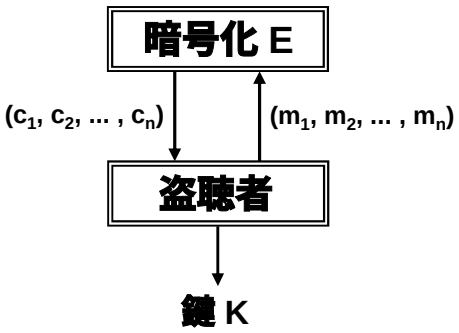

図 2.23. 選択平文攻撃

4. 選択暗号文攻撃

選択平文攻撃に加え、盗聴者が任意に選択した暗号文とそれに対応する平文が利用できる場合(図 2.24)。

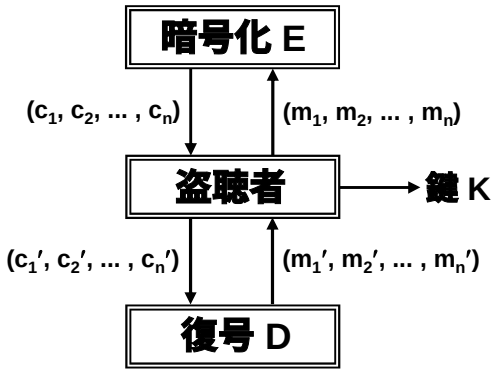

図 2.24. 選択暗号文攻撃

上記の条件を満たす暗号文と平文は、攻撃者にとって 1→4 の順番で入手が困難になる。上記 2-4 では、暗号文に対応する平文がわかっているという条件での攻撃であるが、一般に公開された通信システムで用いられている暗号では、攻撃者が自分に都合の良い平文をその通信システムを通して伝送することにより、対応する平文・暗号文対をいくつも入手することが可能となる。それらを用いた攻撃により使用していた鍵がわかれば、平文が未知な暗号文を解読できることになる。安全な秘密鍵暗号としては、暗号化のアルゴリズムが公開されていても、上記のような攻撃に対して十分強いことが求められる。

第 3 章以降では、提案する暗号手法について述べるが、上述の 4 つの攻撃に対しての安全性が保証されていなければならない。バーナム暗号はこれらの攻撃に対して完全秘匿を実現するが、真の乱数を生成する乱数生成器を送信者と受信者間で共有することが困難であることから、実質的にはバーナム暗号の実現が困難である。そこで、乱数生成器を共有する必要が無く、バーナム暗号と同じ機能を持つ暗号手法があれば、それは魅力的な暗号

方式であると言える。以下の章では、乱数生成器を共有せずともバーナム暗号と同じ機能を持つ暗号を検討する。

第 3 章　双方向通信を用いた暗号方式

3.1. シグマデルタ変調器を用いる暗号方式

3.1.1.　暗号化

提案手法では、従来の暗号化方式で用いられている鍵の代わりに雑音を用いる。送信データに雑音を掛けることで暗号化を行うが、暗号化されたデータの周波数スペクトルを見たときに図 4.1 に示すような特徴を持った雑音を考える。

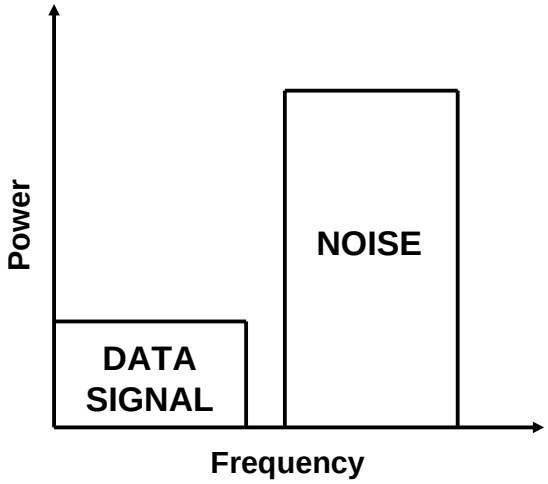

図 3.1 暗号化された送信データの周波数スペクトル

図 3.1 に示すように、暗号化されたデータの周波数スペクトルを見ると、送信データ(低周波数領域)と雑音(高周波数領域)の周波数帯域が分かれていることがわかる。この様にする理由は 3.1.3 の通信方式で述べる。また、本研究では、以上のような特徴を持った雑音を、3.1.4 で述べるΣΔ変調器を用いることで生成する。

3.1.2.　復号化

図 3.1 の様に暗号化されたデータの復号を行うためには、Low Pass Filter を用いる。暗号化されたデータを Low Pass Filter に通すことで、高周波数領域に存在するノイズを除去し、データを取り出す。

3.1.3.　通信方式

通信を行う 2 人(P,Q とする)が通信を行う場合の通信手順を図 3.2 に示す。

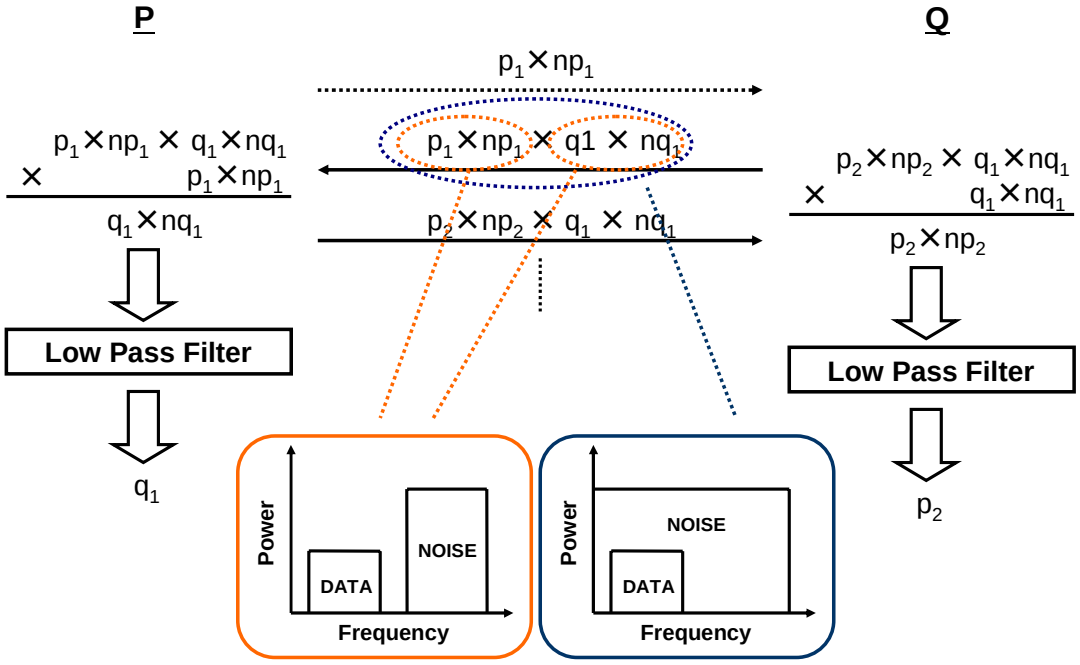

図 3.2. 通信方式

ここで、図 3.2 において、

p_i(i:自然数) : P が送信するデータ

q_i(i:自然数) : Q が送信するデータ

np_i(i:自然数) : P が暗号化に用いる雑音

nq_i(i:自然数) : Q が暗号化に用いる雑音

を表す。通信手順を説明する。

1. P が送信するデータに雑音を掛けたもの($p_i \times np_i$)を Q と共有する。これを秘密鍵とする。
2. Q は共有している秘密鍵($p_i \times np_i$)に自分の送信データと雑音をかけて($p_i \times np_i \times q_i \times nq_i$)P に送信する。
3. P は受信したデータ($p_i \times np_i \times q_i \times nq_i$)に自分が送信したデータ($p_i \times np_i$)をかけて、暗号化された Q の送信データ($q_i \times nq_i$)を得る。それを LPF に通すことで、Q のデータ(q_i)を得る。

以上が通信手順で、Q が P の送信データを得たい場合も同様に復号を行う。ここで、盗聴者が傍受できる P と Q の間の通信路におけるデータに注目すると、図 3.3 に示すように、2 つの高周波数領域にノイズが集まった信号を掛け合わせると、ノイズのスペクトルが低周波から高周波まで均一に分布する白色雑音となる信号になる。そのため、データの信号はノイズの影響を受け、盗聴者は 2 人の通信者と同様にローパスフィルタを用いても、信号を復号することができない。

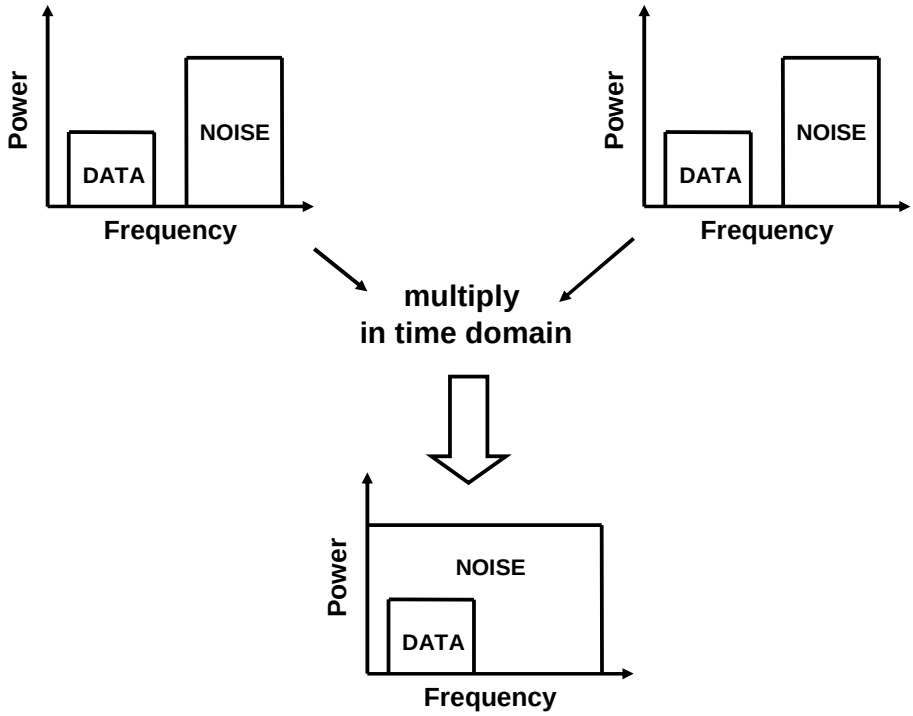

図 3.3. 盗聴者は白色雑音を得る

3.1.4. ΣΔ変調器

提案手法では、送信データにノイズを掛けるために、ΣΔ変調器を用いる。そこで、ΣΔ変調器について述べる。ΣΔ変調器は主に AD 変換に用いられる。AD 変換器による量子化ノイズは、信号帯域内に平坦なパワースペクトラムで分布する白色雑音とみなすことができる。(ローパス)ΣΔ変調器は、図 4.4 に示すように、この平坦なノイズスペクトルを低周波領域から高周波領域に追いやることができる。これをノイズシェーピングと呼ぶ。

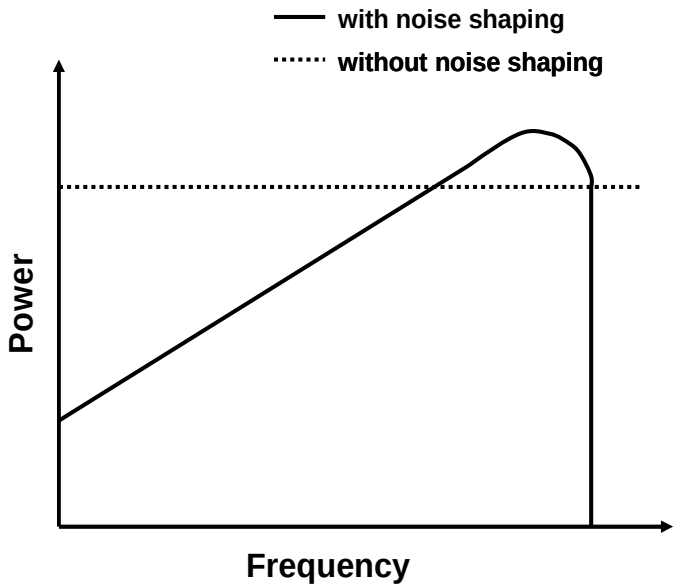

図 3.4. ノイズシェーピング

ΣΔ変調器の構成

ΣΔ変調器では、図 3.5 に示すような離散時間システムにおける積分器を用いて過去の AD 変換の結果を利用することで、量子化雑音を高周波領域に押しやる。

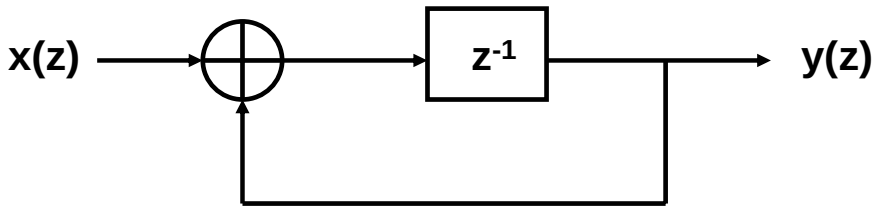

図 3.5. 離散時間システムにおける積分器

ここで、z^{-1} はサンプル周期の遅延を表し、順次遅延した信号を元の信号に加算する。このようにして、積分器として動作することがわかる。また、この積分器の伝達関数 H(z)は、

$$H(z)=\sum_{n=0}^{\infty} z^{-n}=\frac{1}{z-1}$$

となる。次に、この積分器を用いて、Fig に示すような構成を考えると、伝達関数は、

$$y(z) = z^{-1}x(z) + (z-1)z^{-1}e(z)$$

となる。ここで、e(z)は量子化器によって生ずる量子化ノイズを表す。また、図 4.5 の構成では積分器を 1 つ用いているので、1 次のΣΔ変調器という。

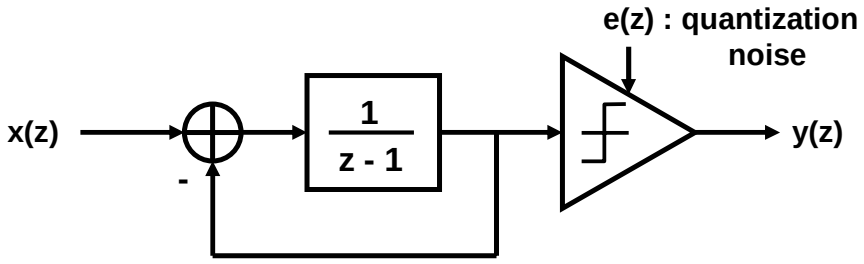

図 3.6. 1 次のΣΔ変調器

伝達関数を見ると、入力は遅延を経てそのまま出力される。しかし、量子化ノイズ e(z)を含む項は、z=1 に零点を持つため、低周波領域におけるノイズが最小となるようになっていることがわかる。図 3.6 の x(z)に直流を与えた時の y(z)の周波数スペクトルを図 3.7 に示す。

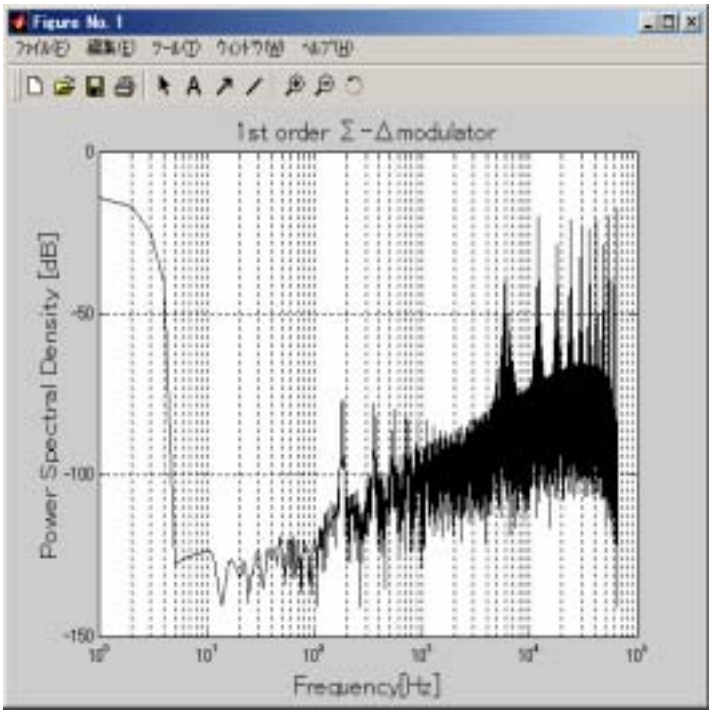

図 3.7. 1 次のΣΔ変調器の周波数スペクトル

図 3.7 において低周波のスペクトルが強くなっているのは、入力に DC を入れたため、それが現れている。量子化ノイズ e(z)について見ると、低周波数領域では押さえられ、高周波数領域に押しやられていることがわかる。しかし、図 3.7 にも見られるように、1 次のΣΔ変調器ではスプリアストーンが生じてしまうことが知られている。

そこで、積分器を 2 つ用いた 2 次のΣΔ変調器を考える。2 次のΣΔ変調器を図 3.8 に示す。

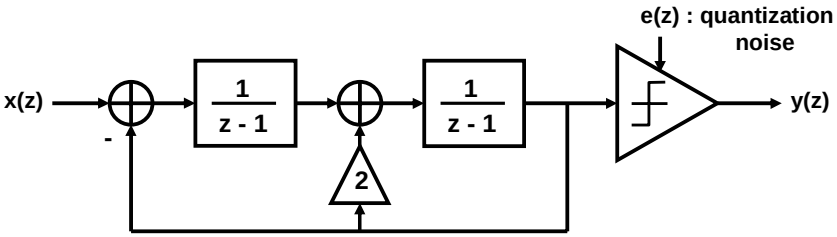

図 3.8. 2 次のΣΔ変調器

1 次の時と同様に、伝達関数を求めると、

$$y(z) = z^{-2}x(z) + (z-1)^2 z^{-2} e(z)$$

となる。1 次の時と同じ様に、入力は遅延を経てそのまま出力される。その一方で、e(z)に関しては、z=1 で零点を持つために低周波数領域でノイズが最小となる。Fig における x(z)に直流を与えたときの y(z)の周波数スペクトルを図 3.9 に示す。

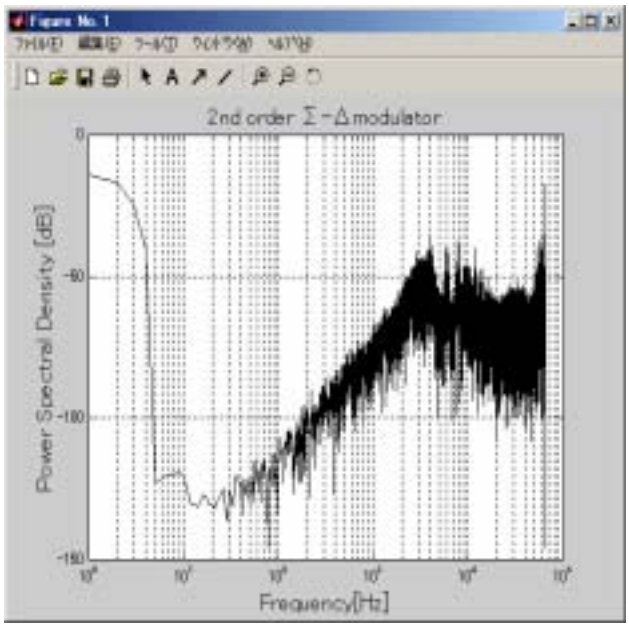

図 3.9. 2 次のΣΔ変調器の周波数スペクトル

図 3.7 と図 3.9 を比較すると、２次のΣΔ変調器ではスプリアストーンが出ておらず、また、量子化ノイズがより多く高周波数領域に押しやられていることがわかる。以上に述べた理由から本手法では２次のΣΔ変調器を用いた。

3.1.5. ΣΔ変調器を用いた暗号化

本手法では、暗号化の鍵に雑音を用いる。その雑音の生成にΣΔ変調器を用いる。暗号化の手順としては以下のようになる。

1. 送信データを±1 の信号に直す。
2. 1 で得た信号をΣΔ変調器の入力として与える。
3. 出力として得たデータを暗号文として用いる。

これらの手順を図 3.10 に示す。

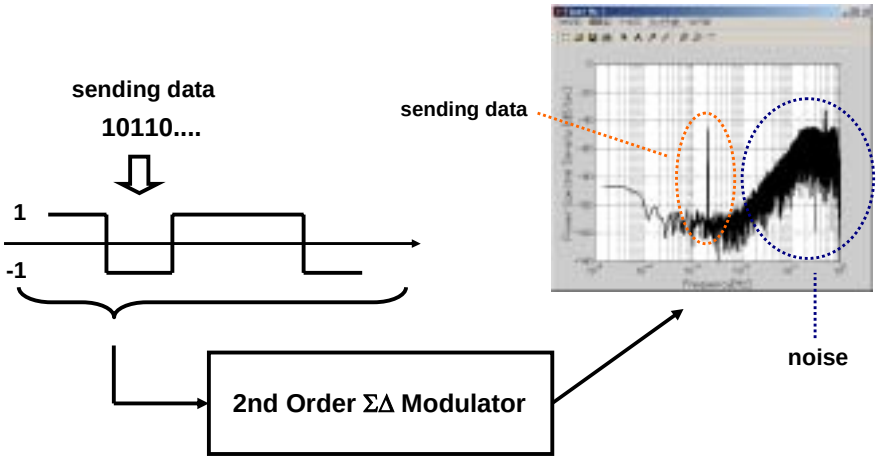

図 3.10. ΣΔ変調器を用いた暗号化

図 3.10 に示す暗号化された送信データのスペクトルを見ると、送信データ(低周波数領域)とノイズ(高周波数領域)の周波数帯域が分かれていることがわかる。また、図 3.11 に上記のスペクトルの特徴を持った信号を時間軸上で掛け合わせた信号のスペクトルを示す。

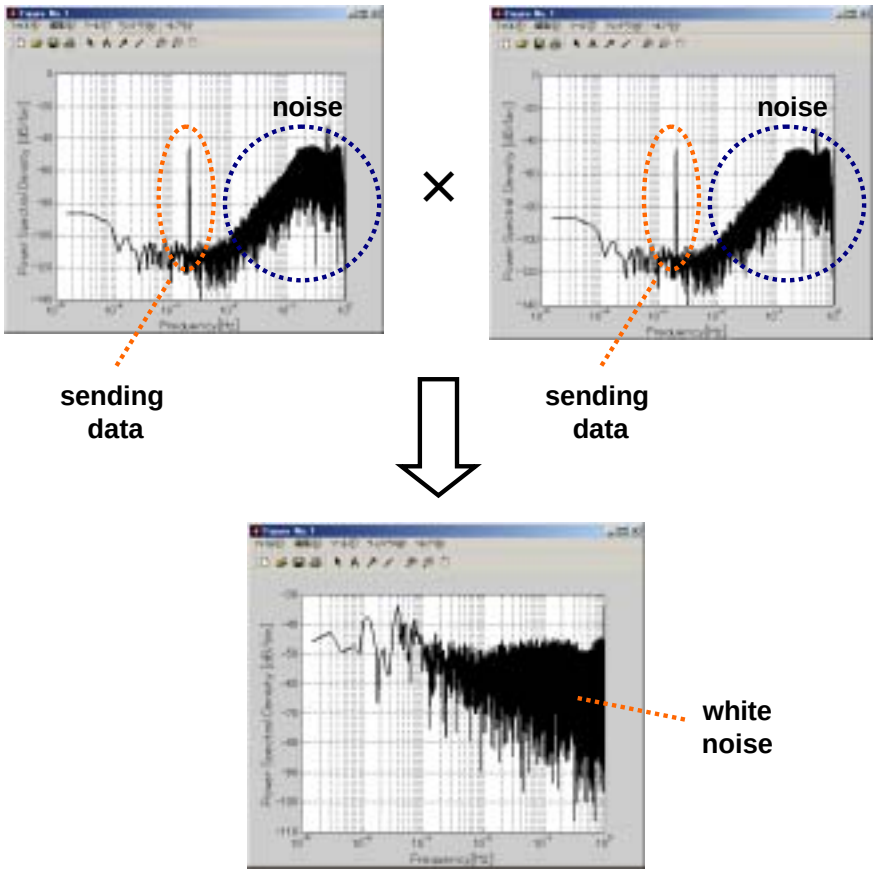

図 3.11. 信号の掛け合わせ

前節の「通信方式」でも述べたが、高周波数領域にノイズが集まった信号同士を掛け合わせると、ノイズが低周波数領域から高周波数領域まで均一に分布して白色雑音となる。2 人の通信者の間の通信路ではこの白色雑音がやりとりされるため、例え傍受されたとしても、ローパスフィルタによって復号されることは無い。以上のようにして、本研究ではΣΔ変調器を用いることで送信データの暗号化を行うことを提案した。

3.1.6. 本手法のまとめ

本手法では、ΣΔ変調器を用いることにより高周波に雑音が乗った信号を生成し、その信号同士を掛け合わせる事で、白色雑音になることをシミュレーションにより示した。そして、その特徴を用いた暗号手法を提案した。しかし、ΣΔ変調器では、完全な乱数を生成することはできず、乱数が周期的に生成されるため、盗聴者による解読が可能となってしまう。そこで、3.2 節では擬似乱数ではなく、お互いが共有する必要がない真の乱数を用いる手法を提案した。

3.2. 送信者が用いる鍵を受信者が決める手法

3.2.1. アルゴリズム

提案する手法の基本的なコンセプトを図 3.12 に示す。

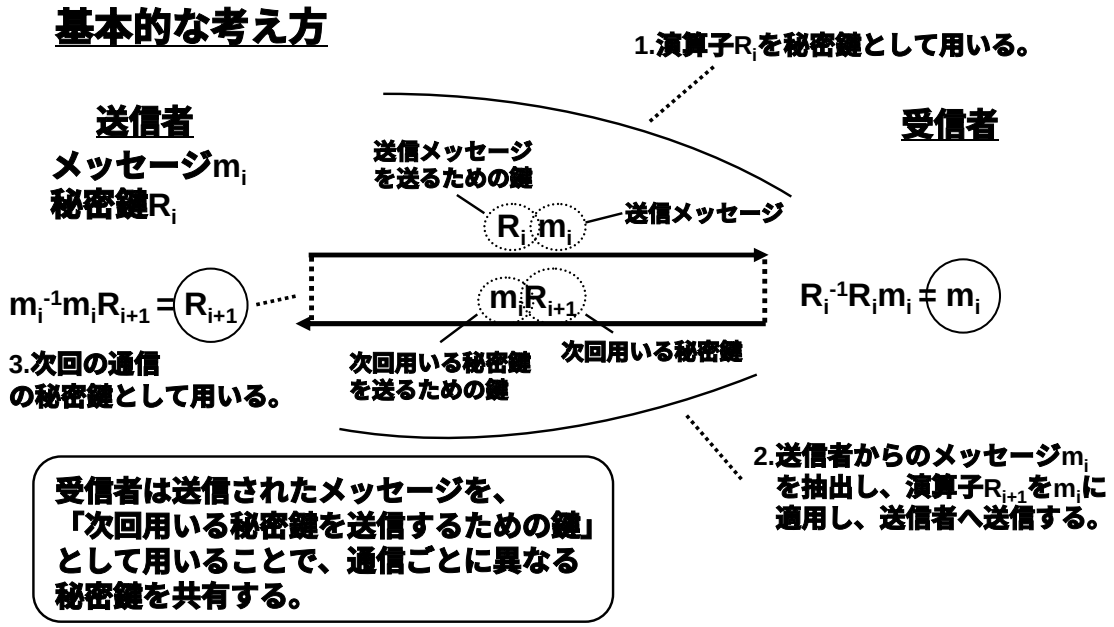

図 3.12. 基本的な考え方

本手法では、送信者が暗号化のために演算子 R_i を秘密鍵として用いる。受信者は、受信したメッセージを使い、次回の通信で用いる秘密鍵を送信者に送付する。送信者は受信者から受信した秘密鍵を用いて次回のメッセージの暗号化を行う。図 3.13 に全体の流れを示す。

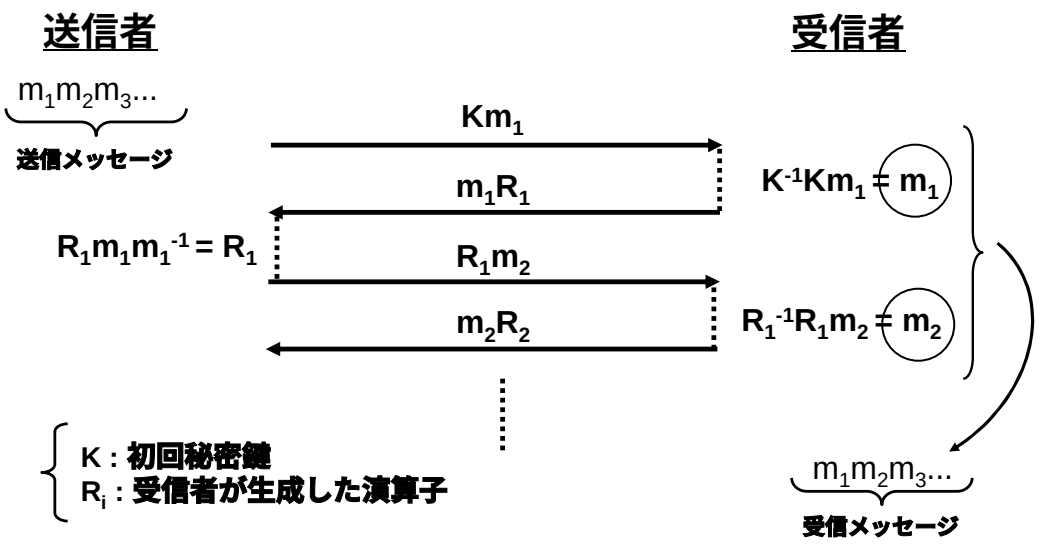

図 3.13. 提案手法の全体の流れ

本手法では 3.1 節で提案した手法と同様に双方向で通信を行う。図 3.13 に示す演算子を XOR で実現すれば回路規模の大幅な削減が実現できる。図 3.14 に演算子を XOR とした場合の全体の流れを示す。

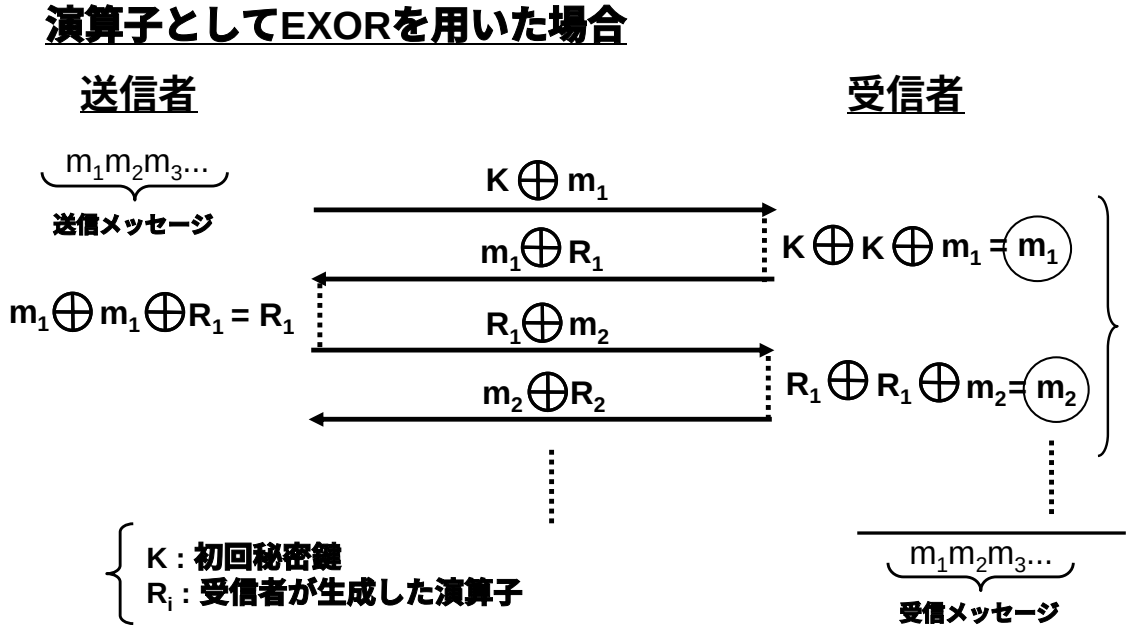

図 3.14. 演算子として XOR を用いる。

3.2.2. 本手法のまとめ

本手法では、受信者が指定した乱数を送信者が通信の鍵として用いる事で、お互いが乱数の系列を共有することなく、バーナム暗号と同様に、通信の度に異なる鍵を用いる事ができた。しかしながら、本手法および 3.1 節の雑音を用いる手法では、選択平文攻撃に対する対策がなされていない。例えば、図 3.14 において盗聴者が m_1 を知っている場合には、受信者が送信する m_1R_1 を得る事で、盗聴者は R_1 の内容がわかる事になる。R_1 がわかれば、盗聴者はその後の通信の内容を全て解読できることになる。第 4 章で提案する手法では、このような第 3 章で述べた盗聴者による攻撃に対する安全性にも考慮した。

第 4 章　キーロンダリングに基づく小型ストリーム暗号回路

4.1. 提案する暗号手法の概要

第 2 章でも述べたように、現在主に用いられている暗号方式は回路面積や消費電力の制限から、センサネットワークなどで用いられている超小型暗号方式に用いることはできない。従来の共通鍵暗号方式では、1 つの秘密鍵を暗号化に用い、その鍵を複雑な演算を用いる事で、盗聴者に知られないようにするというアプローチが取られていた。その結果として、回路規模や消費電力を犠牲にせざるを得なかった。提案する暗号手法では小型で安全な暗号回路を実現するため、暗号化を一時鍵生成とメッセージの送信の 2 つの操作に分離している。4.2 では提案する暗号方式のアルゴリズムを述べる。4.3 では第 3 章で述べた安全性について議論する。

4.2. アルゴリズム

まず、以下で用いる用語の定義を行う。

- n : 送信の際に用いる鍵のビット長。
- i : 通信を行った回数。
- m_i [n bit] : 送信するメッセージ。
- ks_i [n bit] : 送信の際に用いる鍵。
- ss_i (secret seed)[n bit] : 送信の際に用いる鍵を生成するための乱数。通信のたびに異なる乱数を生成し用いる。
- $c1_i$ [n bit] : ss_i を暗号化した符号。
- $c2_i$ [n bit] : m_i を暗号化した符号。
- ki1 [2n bit] : ss_i を暗号化するための鍵。
- ki2 [2n bit] : ks_i を生成するための鍵。

図 6.1 はアルゴリズムのコンセプトを表している。図 6.1 に示すように、送信者はデータを 3 回暗号化する。1 回目は生成した乱数を暗号化し受信者に送信するため、2 回目は ks_i を生成するため、そして 3 回目は送信メッセージを暗号化するためである。

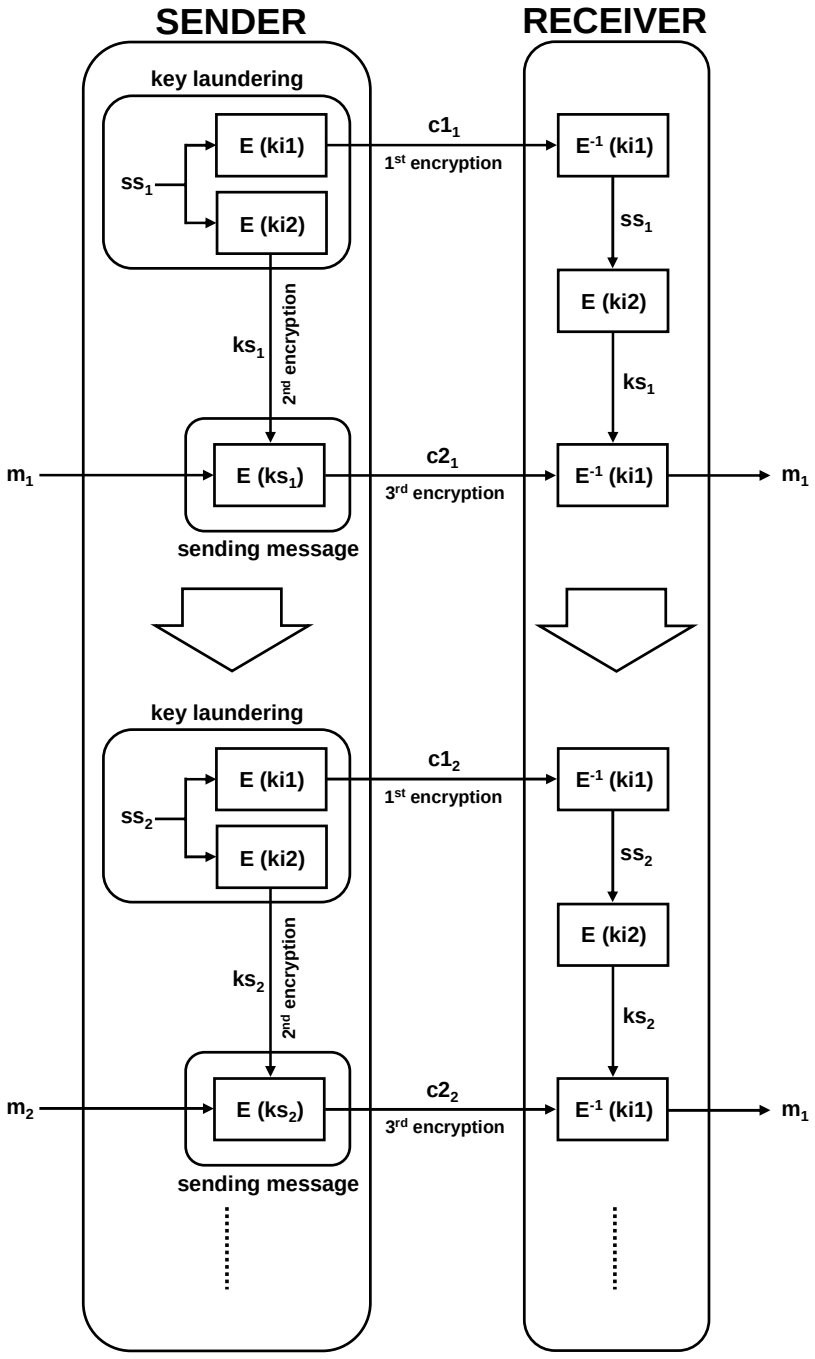

図 4.1. アルゴリズムのコンセプト

提案する暗号方式では、暗号化を一時鍵生成とメッセージの送信の 2 つの操作に分離している。送信者は、送信メッセージを送信する前に一時鍵の生成を行う。送信者は一時鍵の生成により、送信するメッセージを暗号化する鍵を生成する。

4.3. 満たすべき条件

第 2 章でも述べたように、アルゴリズムは以下の 4 つの攻撃に対して安全でなければならない。

1. 暗号文攻撃
2. 既知平文攻撃
3. 選択平文攻撃
4. 選択暗号文攻撃

まず、初めにセンサネットワークにおいて選択平文攻撃がどのように行われるかを図 4.2 に示す。

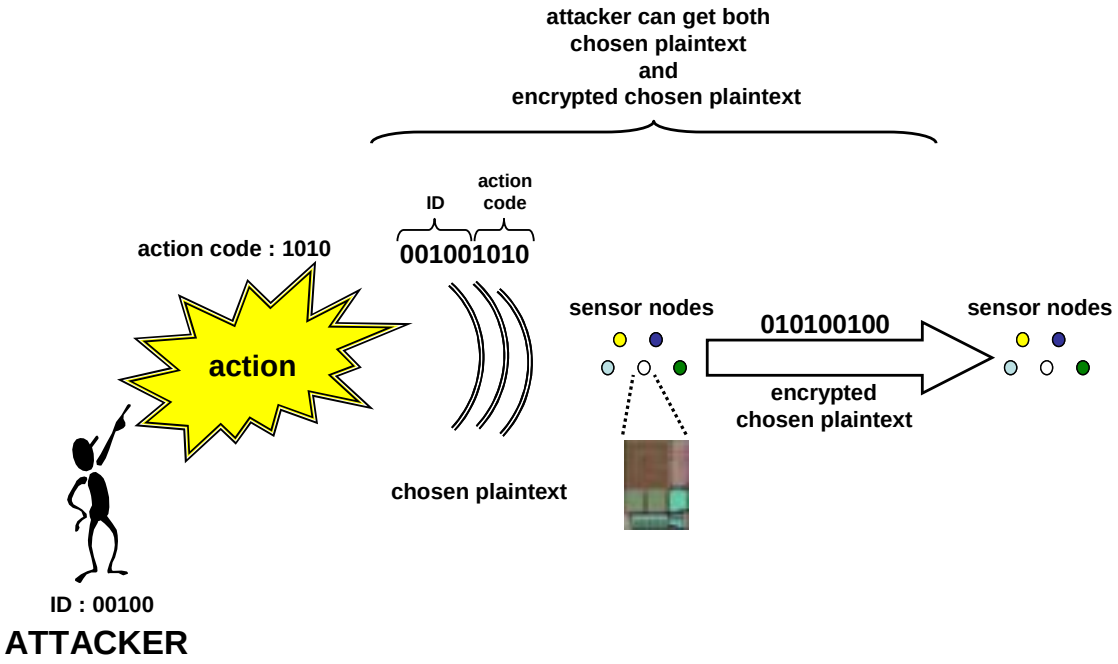

図 4.2. センサネットワークにおける選択平文攻撃

盗聴者が何かしらの行動(例えば、部屋に入る)を起こした場合、センサノードは、そのアクションを感知し、感知したデータを暗号化し、他のノードに送信する。もちろん、送信者は自分が起こした行動についてはわかっているため、平文を選択し、それに対応する暗号文を入手することができる。つまり、センサノードは、感知した行動が、盗聴者により

意図的になされたものだとわからずに、通信を行う。

盗聴者は、図 4.1 において選択平文攻撃および選択暗号文攻撃を行うことができない。理由を以下に示す。盗聴者は、上述したように、センサノードがどのような情報を感知するか知っている場合に選択平文攻撃が可能である。しかしながら、ss_i はセンサノードによって生成され、盗聴者はその内容を知ることはできない。よって、選択平文攻撃および選択暗号文攻撃は一時鍵生成においては有効ではない。反対に、メッセージの送信の操作は選択平文攻撃および選択暗号文を受ける可能性がある。その結果、盗聴者に ks_i が知られる可能性がある。以上から、アルゴリズム全体が安全性は一時鍵生成の図 4.3 に示す特殊な暗号文攻撃に対する安全性を考えればよい。これは、図 4.1 における受信者側に注目すれば、図 4.4 に示す既知平文攻撃に対する安全性と等価である事がわかる。

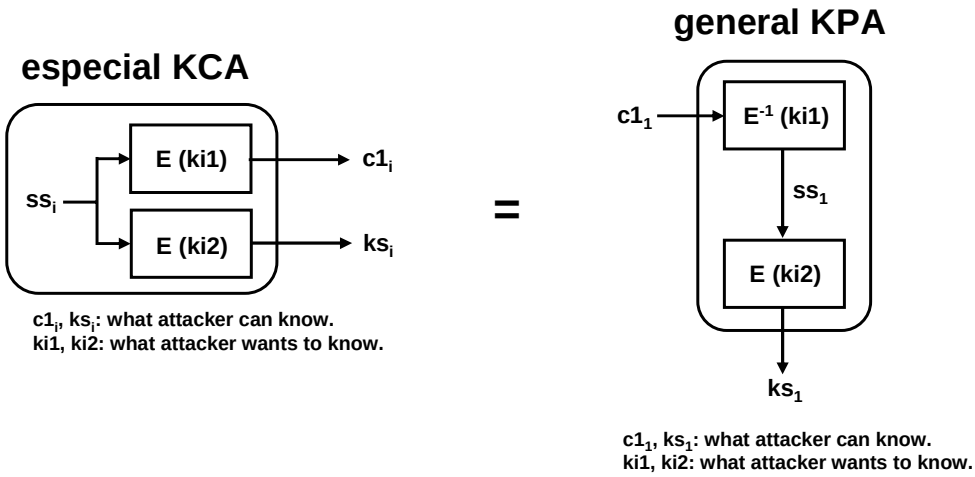

図 4.3. key laundering は一般的な既知平文攻撃に対し安全であればよい。

以上から、アルゴリズム全体の安全性を保つためには一時鍵生成が以下の条件を満たせばよい。

メッセージの送信が満たすべき条件

一時鍵生成の図 4.3 に示す既知平文攻撃に対して安全である。 (cond 4.1)

4.4. アルゴリズムの具体例

(cond 6.1)を満たす一時鍵生成とメッセージの送信は色々と考えられるが、以下ではその一例を示す。

4.4.1. 一時鍵生成

送信者手順

Step1. Generate random number ss_i.

Step2. ss_i1, $ss_i2 \leftarrow ss_i$.

Step3. $ss_i1 \leftarrow ss_i1 \oplus ki1[n:2n-1]$.

Step4.

```
for (x=0; x<n; x++) {
    if (ki1[x] = = 1) {
        Permutate ssi1[x] and ssi1[x+1]. (Note that ssi1[n] implies ssi1[0])
        }
    }
```

Step5. $c1_i \leftarrow ss_i1$.

Step6. $ss_i2 \leftarrow ss_i2 \oplus ki2[n:2n-1]$.

Step7.

```
for (p=0; p<n; p++) {
    if (ki2[p] = = 1) {
        Permutate ssi2[p] and ssi2[p+1]. (Note that ssi2[n] implies ssi2[0])
        }
    }
```

Step8. $ks_i \leftarrow ss_i2$

受信者手順

Step1. Calculate ss_i from received $c1_i$ and ki1.

Step2. $ss_i \leftarrow ss_i \oplus ki2[n:2n-1]$.

Step3.

```
for (p=0; p<n-1; p++) {
    if (ki2[p] = = 1) {
        Permutate ssi[p] and ssi[p+1]. (Note that ssi[n] implies ssi[0])
        }
    }
```

Step4. $ks_i \leftarrow ss_i$

4.4.2. メッセージの送信

送信者手順

Step1. $c2_i \leftarrow ks_i \oplus m_i$

受信者手順

Step1. Calculate m_i from received $c2_i$ and ks_i.

4.5. 各操作の安全性

本節では 4.4 節で挙げた、一事鍵生成, メッセージの送信の具体例における安全性について述べる。

4.5.1. メッセージの送信

メッセージの送信の安全性に関しては、ks_i の安全性が保障されていれば、第 2 章で述べたバーナム暗号と同じになる。すなわち、盗聴者による暗号文攻撃に対しては、安全(ks_i のエントロピーが n に保たれる)だが、既知平文攻撃、選択平文攻撃そして選択暗号文攻撃が行われれば、ks_i は一意に特定する事ができる。ks_i 特定されたとしても ki1, ki2 は特定されてはならない。

4.5.2. 一時鍵生成

一時鍵生成が(cond 4.1)の条件を満たすかどうかについて検討する。初期鍵 ki1, ki2 は図 4.5 に示すように、XOR を取る部分と入れ替えを行う部分の 2 パートに分かれている。そこで、ここでは XOR を行うビットを $ki1_a$ [n bit], $ki2_a$ [n bit]とし、入れ替えを行う部分を $ki1_b$ [n bit], $ki2_b$ [n bit]とおく。

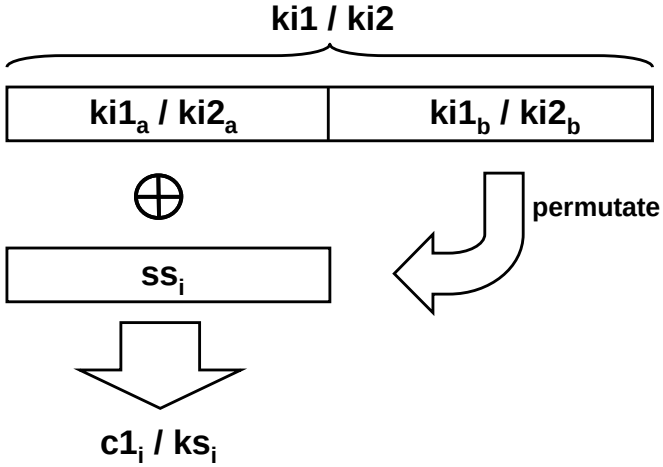

図 4.4. $ki1_a$, $ki1_b$, $ki2_a$, $ki2_b$ の定義

図 4.3 に示す特殊な暗号文攻撃に対する安全性を示すことで、(cond4.1)を満たす事を証明する。図 4.4 の定義を用いて、図 4.3 の暗号文攻撃をより具体的に書き直すと図 4.5 になる。

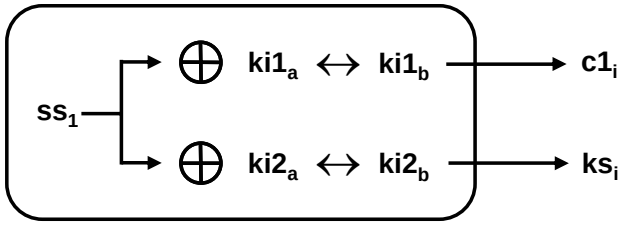

図 4.5. 安全性を保つべき暗号文攻撃 (sending message において選択平文攻撃および選択暗号文攻撃を受けた場合は、盗聴者は ks_i を知ることができる)

(cond4.1)を満たすためには、以下の 2 条件を満たす必要がある。また、本手法では、ss_i に対して XOR を取り、入れ替えを行っているが、以下の議論では、その演算を"∗"で表す。

・ (cond 4.1a)

$c1_i$ と ks_i は独立である必要がある。以下にその理由を示す。まず、$c1_i$ と ks_i は(4.1), (4.2)を満たす。

$$c1_i = ki1 * ss_i \quad (4.1)$$

$$ks_i = ki2 * ss_i \quad (4.2)$$

もし、$c1_i$ と ks_i が独立でなければ、(4.3)を満たす関数 f が存在することになる。

$$ks_i = f(c1_i, ki1, ki2) \quad (4.3)$$

盗聴者は、一時鍵生成における暗号文攻撃を行うことで $c1_i$ を得ることができる。従って、もし $c1_i$ と ks_i が独立でなければ、(4.3)から ks_i が入手できることになってしまう。

・ (cond 4.1b)

4.5.1 でも述べたように、盗聴者がメッセージの送信において選択平文攻撃および選択暗号文攻撃を行うことで、$(c1_i, ks_i)$の組をいくつも入手できることになる。一時鍵生成において、盗聴者がいくつも$(c1_i, ks_i)$の組を入手している状態でも、$c1_i$ から(ki1, ki2)が特定できてはならない。

・ (cond 4.1a)を満たすことの証明

ここで、任意の演算子"•"を考える。"•"を(4.3)の辺々に適用することで(4.4)を得る。

$$c1_i \bullet ks_i = ki1 * ss_i \bullet ki2 * ss_i \quad (4.4)$$

演算子"∗"は可換ではないため、演算子"•"を ss_i 同士に適用する事はできない。つまり、ks_i を、$c1_i$ を含む関数で表すことを考えると、(4.5)のように ss_i を含む関数となる。

$$ks_i = g\,(c1_i, ss_i\, ki1, ki2) \quad (4.5)$$

ss_i は通信のたびに異なるものを用いるので、$c1_i$ を知っているだけでは ks_i を導出することはできない。

例えば演算子"∗"が XOR 演算である場合を考える。(4.1), (4.2)は(4.6), (4.7)のように表すことができる。

$$c1_i = ki1 \oplus ss_i \quad (4.6)$$
$$ks_i = ki2 \oplus ss_i \quad (4.7)$$

(4.6), (4.7)の辺々の XOR をとる (つまり演算子"•" = ⊕)と(4.8)のようになる。

$$c1_i \oplus ks_i = ki1 \oplus ss_i \oplus ki2 \oplus ss_i \quad (4.8)$$

ここで、"⊕"は可換な演算子であるので、(4.8)は(4.9)のように表すことができる。

$$c1_i \oplus ks_i = ki1 \oplus ki2 \oplus ss_i \oplus ss_i \quad (4.9)$$

従って、(4.10)を得る。

$$c1_i \oplus ks_i = ki1 \oplus ki2 \quad (4.10)$$

ki1, ki2 は全ての通信において同一のものを用いるため、ki1 ⊕ ki2 は定数となる。Sending message における選択平文攻撃および選択暗号文攻撃から盗聴者が ks_i を知った場合、この定数の値がわかることになる。盗聴者はその定数の値をもちいて、次の通信の解読が実行できてしまう。以上から、演算子"∗"は可換であってはならない。

・ (cond 4.1b)を満たすことの証明

(4.11)を示すことにより(cond 4.1b)を満たすことを証明する。

$$H(ki1, ki2) \geq n \tag{4.11}$$

以下の証明で用いる用語の定義を行う。

1. $ki1_a(x)$: n ビットの系列 x に対して $ki1_a$ を適用する。$ki1_b$, $ki2_a$, $ki2_b$ に関しても同様。
2. ki1: 任意の n ビットの系列 x に対して $ki1_b(ki1_a(x))$を表す。ki2 も同様に定義される。
3. $R(ki1_a(x), ki1_a'(y))$: $ki1_a(x) = ki1_a'(y)$とした時に、x と y が満たす関係式。

任意の ki1′に対して、(4.12)を満たす ki2′, ss_i'が存在する。

$$R(ki1(ss_i), ki1'(ss_i')) = R(ki2(ss_i), ki2'(ss_i')) \tag{4.12}$$

(4.12)から(4.13), (4.14)が成り立つ。

$$ki1(ss_i) = ki1'(ss_i') = c1_i \tag{4.13}$$
$$ki2(ss_i) = ki2'(ss_i') = ks_i \tag{4.14}$$

盗聴者が sending message において選択平文攻撃および選択暗号文攻撃を行うことで、$(c1_i, ks_i)$の組をいくつも入手できたとしても、盗聴者は(ki1, ki2)の候補を(ki1′, ki2′)の候補の数までしか絞り込むことができない。そして、(ki1′, ki2′)の候補の数は少なくとも ‖ ki1 ‖ 個存在する。よって(4.11)が成り立つ。

以上の証明の具体的な例を挙げる。ここでは以下のように仮定する。

n: 3
$ki1_a$: (0, 0, 1)
$ki1_b$: (1, 1, 0)
$ki2_a$: (1, 0, 0)
$ki2_b$: (0, 1, 0)

$ki1_a'$, $ki1_b'$は任意であるが、ここでは次のように仮定する。

$ki1_a'$: (1, 0, 1)
$ki1_b'$: (0, 1, 1)

また、次のようにおく。

$$ss_i = (ss_0, ss_1, ss_2)$$
$$ss_i' = (ss_0', ss_1', ss_2')$$

以上の仮定の下で議論を進める。まず、$ki1(ss_i)$, $ki1'(ss_i')$は(4.15), (4.16)の様に表される。

$$ki1(ss_i) = ki1_b ki1_a(ss_i) = ki1_b(ss_0, \overline{ss_1}, \overline{ss_2}) = (\overline{ss_1}, \overline{ss_2}, ss_0) \quad (4.15)$$

$$ki1'(ss_i') = ki1_b' ki1_a'(ss_i') = ki1_b'(ss_0', \overline{ss_1'}, ss_2') = (\overline{ss_1'}, ss_2', ss_0') \quad (4.16)$$

従って、$R(ki1(ss_i), ki1'(ss_i'))$の関係とは具体的に言えば(4.17)の関係式になる。

$$\overline{ss_1} = \overline{ss_1'}$$
$$\overline{ss_2} = ss_2'$$
$$ss_0 = ss_0' \quad (4.17)$$

次に(4.12)を満たす ki2′について述べる。まず、$ki2(ss_i)$は(4.18)の様に表される。

$$ki2(ss_i) = ki2_b ki2_a(ss_i) = ki2_b(\overline{ss_0}, ss_1, ss_2) = (\overline{ss_0}, ss_2, ss_1) \quad (4.18)$$

ここで、(4.12)を満たすためには、$ki2'(ss_i')$は(4.19)を満たせばよい。

$$ki2'(ss_i') = (\overline{ss_0'}, \overline{ss_2'}, ss_1') \quad (4.19)$$

このような ki2′として(4.20)が挙げられる。

$$(ki2_a, ki2_b) = \{(1, 0, 1), (0, 1, 0)\}, \{(1, 0, 1), (1, 1, 1)\} \quad (4.20)$$

以上の例では、

$ki1_a'$: (1, 0, 1)
$ki1_b'$: (0, 1, 1)

のように仮定したが、任意の ki1′に対して、(4.12)を満たす ki2′が上述した方法と同様にして求めることができる。従って、盗聴者がメッセージの送信において選択平文攻撃および選択暗号文攻撃を行うことで、($c1_i$, ks_i)の組をいくつも入手できたとしても、盗聴者は(ki1, ki2)の候補を(ki1′, ki2′)の候補の数までしか絞り込むことができない。そして、(ki1′, ki2′)の候補の数は少なくとも ‖ ki1 ‖ 個存在する。よって(4.11)が成り立つ。

4.5.3. バーナム暗号との比較

本手法は第 3 章で述べたバーナム暗号と類似点がいくつかある。ここでは、バーナム暗号との比較について述べる。

図 4.7 において、一時鍵生成とメッセージの送信はそれぞれ、以下に示す攻撃に対する安全性を保っていれば、アルゴリズム全体の安全性は保証される。

- 一時鍵生成: 暗号文攻撃、既知平文攻撃
- メッセージの送信: 暗号文攻撃

ここで、メッセージの送信に着目すると、バーナム暗号と同様な構造になっていることが分かる。表 4.1 に本手法とバーナム暗号の比較を行った。

	Vernam 暗号	**本手法**
必要な鍵長	送信メッセージと同じビット長	固定
乱数の共有	必要あり	必要なし
安全性	完全秘匿	$\frac{1}{2}$H(K)

表 4.1. 本手法とバーナム暗号の比較

バーナム暗号は、鍵として送信するメッセージと同じ長さのビット長を保持しておかなければならないが、本手法では、鍵長は一定値である。また、バーナム暗号では、乱数自体が暗号化に用いる鍵となっているので、乱数生成器を共有する必要がある。本手法では、乱数生成器の共有をする必要がないため、共有することが難しい物理乱数なども乱数として用いる事ができる。また、安全性については、バーナム暗号は完全秘匿(鍵のエントロピーが失われない)が実現できているのに対し、提案手法では鍵のエントロピーが半分失われる。しかし、鍵長をその分長く持つことにより、この問題は解決される。

4.6. 提案手法による暗号回路

以下では本手法を用いる事で、小面積で暗号回路が実現できる事を示す。

4.6.1. 回路構成

Key Laundering および Sending Message を行うブロック図を図 4.1 に示す。

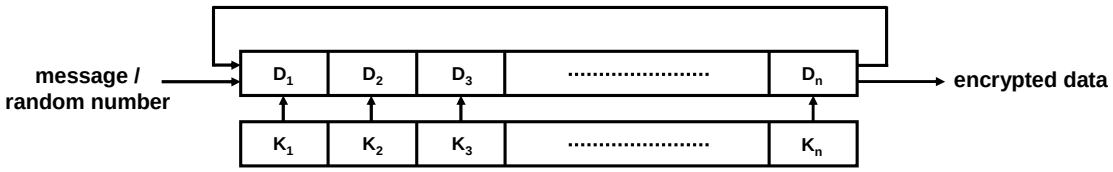

図 4.6　回路構成

1. Register D_1-D_n

一時鍵生成

- 乱数 r_i で初期化する。

Sending Message

- 送信メッセージ m_i で初期化する。

2. Register K_1-K_n

一時鍵生成

- $c1_i$ を生成する場合は、ki1 で初期化する。
- ks_i を生成する場合は、ki2 で初期化する。

Sending Message

- ks_i で初期化する。

4.6.2. 動作説明

1 回の通信についての動作説明をする。

Step1. (送信者側)

- Register D_1-D_n を乱数で初期化する。
- Register K_1-K_n を ki1 で初期化する。

Step2. (送信者側)

```
for (i=1; i≤n; i++) {
        if (Ki = = 1) {
                Di+1 ↔ Di ; //  ただし、Dn+1 は D1 を意味する。
        }
}
```

Step3. (送信者側)

・ Register D_1-D_n のデータを $c1_i$ として受信者に送信。

Step4. (受信者側)

・ 受信した $c1_i$ を Register D_1-D_n に格納する。

・ Register K_1-K_n を ki1 で初期化する。

Step5. (受信者側)

```
for (i=n; i≥1; i--) {
        if (Ki = = 1) {
                Di+1 ↔ Di ; //  ただし、Dn+1 は D1 を意味する。
        }
}
```

Step6. (受信者側)

・ RegisterK_1-K_n に ki2 を格納する。

Step7. (受信者側)

```
for (i=1; i≤n; i++) {
        if (Ki = = 1) {
                Di+1 ↔ Di ; //  ただし、Dn+1 は D1 を意味する。
        }
}
```

Step8. (受信者側)

・ Register D_1-D_n のデータを Register K_1-K_n に格納する。

Step9. (送信者側)

・ Register D_1-D_n を乱数 r_i で初期化する。

・ Register K_1-K_n を ki2 で初期化する。

Step10. (送信者側)

```
for (i=1; i≤n; i++) {
        if (Ki = = 1) {
                Di+1 ↔ Di ; //  ただし、Dn+1 は D1 を意味する。
        }
}
```

Step11. (送信者側)

・ RegisterD_1-D_n のデータを RegisterK_1-K_n に格納する。

Step12. (送信者側)

・ 送信メッセージを RegisterD_1-D_n に格納する。

Step13. (送信者側)

```
for (i=1; i≤n; i++) {
```

if (K_i = = 1) {

$D_{i+1} \leftrightarrow D_i$; // ただし、D_{n+1} は D_1 を意味する。

}

}

Step14. (送信者側)

・ Register D_1-D_n のデータを $c2_i$ として送信する。

Step15. (受信者側)

・ 受信した $c2_i$ を Register D_1-D_n に格納する。

Step16. (受信者側)

for (i=n; i≥1; i--) {

if (K_i = = 1) {

$D_{i+1} \leftrightarrow D_i$; // ただし、D_{n+1} は D_1 を意味する。

}

}

Step17. (受信者側)

・ Register D_1-D_n に格納されたデータが送信メッセージとなる。

4.6.3. 回路規模

上記の回路を構成するのに必要なゲート数を見積もる。ただし、秘密鍵長を 128bit とする(n=128)。

・ Register D_1-D_n : 4gates*128=512 gates

・ Register K_1-K_n : 4gates*128=512 gates

・ Total : 512+512=1,024 gates

以上から、FIFO 暗号は、約 1,000 ゲート程度で実現できることが分かる。 [6]に示すように、従来手法であれば 100,00 ゲート程度必要となる。乱数を生成する回路については、[12]のようにほぼ 1 つのレジスタ程度のゲート数で実現できるものがあり、これを用いることにより小型化が見込める。

第 5 章　結論

本論文では、センサノードに超小型集積回路を用いるセンサネットワークに適した、超小型暗号回路を実現するアルゴリズムを提案した。暗号アルゴリズムが安全性を保つために満たすべき条件を情報理論的に説明し、提案したアルゴリズムがその条件を満たしているかどうかを検証した。検証した結果を表 5.1 に示す。キーロンダリングに基づく手法においては、どの攻撃に対しても安全であることを証明し、実際に回路に実装した際にどれだけの回路規模になるかを見積もり、従来手法のものと比較を行い、十分小型化が見込める暗号手法である事を示した。

	シグマデルタ変調器を用いる暗号方式	**送信者が用いる鍵を受信者が決める方式**	**キーロンダリングに基づく方式**
暗号文攻撃	×	○	○
既知平文攻撃	×	×	○
選択平文攻撃	×	×	○
選択暗号文攻撃	×	×	○

表 5.1. 各手法の攻撃に対する安全性

參考文獻

[1] A. Sinha and A. Chandrakasan, "Dynamic power management in wireless sensor networks" IEEE Design and Test of Computers, pp. 62-74, Mar-Apr 2001.

[2] S. Elaine and A. Perrig, "Designing Secure Sensor Networks" Wireless Communications, IEEE, Volume: 11, Issue: 6, Dec 2004.

[3] ”Data Encryption Standard” Federal Information Processing Standards Publication 46, National Bureau of Standards, U.S. Department of Commerce, 1977

[4] J. Daemen and V. Rijmen, "AES Proposal Rijndael", AES Round1 Technical Evaluation CD-1: Documentation, National Institute of Standards and Technology, Aug 1998.

[5]. National Institute of Standards and Technology, http://www.nist.gov

[6]. S. Mangard, M. Aigner and S. Dominikus, "A Highly Regular and Scalable Hardware Architecuture", IEEE Transactions on Computers, vol. 52, no. 4, Apr 2003.

[7] A. Sinha and A. Chandrakasan, “Dynamic power management in wireless sensor networks” IEEE Design and Test of Computers, pp. 62-74, Mar-Apr 2001.

[8] A. Perrig, R. Szewczyk, J.D. Tygar, V.Wen, and D.E. Culler, “SPINS: Security protocols for sensor networks”, Wireless networks 8,521-534, 2002, Kluwer Academic Publications.

[9] H. Feistel “Cryptography and computer privacy “, 1973.

[10] A. Hodjat and Ingrid Verbauwhede, “Minimum Area Cost for a 30 to 70 Gbits/s AES Processor” Proceedings of the IEEE Computer Society Annual Symposium on VLSI Emerging Trends in VLSI Systems Design, pp. 83-88, Feb 2004.

[11] I. Verbauwhede, P. Schaumont, H. Kuo, “Design and Performance Testing of a 2.29-GB/s Rijndael Processor”, IEEE Journal of Solid-State Circuits, vol.38, no.3, pp. 569-572, Mar 2003.

[12] S. Yasuda, T. Tanamoto, H. Satake, and S. Fujita, “Novel Random Number Generator Using MOS Gate After Soft-Breakdown”, Extended Abstracts of the 2002 International Conference on Solid States Device and Materials, pp. 250-251, 2002.

[13]. H.Kuo and I.Verbauwhede, “Architectural optimization for a 1.82 Gbit/sec VLSI implementation of the AES Rijndael algorithm”, in Cryptographic Hardware and Embedded Systems (CHES) 2001.

[14]. Chih-Pin Su; Tsung-Fu Lin; Chih-Tsun Huang; Cheng-Wen Wu, “A Highly Efficient AES Cipher Chip”, Design Automation Conference, 2003. Proceedings of the ASP-DAC 2003. Asia and South Pacific , 21-24 Jan. 2003.

[15]. Lan Liu,Luke D. “Implementation of AES as a CMOS core”, Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference Volume: 1 , May 4-7, 2003

[16]. Sklavos N, Koufopavlou O, “Architectures and VLSI implementations of the AES-Proposal Rijndael”, Computers, IEEE Transactions on , Volume: 51 , Issue: 12 , Dec. 2002

[17]. Liang Deng, Hongyi Chen “A new VLSI implementation of the AES algorithm” Communications, Circuits and Systems and West Sino Expositions, IEEE 2002 International Conference on , Volume: 2 , 29 June-1 July 2002

[18]. Verbauwhede, I.; Schaumont, P.; Kuo, H.; “Design and performance testing of a 2.29-GB/s Rijndael processor” Solid-State Circuits, IEEE Journal of , Volume: 38 , Issue: 3 , March 2003

[19]. 総務省ホームページ　http://www.soumu.go.jp
[20]. 経済産業省ホームページ　http://www.meti.go.jp
[21]. 石田光一, 「ハイパスシグマデルタ変調を用いた低雑音アナログ・ディジタル変換回路」東京大学大学院工学系研究科電子工学専攻修士論文 2002
[22]. 佐藤証、森岡澄夫、宗藤誠治,"Rijndael の小型ハードウェア実装," マルチメディア、分散、協調とモバイル (DICOMO) シンポジウム, pp.663-668, Jun.2001.

本研究に関する発表

- 金子秀彦, 藤島実, “小型集積回路用簡易暗号回路の提案”, 第 8 回システム LSI ワークショップ (2004 年 11 月 29〜12 月 1 日) ポスターセッション, 北九州国際会議場
- Minoru FUJISHIMA and Hidehiko KANEKO, "A Tiny Streaming Cipher Circuit Based on Key Laundering", IEICE Transactions on Fundamentals, to be submitted.

謝辞

本研究を進めるにあたり、藤島実助教授には、ご多忙の中丁寧に指導して頂き、大変感謝しております。2 年間の間にくじけそうになった事が多々ありましたが、その度に支えて頂きました。また、人生観や生き方など、研究以外の面でも数々の御指導を頂きました。教官室での時間は確実に自分を成長させてくれたと思います。ありがとうございました。

研究室の先輩方や後輩達にも大変感謝しております。山本憲さんには、あらゆる分野でのアドバイスをして頂き、感謝しております。物事に徹底的に取り組む姿勢には、いつも隣の席で感心しておりました。今後の糧とさせて頂きます。2 年間共に研究をした志村君には私生活でもお世話になりました。その明るい性格には何度も助けられました。ありがとう。また、藤島研卒業生の肥後さんには、まだ研究の立ち上がり段階だったのにも関わらず、多くの助言を頂きました。感謝しております。

研究生活を取り仕切ってくださった助手の北澤さん、秘書の渋谷さん、鈴木さんには大変お世話になりました。ありがとうございます。1 年間だけでしたが、Lai さん、谷本君、金子暁彦君、王君、渡辺君、小林君、鬼塚君、山谷君とはとても楽しく過ごすことができました。そして、2 年間の修士生活を支えてくれた家族には心から感謝致します。

最後に、2 年間の修士生活を支えて頂いた皆様に、改めて感謝致します。