(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2015-82077
(P2015-82077A)
(43)公開日　平成27年4月27日(2015.4.27)

(51)Int.Cl.		F I			テーマコード（参考）
G09C 1/00	(2006.01)	G09C	1/00	610A	5J104
G06F 21/62	(2013.01)	G06F	21/24	166A	

審査請求 未請求　請求項の数 15　ＯＬ　（全 21 頁）

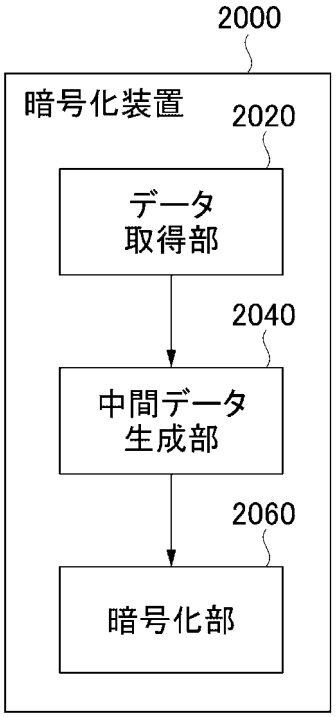
(21)出願番号　特願2013-221153 (P2013-221153)
(22)出願日　平成25年10月24日 (2013.10.24)

(71)出願人　800000068
学校法人東京電機大学
東京都足立区千住旭町５番
(74)代理人　100110928
弁理士　速水　進治
(72)発明者　鈴木　秀一
東京都足立区千住旭町５番　学校法人東京電機大学内

Ｆターム(参考)　5J104 AA16 AA32 AA41 DA04 EA04 EA18 JA03 NA02 NA37 PA14

(54)【発明の名称】暗号化装置、制御方法、及びプログラム

(57)【要約】
【課題】暗号化対象のデータを固定長のブロックに分割して各ブロックを独立に暗号化して暗号文を生成する場合において、暗号文の安全性を高める技術を提供する。
【解決手段】データ取得部２０２０は対象データを取得する。中間データ生成部２０４０は、対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なるブロックに含まれる部分データを用いて復元可能に変換することで、中間データを生成する。暗号化部２０６０は、中間データを所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各暗号化対象ブロックを独立に暗号化することで、中間データから暗号文を生成する。
【選択図】図１



【特許請求の範囲】
【請求項１】
対象データを取得するデータ取得手段と、
前記対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なる前記ブロックに含まれる部分データを用いて復元可能に変換することで、前記対象データから中間データを生成する中間データ生成手段と、
前記中間データを前記所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各前記暗号化対象ブロックを独立に暗号化して暗号文を生成する暗号文生成手段と、
を有する暗号化装置。
【請求項２】
前記中間データ生成手段は、前記対象データのサイズが前記所定サイズの倍数でない場合、前記対象データに対して、所定規則に従う並びを持つビット列を付加することで、前記対象データのサイズを前記所定サイズの倍数にする請求項１に記載の暗号化装置。
【請求項３】
前記中間データ生成手段は、前記対象データから得られる所定サイズの各前記ブロックについて、そのブロックに含まれる部分データを、そのブロックから所定個離れた別のブロックに含まれる部分データを用いて変換することで、前記対象データから前記中間データを生成する請求項１又は２に記載の暗号化装置。
【請求項４】
前記中間データ生成手段は、前記対象データに含まれる各ビットを、そのビットから所定個離れたビットを用いて変換することで、前記中間データを生成する請求項３に記載の暗号化装置。
【請求項５】
入力された整数とは異なる整数を出力する整数変換手段を有し、
前記中間データ生成手段は、前記対象データに含まれる各部分データについて、その部分データの位置を前記整数変換手段に入力し、その部分データの値を、前記整数変換手段から出力された整数が表す位置にある部分データの値を用いて変換する請求項１又は２に記載の暗号化装置。
【請求項６】
暗号化装置を制御するコンピュータによって実行される制御方法であって、
対象データを取得するデータ取得ステップと、
前記対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なる前記ブロックに含まれる部分データを用いて復元可能に変換することで、前記対象データから中間データを生成する中間データ生成ステップと、
前記中間データを前記所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各前記暗号化対象ブロックを独立に暗号化して暗号文を生成する暗号文生成ステップと、
を有する制御方法。
【請求項７】
前記中間データ生成ステップは、前記対象データのサイズが前記所定サイズの倍数でない場合、前記対象データに対して、所定規則に従う並びを持つビット列を付加することで、前記対象データのサイズを前記所定サイズの倍数にする請求項６に記載の制御方法。
【請求項８】
前記中間データ生成ステップは、前記対象データから得られる所定サイズの各前記ブロックについて、そのブロックに含まれる部分データを、そのブロックから所定個離れた別のブロックに含まれる部分データを用いて変換することで、前記対象データから前記中間データを生成する請求項６又は７に記載の制御方法。
【請求項９】
前記中間データ生成ステップは、前記対象データに含まれる各ビットを、そのビットか

ら所定個離れたビットを用いて変換することで、前記中間データを生成する請求項８に記載の制御方法。
【請求項１０】
入力された整数とは異なる整数を出力する整数変換ステップを有し、
前記中間データ生成ステップは、前記対象データに含まれる各部分データについて、その部分データの位置を入力として前記整数変換ステップを実行し、その部分データの値を、前記整数変換ステップによって出力された整数が表す位置にある部分データの値を用いて変換する請求項６又は７に記載の制御方法。
【請求項１１】
コンピュータに、暗号化装置として動作する機能を持たせるプログラムであって、前記コンピュータに、
対象データを取得するデータ取得機能と、
前記対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なる前記ブロックに含まれる部分データを用いて復元可能に変換することで、前記対象データから中間データを生成する中間データ生成機能と、
前記中間データを前記所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各前記暗号化対象ブロックを独立に暗号化して暗号文を生成する暗号文生成機能と、
を持たせるプログラム。
【請求項１２】
前記中間データ生成機能は、前記対象データのサイズが前記所定サイズの倍数でない場合、前記対象データに対して、所定規則に従う並びを持つビット列を付加することで、前記対象データのサイズを前記所定サイズの倍数にする請求項１１に記載のプログラム。
【請求項１３】
前記中間データ生成機能は、前記対象データから得られる所定サイズの各前記ブロックについて、そのブロックに含まれる部分データを、そのブロックから所定個離れた別のブロックに含まれる部分データを用いて変換することで、前記対象データから前記中間データを生成する請求項１１又は１２に記載のプログラム。
【請求項１４】
前記中間データ生成機能は、前記対象データに含まれる各ビットを、そのビットから所定個離れたビットを用いて変換することで、前記中間データを生成する請求項１３に記載のプログラム。
【請求項１５】
前記コンピュータに、入力された整数とは異なる整数を出力する整数変換機能を持たせ、
前記中間データ生成機能は、前記対象データに含まれる各部分データについて、その部分データの位置を入力として前記整数変換機能を実行し、その部分データの値を、前記整数変換機能によって出力された整数が表す位置にある部分データの値を用いて変換する請求項１１又は１２に記載のプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、暗号化装置、制御方法、及びプログラムに関する。
【背景技術】
【０００２】
ブロック暗号は、暗号化対象のデータを、固定長のデータ（以下、ブロック）単位に分割して処理することで生成される暗号である。ブロック暗号を生成する暗号方式には、DES(Data Encryption Standard) 暗号や AES(Advanced Encryption Standard) 暗号がある。
【０００３】

ブロック暗号を生成する際に、暗号化対象のデータをブロック単位で処理する処理方法は、暗号利用モード（Block Cipher Modes of Operation）と呼ばれる。暗号利用モードの１つに、ECB モード（Electric Code Book Mode）がある。ECB モードは、各ブロックをそれぞれ独立に暗号化する処理方法である。

【０００４】

暗号処理に関する先行技術を示す文献として、例えば特許文献１がある。特許文献１の暗号化装置は、暗号化対象の情報の価値に基づいて、暗号化の強度を決定する。

【先行技術文献】

【特許文献】

【０００５】

【特許文献１】特開２００６－９４２４１号公報

【発明の概要】

【発明が解決しようとする課題】

【０００６】

ECB モードには脆弱性がある。例えば、「生成された暗号文において、ブロック単位でのすり替えが可能である」という脆弱性である。以下、具体的に説明する。

【０００７】

暗号化対象のデータであるメッセージ M を３つのブロック M1、M2、M3 に分割し、各ブロックをそれぞれ暗号化するとする。ここで、ブロック Mi を暗号化することで生成された暗号化済みのブロックを、E(Mi) と表記する。E は、１つのブロックを暗号化する関数である。この暗号化により、メッセージ Mは、M1 を暗号化した E(M1)、M2を暗号化した E(M2)、M3 を暗号化した E(M3) という３つの暗号化済みブロックから成る暗号文 E(M) = {E(M1)、E(M2)、E(M3)}に暗号化される。

【０００８】

この暗号文を受け取った正規の宛先では、E(M1)、E(M2)、及びE(M3) をそれぞれ復号することで、E(M) からメッセージ M = {M1、M2、M3} を得る。

【０００９】

ここで、悪意ある攻撃者は、以下のようにして、正規の宛先に気づかれることなく、上記メッセージ M の一部を改ざんすることができてしまう。まず、攻撃者は、改ざん後の内容を表すブロック Mz を用意する。また、攻撃者は、上記関数 E(X) を何らかの方法で入手しておく。そして、攻撃者は、この関数を用いて、Mz から E(Mz) を生成する。

【００１０】

攻撃者は、暗号文 E(M) を生成した送信元から宛先に E(M) が送信された時に、その通信を盗聴することで、暗号文 E(M) を取得する。そして、E(M) に含まれる暗号化済みブロックのうち、改ざん対象のブロックを、E(Mz) に変更する。ここでは、E(M2) が改ざん対象であるとする。これにより、宛先に送られる暗号文は、E'(M') = {E(M1)、E(Mz)、E(M3)} になる。宛先は、E'(M') を復号することで、M' = {M1、Mz、M3} を得る。宛先は、暗号文の復号処理においてエラーが発生しないため、M2 が Mz に変更されたことを検知することができない。このように、攻撃者によるメッセージの改ざんが成功してしまう。

【００１１】

例えば、上述のメッセージ M がオンラインバンキングシステムにおける送金を表すメッセージであり、M1 が送金元、M2 が送金先、M3 が送金金額を表しているとする。この場合、攻撃者は、上述の例において Mz の内容を攻撃者の銀行口座等にしておくことで、送金を横取りすることができてしまう。

【００１２】

特許文献１は、上述した脆弱性及びこの脆弱性に対処する技術については開示していない。

【００１３】

本発明は、以上の課題に鑑みてなされたものである。本発明の目的は、暗号化対象のデータを固定長のブロックに分割して各ブロックを独立に暗号化して暗号文を生成する場合

において、暗号文の安全性を高める技術を提供することである。
【課題を解決するための手段】
【００１４】
本発明が提供する暗号化装置は、対象データを取得するデータ取得手段と、前記対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なる前記ブロックに含まれる部分データを用いて復元可能に変換することで、前記対象データから中間データを生成する中間データ生成手段と、前記中間データを前記所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各前記暗号化対象ブロックを独立に暗号化して暗号文を生成する暗号文生成手段と、を有する。
【００１５】
本発明が提供する制御方法は、本発明が提供する暗号化装置を制御するコンピュータによって実行される制御方法である。当該制御方法は、対象データを取得するデータ取得ステップと、前記対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なる前記ブロックに含まれる部分データを用いて復元可能に変換することで、前記対象データから中間データを生成する中間データ生成ステップと、前記中間データを前記所定サイズごとに分割して複数の暗号化対象ブロックを生成し、各前記暗号化対象ブロックを独立に暗号化して暗号文を生成する暗号文生成ステップと、を有する。
【００１６】
本発明が提供するプログラムは、コンピュータに、本発明が提供する各機能構成部が持つ機能を持たせる。したがって、当該プログラムは、コンピュータに、本発明が提供する暗号化装置として動作する機能を持たせる。
【発明の効果】
【００１７】
本発明によれば、暗号化対象のデータを固定長のブロックに分割して各ブロックを独立に暗号化して暗号文を生成する場合において、暗号文の安全性を高める技術が提供される。
【図面の簡単な説明】
【００１８】
【図１】実施形態１に係る暗号化装置を例示するブロック図である。
【図２】中間データ生成部が行う処理の具体例を示す図である。
【図３】暗号化装置のハードウエア構成を例示するブロック図である。
【図４】実施形態１の暗号化装置によって実行される処理の流れを例示するフローチャートである。
【図５】復号装置を例示するブロック図である。
【図６】実施形態２の暗号化装置が中間データを生成する処理を擬似コードとして例示する図である。
【図７】図６に示す処理で生成された中間データから対象データを復元する処理を擬似コードとして例示する図である。
【図８】実施形態３に係る暗号化装置を例示するブロック図である。
【図９】S-BOX が有する対応テーブルを例示する図である。
【発明を実施するための形態】
【００１９】
以下、本発明の実施の形態について、図面を用いて説明する。尚、すべての図面において、同様な構成要素には同様の符号を付し、適宜説明を省略する。
【００２０】
[実施形態１]
図１は、実施形態１に係る暗号化装置２０００を、その使用環境と共に示すブロック図である。図１において、矢印の流れは情報の流れを示している。また、図１において、各ブロックは、ハードウエア単位の構成ではなく、機能単位の構成を示している。

【００２１】
暗号化装置２０００は、暗号化対象のデータを取得し、取得したデータの暗号化を行う。以下、暗号化対象のデータを、対象データと表記する。
【００２２】
暗号化装置２０００は、データ取得部２０２０、中間データ生成部２０４０、及び暗号化部２０６０を有する。以下、それぞれについて説明する。
【００２３】
＜データ取得部２０２０＞
データ取得部２０２０は対象データを取得する。
【００２４】
＜中間データ生成部２０４０＞
中間データ生成部２０４０は、対象データから中間データを生成する。具体的には、中間データ生成部２０４０は、対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、そのブロックと異なるブロックに含まれる部分データを用いて復元可能に変換することで、中間データを生成する。ここで、部分データのサイズは、上記所定サイズの単位をビットとした場合、１ビット以上かつ所定サイズ以下の値である。また、あるブロックに含まれる部分データのサイズと、その部分データの変換に用いる他のブロックに含まれる部分データのサイズとは、同一であってもよいし異なっていてもよい。
【００２５】
図２は、中間データ生成部２０４０が行う処理の具体例を示す図である。図２において、対象データＸは、「111110000010」という１２ビットのデータである。ここで、図２において、所定サイズは４ビットである。そのため、対象データＸから得られる所定サイズのブロックは、「1111」、「1000」、及び「0010」という３つのブロックである。また、全ての部分データのサイズを１ビットとする。
【００２６】
なお、図２において、対象データＸを区切る点線が示されているものの、これは理解を容易にするための図示である。中間データ生成部２０４０は、実際に対象データＸの分割を行う必要はない。ただし、中間データ生成部２０４０は、実際に対象データＸを分割してもよい。
【００２７】
図２は、中間データ生成部２０４０が、各ブロックの末尾のビット（部分データ）が示す値を、そのビットの値と、次のブロックの先頭のビット（部分データ）が示す値との排他的論理和に変換する場合を例示している。１番目のブロックの末尾のビットが示す値である 1 は、２番目のブロックの先頭のビットが示す値である 1 との排他的論理和である 0 に変換される。同様に、２番目のブロックの末尾のビットが示す値、及び３番目のブロックの末尾のビットが示す値も変換される。なお、３番目のブロックは末尾のブロックであるため、中間データ生成部２０４０は、１番目のブロックを、３番目のブロックの次のブロックとして扱う。その結果、中間データ生成部２０４０は、「111010000011」という中間データ Y を生成する。
【００２８】
なお、中間データ生成部２０４０は、あるブロックに含まれる部分データを、複数の部分データを用いて変換してもよい。この際、複数の部分データは、同一のブロックに含まれていてもよいし、異なるブロックに含まれていてもよい。
【００２９】
また、中間データ生成部２０４０は、あるブロックに含まれる部分データを、他のブロックに含まれる部分データを加工したデータを用いて変換してもよい。例えば中間データ生成部２０４０は、あるブロックに含まれる部分データを、他のブロックに含まれる部分データのハッシュ値を用いて変換する。この場合、中間データ生成部２０４０は、ハッシュ値を算出する機能を有する。

【００３０】
＜暗号化部２０６０＞
暗号化部２０６０は、中間データを、中間データ生成部２０４０の説明で述べた所定サイズごとに分割して、複数の暗号化対象ブロックを生成する。暗号化部２０６０は、各暗号化対象ブロックを独立に暗号化することで、中間データから暗号文を生成する。ここで、「暗号化対象ブロックを独立に暗号化する」とは、ある暗号化対象ブロックを暗号化する際に、他の暗号化対象ブロックを利用しないことを意味する。これは、ブロック暗号の暗号利用モードのうち、ECB モードに相当する処理である。
【００３１】
なお、上記所定サイズとしては、任意の大きさを用いることができる。例えば所定サイズは、AES 暗号におけるブロック長と同じ 128 ビットである。
【００３２】
＜ハードウエア構成＞
暗号化装置２０００が有する各機能構成部は、例えば、個々に又は複数組み合わせられた状態で、少なくとも１つのハードウエア構成要素として実現される。その他にも例えば、各機能構成部は、少なくとも１つのソフトウエア構成要素として実現される。その他にも例えば、各機能構成部は、ハードウエア構成要素とソフトウエア構成要素の組み合わせにより実現される。
【００３３】
図３は、暗号化装置２０００のハードウエア構成を例示するブロック図である。図３において、暗号化装置２０００は、バス１０２０、プロセッサ１０４０、メモリ１０６０、及びストレージ１０８０を有する。
【００３４】
バス１０２０は、プロセッサ１０４０、メモリ１０６０、及びストレージ１０８０が、相互にデータを送受信するためのデータ伝送路である。プロセッサ１０４０は、例えば CPU (Central Processing Unit) や GPU (Graphics Processing Unit) などの演算処理装置である。メモリ１０６０は、例えば RAM (Random Access Memory) や ROM (Read Only Memory) などのメモリである。ストレージ１０８０は、例えばハードディスク、SSD (Solid State Drive) 、又はメモリカードなどの記憶装置である。また、ストレージ１０８０は、RAM や ROM 等のメモリであってもよい。
【００３５】
データ取得モジュール１２２０は、暗号化装置２０００に、データ取得部２０２０の機能を持たせるためのプログラムである。プロセッサ１０４０は、データ取得モジュール１２２０を実行することで、データ取得部２０２０の機能を実現する。
【００３６】
中間データ生成モジュール１２４０は、暗号化装置２０００に、中間データ生成部２０４０の機能を持たせるためのプログラムである。プロセッサ１０４０は、中間データ生成モジュール１２４０を実行することで、中間データ生成部２０４０の機能を実現する。
【００３７】
暗号化モジュール１２６０は、暗号化装置２０００に、暗号化部２０６０の機能を持たせるためのプログラムである。プロセッサ１０４０は、暗号化モジュール１２６０を実行することで、暗号化部２０６０の機能を実現する。
【００３８】
例えばプロセッサ１０４０は、上記各モジュールをメモリ１０６０上に読み出して実行する。ただし、プロセッサ１０４０は、上記各モジュールを、メモリ１０６０上に読み出さずに実行してもよい。
【００３９】
ストレージ１０８０は、上記各モジュールを格納する。
【００４０】
暗号化装置２０００のハードウエア構成は、図３に示した構成に限定されない。例えば

、各モジュールはメモリ１０６０に格納されてもよい。この場合、暗号化装置２０００は、ストレージ１０８０を備えていなくてもよい。

【００４１】

＜処理の流れ＞

図４は、実施形態１の暗号化装置２０００によって実行される処理の流れを例示するフローチャートである。ステップＳ１０２において、データ取得部２０２０は、対象データを取得する。ステップＳ１０４において、中間データ生成部２０４０は、対象データから中間データを生成する。ステップＳ１０６において、暗号化部２０６０は、中間データを複数の暗号化対象ブロックに分割する。ステップＳ１０８において、暗号化部２０６０は、各暗号化対象ブロックを暗号化して暗号文を生成する。

【００４２】

＜作用・効果＞

暗号化装置２０００は、対象データから中間データを生成し、その中間データから暗号文（ブロック暗号）を生成する。ここで、中間データ生成部２０４０は、対象データから得られる所定サイズの各ブロックについて、そのブロックに含まれる部分データを、そのブロックとは異なるブロックに含まれる部分データを用いて変換する。暗号化装置２０００は、このように各ブロックに含まれる部分データを別のブロックに含まれる部分データを用いて変換することで、ブロック間に関連性を持たせる。

【００４３】

ここで前述したように、各ブロックを独立して処理する ECB モードには、「生成された暗号文において、ブロック単位での改ざんが可能である」という脆弱性がある。本実施形態の暗号化装置２０００は、暗号化を行う前にブロック間に関連性を持たせた中間データを生成し、その中間データから暗号文を生成する。このようにすることで、暗号化装置２０００によって生成される暗号文に対し、ブロック単位ですり替えを行う攻撃を行うことが困難になるため、暗号文の安全性が高まる。以下に、暗号文に対してブロック単位ですり替えを行う攻撃を行うことが困難になる理由を説明する。

【００４４】

まず、本実施形態の暗号化装置２０００によれば、改ざんされたブロックを含む暗号文を復号することで中間データを得ても、その中間データから対象データを算出することができない。そのため、暗号化装置２０００によって生成された暗号文を復号する復号装置は、ブロックをすり替えられている暗号文を取得した場合、その暗号文を正常に復号することができない。その結果、復号装置は、暗号文に何らかの攻撃が行われたことを知ることができる。したがって、暗号化装置２０００によれば、復号装置において、ブロック単位ですり替えが行われた暗号文が、正しい暗号文として扱われてしまうことを防ぐことができる。よって、暗号文に対してブロック単位ですり替えを行う攻撃を行うことが困難になる。

【００４５】

また、本実施形態において「対象データを中間データに変換してから暗号化を行う」という方法で生成された暗号文の安全性が、対象データを直接暗号化する ECB モードの暗号化によって生成される暗号文の安全性よりも低くすることはない。これは、以下のように数学的に証明できる。

【００４６】

証明する命題は、以下の数式（１）に示す命題である。この命題は、「ECB モードの処理で生成された暗号文は安全である」が成り立てば、「暗号化装置２０００によって生成される暗号文は安全である」が成り立つことを示している。この数式において、E(K, M) はECB モードの処理で対象データを直接暗号化することで得られる暗号文を表し、E(K, f(M)) は本実施形態の暗号化装置２０００によって生成される暗号文を表す。E は ECB モードで行う暗号化を表す関数、K は暗号化に用いる鍵、M は対象データである。また、関数 f は、中間データ生成部２０４０が行う処理を表す。つまり、f(M) は、中間データを表す。

【数１】

$$E(K,M)\text{は安全である}\Rightarrow E(K,f(M))\text{は安全である}\cdots(1)$$

【００４７】

ここで、上記命題を証明するために、この命題の対偶を証明する。この対偶は、下記数式（２）で表される。

【数２】

$$E(K,f(M))\text{は安全でない}\Rightarrow E(K,M)\text{は安全でない}\cdots(2)$$

【００４８】

まず、E(K, f(M)) を解読可能なアルゴリズム A が存在すると仮定する。したがって、数式（３）に示すように、このアルゴリズム A に E(K, f(M)) を与えると M が算出される。すると、以下の数式（４）が成り立つこととなる。

【数３】

$$A(E(K,M))=M\cdots(3)$$

$$\begin{aligned} f(A(E(K,M))&=f(A(E(K,f(f^{-1}(M))))) \\ &=f(f^{-1}(M)) \\ &=M\cdots(4) \end{aligned}$$

【００４９】

上記（４）により、E(K, f(M)) を解読可能なアルゴリズムがあれば、E(K, M) を解読可能なアルゴリズムが存在することになる。このことから、数式（２）で表される対偶は真である。よって、数式（１）で表される命題も真である。

【００５０】

以上のように、本実施形態の暗号化装置２０００によって生成される暗号文の安全性が、一般の ECB モードの暗号化によって生成される暗号文の安全性より低くなることはない。

【００５１】

＜＜IV(Initial Value) を用いて生成されるブロック暗号の脆弱性＞＞

暗号化に IV と呼ばれる値を用いる暗号利用モード（以下、IV モード）が広く利用されている。ここで、IV モードには、CBC モード（Cipher Block Chaining Mode）、CTR モード（Counter Mode）、CFB モード（Cipher Feedback Mode）、及び OFB モード（Output Feedback Mode）などがある。

【００５２】

本発明者は、IV モードで生成されるブロック暗号に脆弱性があり、この脆弱性を無くすことは困難であることを発見した。そのため本発明者は、この脆弱性を持たない ECB モードで生成されるブロック暗号を、本発明によってさらに安全なものにすることで、ECB モードを利用しやすいものにすることを考えた。以下、IV モードで生成されるブロック暗号の脆弱性について説明する。

【００５３】

まず、IV モードの利用方法について簡単に説明する。ここで、IV モードで生成される

暗号文を I(K, M) と表記する。K は暗号化処理及び復号処理の双方で利用される秘密鍵であり、M は暗号化対象のメッセージである。暗号化装置では、I(K, M) が、K 及び IV を利用して M を暗号化することで、暗号文 I(K, M) を生成する。

【００５４】

I(K, M) を復号する復号装置では、K 及び IV が必要である。正規の復号装置は、事前に暗号を生成する暗号化装置とやりとりをしておくことで、秘密鍵 K を入手している。また、復号装置は、I(K, M) と共に IV を取得する。例えば復号装置は、I(K, M) と IV とが連結したデータを、暗号化装置から取得する。このように、秘密鍵 K は第三者が知り得ない状態でやりとりされるのに対し、IV は第三者でも知り得る状態でやりとりされる。

【００５５】

本発明者が発見した IV モードで生成されたブロック暗号に存在する脆弱性は、「暗号化装置や暗号化装置で用いられる暗号プログラムを提供する者が、装置で行われる処理に不正な処理を含めることで、暗号化装置のユーザや復号装置のユーザに気づかれずに秘密鍵 K を取得できてしまう」という脆弱性である。このように、「暗号化装置や暗号化装置で用いられる暗号プログラムを提供する者が、装置で行われる処理に不正な処理を含めることで、ユーザに気づかれないように行う攻撃」を、インサイダー攻撃と呼ぶ。

【００５６】

まず、インサイダー攻撃のモデルを、以下の５つの要件を満たすものとして定義する。このモデルは、以降で説明する各インサイダー攻撃に共通のモデルである。第１の要件は、攻撃者が、暗号化装置で用いられる暗号プログラムの納入業者であることである。

【００５７】

第２の要件は、暗号化装置のユーザが攻撃の対象であることである。そのため、例えば攻撃の内容は、この暗号化装置を用いて生成された暗号文を解読することで、ユーザが正規の宛先にのみ伝達したいと考えている情報を解読してしまうというものである。

【００５８】

第３の要件は、攻撃者が、暗号化装置から復号装置に送られる暗号文を取得できることである。一般に、暗号文はネットワークを介してやりとりされることが多い。例えば、メールの送信元が暗号化装置である場合、暗号化装置は、メールの本文や添付ファイルを暗号化して暗号文を生成し、この暗号文を復号装置へ送信する。そのため、攻撃者は、暗号化装置と復号装置との間でやりとりされる通信を盗聴することで、暗号文を取得できる。このような盗聴の方法は既知の技術であるため、説明を省略する。

【００５９】

第４の要件は、暗号化装置が、暗号文を生成するために用いた暗号鍵を外部に出力しないということである。例えば暗号化装置がネットワークを介して暗号鍵を外部に出力してしまうとする。こうすると暗号文は、ここで説明するインサイダー攻撃とは無関係に、容易に解読されてしまう。そこで、インサイダー攻撃のモデルでは、このように暗号鍵を外部に出力するような攻撃を行えない状況を仮定する。

【００６０】

第５の要件は、暗号化装置のユーザが、暗号化装置によって出力された暗号文を復号できることである。この要件により、ユーザは、暗号化装置に暗号化させたデータ（本実施形態における対象データ）が、暗号化装置によって改ざんされていないことを確認できる。

【００６１】

以下、上記要件が満たされているという前提の下で、本発明者が見出したインサイダー攻撃について説明する。このインサイダー攻撃は、「攻撃者が、暗号化装置に提供する暗号プログラムのうち、IV を生成するプログラムを悪意あるプログラムにすることで、秘密鍵 K を取得する」というものである。一般に、IV は乱数として生成されることが多い。これに対し、攻撃者は、提供する暗号プログラムにおいて、IV の生成方法を、「秘密鍵 K を暗号化した値 e(K) を算出し、この e(K) の値を IV として用いる」という方法

にする。ここで、e(K) は、攻撃者のみが知る復号用関数 d によって復号できる。

【００６２】

暗号化装置では、値が e(K) である IV を用いて暗号化を行い、暗号文 I(K, M)を生成する。そして、暗号化装置は、I(K, M) と IV を、復号装置へ送信する。攻撃者は、暗号化装置と復号装置との間の通信を盗聴することで、I(K,M) と IV を取得する。攻撃者は、前述した復号用関数 d を用いて IV を復号することで、秘密鍵 K を取得できる。こうすると、攻撃者は、取得した秘密鍵 K を用いて I(K,M) を復号し、メッセージ M を取得することができてしまう。

【００６３】

暗号化装置や復号装置のユーザが上述の攻撃に気づくことは難しい。一般の暗号プログラムの場合は IV に乱数が設定される一方、上述の悪意あるプログラムの場合は IV に暗号文 e(K) が設定される。しかし、ユーザがある値を見たときに、それが乱数なのか暗号文なのかを見分けることは困難である。そのため、暗号化装置や復号装置のユーザが IV の値 e(K) を見ても、この値が秘密鍵 K を暗号化することで生成されたものなのか、一般的な乱数なのかを見分けることは困難である。

【００６４】

以上のように、暗号化装置に対して暗号プログラムを提供する者は、暗号化装置や復号装置のユーザに気づかれない方法で、IV モードで生成されたブロック暗号に用いられる暗号鍵を取得できてしまう。その結果、暗号文を復号することができてしまう。

【００６５】

＜暗号化装置２０００の詳細＞

次に、本実施形態の暗号化装置２０００について更に詳細に説明する。

【００６６】

＜＜データ取得部２０２０の詳細＞＞

データ取得部２０２０が対象データを取得する方法は様々である。データ取得部２０２０は、暗号化装置２０００の内部又は外部から対象データを取得する。例えば暗号化装置２０００は、種々のアプリケーションによって生成されたデータ（例：メーラによって生成されたメール）を対象データとして取得する。この場合、例えばデータ取得部２０２０は、暗号化装置２０００の内部又は外部で動作するアプリケーションから対象データを取得する。また、暗号化装置２０００は、対象データが格納されている格納部から対象データを取得してもよい。また、暗号化装置２０００は、外部から手動又は自動で入力される対象データを取得してもよい。

【００６７】

＜＜暗号化部２０６０の詳細＞＞

暗号化部２０６０が各暗号化対象ブロックを暗号化する暗号アルゴリズムは、各暗号化対象ブロックを独立に暗号化するものであればよい。ブロック暗号において各ブロックを独立に暗号化する ECB モードでは、種々の暗号アルゴリズムが用いられている。暗号化部２０６０は、この種々の暗号アルゴリズムを利用できる。

【００６８】

暗号化部２０６０は、暗号化装置２０００の内部又は外部から取得した暗号鍵を用いて、各暗号化対象ブロックを暗号化する。なお、暗号化装置と復号装置との間で暗号鍵をやりとりする方法は既知の技術であるため、説明を省略する。

【００６９】

＜＜中間データ生成部２０４０の詳細＞＞

中間データ生成部２０４０は、「値の並びが所定規則に従うビット列を対象データに付加することで、対象データのサイズを所定サイズの倍数にする」という処理を行ってもよい。前述したように、中間データ生成部２０４０は、対象データから得られる所定サイズの各ブロックについて処理を行うことで、中間データを生成する。ここで、対象データのサイズが所定サイズの倍数でない場合、得られるブロックの中には、所定サイズに満たないブロックが含まれる。これに対し、中間データ生成部２０４０が対象データのサイズを

所定サイズの倍数にすれば、全てのブロックのサイズが所定サイズになる。

【００７０】

対象データに対してビット列を付加する処理の具体例を説明する。まず中間データ生成部２０４０は、対象データのサイズを所定サイズで割った余りを算出する。そして、中間データ生成部２０４０は、この余りの値で表されるサイズを持つビット列であり、かつ値の並びが所定規則に従うビット列を、その暗号化対象ブロックに付加する。

【００７１】

上述した、値の並びが所定規則に従うビット列は様々である。例えばこのビット列は、全てのビットの値が０であるビット列である。また例えば、このビット列は、全てのビットの値が１であるビット列である。その他にも例えば、このビット列は、「01」や「110」など、特定のビット列の繰り返しで表されるビット列である。以下、この特定のビット列を、繰り返し単位ビット列と表記する。

【００７２】

ここで、中間データ生成部２０４０が対象データに付加するビット列のサイズが、繰り返し単位ビット列のサイズの倍数でない場合も考えられる。例えば、中間データ生成部２０４０が付加するビット列の長さが１０ビットである場合に、繰り返し単位ビット列が「110」であるとする。この場合、例えば中間データ生成部２０４０は、「110+110+110+11」のように、繰り返し単位ビット列をできる限り繰り返すように、付加するビット列を生成する。

【００７３】

ただし、中間データ生成部２０４０は、繰り返し単位ビット列のサイズが、対象データに付加するビット列のサイズの約数となるようにしてもよい。例えば中間データ生成部２０４０は、サイズの異なる繰り返し単位ビット列を複数用意しておき、その中から、サイズが対象データに付加するビット列の約数である繰り返し単位ビット列を選択して用いる。

【００７４】

中間データ生成部２０４０が用いる繰り返し単位ビット列は、予め暗号化装置２０００の内部に格納されていてもよいし、外部から与えられてもよい。繰り返し単位ビット列が外部から与えられる場合、例えばデータ取得部２０２０が、対象データと共に繰り返し単位ビット列を取得してもよい。

【００７５】

以上のように所定規則に従うビット列を付加することによって対象データのサイズを所定サイズの倍数にすることには、インサイダー攻撃を防ぐことができるという作用効果がある。以下、具体的に説明する。

【００７６】

暗号化処理と復号処理とで共通の秘密鍵が用いられると仮定する。この場合、例えば、「暗号化部２０６０によって生成される暗号文の一部が秘密鍵の一部を表すように、対象データに対して悪意あるビット列を付加する」というインサイダー攻撃を行うことが考えられる。具体的には、次の流れで攻撃を行う。１）対象データのサイズが所定サイズの倍数になるまで、対象データの末尾に、ランダムに生成したビットを付加する。２）ビット列を付加した後の対象データに対して中間データ生成部２０４０と暗号化部２０６０による処理を実行することで、暗号文を生成する。３）生成された暗号文の末尾ｎビットと、秘密鍵の末尾ｎビットとを比較する。ここで、例えばｎは１である。４）生成された暗号文の末尾ｎビットと、秘密鍵の末尾ｎビットとが同一になるまで、１）から３）の処理を繰り返す。５）生成された暗号文の末尾ｎビットと、秘密鍵の末尾ｎビットとが同一になった場合、この暗号文を暗号化部２０６０による出力とする。

【００７７】

以上の流れで処理を行うことにより、暗号化部２０６０によって生成された暗号文の末尾ｎビットが、秘密鍵の末尾ｎビットを表すようになる。この暗号文を、暗号文Ｘと表記する。このインサイダー攻撃を行う攻撃者は、さらに次の流れで攻撃を行うことで、秘

密鍵全体を取得できる。まず、暗号化装置２０００と復号装置との通信を盗聴することで、暗号文Ｘを取得する。この攻撃者は、暗号文Ｘの末尾ｎビットを抽出することで、秘密鍵の末尾ｎビットを知る。そして、この攻撃者は、秘密鍵の残りの部分をブルートフォース等の方法で見つけることで、秘密鍵全体を取得する。

【００７８】

ここで一般に、暗号鍵のサイズは大きいため、ブルートフォースで暗号鍵を見つけようとすると膨大な時間がかかり、現実的な時間で見つけることは難しい。しかし、上記の方法で秘密鍵の一部を知ることができれば、ブルートフォースで秘密鍵を見つけるために要する時間が指数的に短縮される。その結果、攻撃者にとって、ブルートフォースで秘密鍵を見つけることが容易になってしまう。

【００７９】

そこで、中間データ生成部２０４０は、値の並びが所定規則に従うビット列を対象データに付加することで対象データのサイズを所定サイズの倍数にする。この所定サイズは、暗号化部２０６０が暗号化の処理単位とする暗号化対象ブロックのサイズである。これにより、上述したインサイダー攻撃を行うことが難しくなる。その理由は、以下に示す通りである。

【００８０】

まず、上述の方法でインサイダー攻撃を行う場合に、対象データに付加する悪意あるビット列が、所定規則に従っていないとする。この場合、暗号文から対象データを復元する復号装置は、対象データに付加されているビット列が所定規則に従っているか否かのチェックを行うことで、このビット列に悪意があるか否かをチェックすることができる。また、対象データに付加する悪意あるビット列として、所定規則に従うビット列であり、かつ攻撃者の意図に合うビット列（例：秘密鍵の一部を表すビット列）を生成することは確率的に困難である。よって、例えば上記所定規則を公開するように暗号化装置２０００の製造業者に義務づけておき、復号装置における上記のチェックを可能にすれば、ユーザに気づかれないようにインサイダー攻撃が行われることを防ぐことができる。

【００８１】

さらに、「中間データ生成部２０４０が、値が所定規則に従うビット列を付加して対象データのサイズを所定サイズの倍数にした場合において、暗号化装置２０００によって生成される暗号文は、暗号鍵の情報を暗号文に含めようとするインサイダー攻撃に対して安全である」という命題を証明する。証明する命題を、命題２とおく。ただし、１）中間データ生成部２０４０が対象データと同じサイズの中間データを生成すること、及び２）暗号化部２０６０が中間データと同じサイズの暗号文を生成すること、を前提条件とする。対象データに対してビット列を付加することで、対象データのサイズは、所定サイズの倍数になっている。そのため、その後、中間データ生成部２０４０や暗号化部２０６０は、対象データのサイズを変更する必要がない。

【００８２】

上記前提条件は、数式（５）に表すことができる。ここで、Length(X) は、データＸのサイズを表す。また、M0 は、値が所定規則に従うビット列を付加することでサイズが所定サイズの倍数となった対象データである。f(M0) は、M0 から生成される中間データである。また、数式（１）と同様に、E は ECB モードで行う暗号化を表す関数であり、K は暗号鍵である。

【数４】

$$Length(M_0) = Length(f(M_0)) = Length(E(K, f(M_0))) \cdots (5)$$

【００８３】

以下、上記命題を証明する。まず、暗号化装置２０００によって暗号文 C = E(K, f(M0)) が生成されたとする。そして、暗号文 C は、インサイダー攻撃に対して安全でないと仮定する。この仮定により、前述したインサイダー攻撃のモデルにおける第５の要件から

、暗号化装置２０００のユーザは、暗号文 C を復号できることになる。
【００８４】
インサイダー攻撃は、暗号文に対して、暗号鍵 K の情報の一部 K' を加えるとする。ここで、Length(K') > 0 であることは明らかである。そのため、この不等式及び数式（５）より、Length(K') + Length(M0) > Length(C) を得る。よって、暗号文 C のサイズは、K' のサイズと M0 のサイズの和より小さくなる。これは、暗号文 C を生成する際に、データのサイズを圧縮する非自明な圧縮処理が行われることを意味する。
【００８５】
ここで、暗号化装置２０００のユーザは、このような非自明な圧縮処理が行われた暗号文 C を復号することができない。よって、前述した仮定と矛盾する。このことから、命題が真であることが証明される。
【００８６】
＜＜暗号文の復号方法＞＞
例えば対象データが宛先へ送るメッセージである場合、この宛先は、暗号化装置２０００がこの対象データから生成した暗号文を復号できる必要がある。図５は、暗号化装置２０００によって生成された暗号文を復号する復号装置３０００を示す図である。以下、この復号装置３０００について説明する。
【００８７】
復号装置３０００は、暗号文取得部３０２０、暗号文復号部３０４０、及び対象データ復元部３０６０を有する。暗号文取得部３０２０は、暗号化装置２０００によって生成された暗号文を取得する。暗号文復号部３０４０は、この暗号文を復号して、中間データを算出する。対象データ復元部３０６０は、この中間データから対象データを復元する。
【００８８】
暗号文取得部３０２０が暗号文を取得する方法は様々である。例えば暗号化装置２０００がネットワークを介して復号装置３０００に対して暗号文を送信する場合、暗号文取得部３０２０は、この暗号文を受信することで、暗号文を取得する。また例えば、暗号化装置２０００が復号装置３０００からアクセス可能な記憶装置に暗号文を格納する場合、暗号文取得部３０２０は、この記憶装置から暗号文を読み出すことで、暗号文を取得する。
【００８９】
暗号文復号部３０４０は、暗号文を所定サイズごとに分割して復号対象ブロックを生成し、各復号対象ブロックを復号する。各復号対象ブロックを復号する方法は、暗号化部２０６０が各暗号化対象ブロックを暗号化した方法に依存する。例えば、暗号化部２０６０と復号装置との双方で共通鍵を利用する場合、暗号文復号部３０４０は、この共通鍵を予め保持しておき、この暗号鍵を用いて各復号対象ブロックを復号する。暗号化部２０６０によって利用される暗号アルゴリズムは、例えば一般の ECB モードの処理で実行される暗号アルゴリズムである。そのため、この暗号アルゴリズムで暗号化された暗号文を復号する復号アルゴリズムも既知である。
【００９０】
ここで、復号装置３０００が複数の暗号化装置２０００それぞれから暗号文を取得する場合、暗号文復号部３０４０は、各暗号化装置２０００の ID に対応づけて、その暗号化装置２０００から取得する暗号文から中間データを算出するための情報を保持しておく。例えば上述のように共通鍵を用いる方法の場合、復号装置３０００は、暗号化装置２０００の ID に対応づけて共通鍵を保持しておく。
【００９１】
対象データ復元部３０６０は、復号対象ブロックを結合した中間データを得る。この中間データは、中間データ生成部２０４０によって生成された中間データと同一である。対象データ復元部３０６０は、この中間データから、データ取得部２０２０によって取得された対象データを算出する。この算出方法は、中間データ生成部２０４０が中間データを生成した方法に依存する。例えば復号装置３０００が複数の暗号化装置２０００それぞれから暗号文を取得する場合、対象データ復元部３０６０は、暗号化装置２０００の ID に

対応づけて、その暗号化装置２０００の中間データ生成部２０４０によって生成された中間データから対象データを算出する方法を予め保持しておく。

【００９２】

[実施形態２]

実施形態２に係る暗号化装置２０００は、実施形態１に係る暗号化装置２０００と同様に、例えば図１で表される。実施形態２の暗号化装置２０００は、以下で説明する点を除き、実施形態１に係る暗号化装置２０００と同様の機能を有する。

【００９３】

実施形態２の中間データ生成部２０４０は、対象データから得られる各ブロックについて、そのブロックに含まれる部分データを、そのブロックから所定個離れた別のブロックに含まれる部分データを用いて変換する。これにより、中間データ生成部２０４０は、対象データから中間データを生成する。なお、あるブロックＡから所定個離れたブロックＢは、ブロックＡより所定個後ろのブロックであってもよいし、ブロックＡより所定個前のブロックであってもよい。例えば、上記所定個が３個である場合、中間データ生成部２０４０は、i 番目のブロックに含まれる部分データの変換に、(i+3) 番目のブロックに含まれる部分データ又は (i-3) 番目のブロックに含まれる部分データを用いる。

【００９４】

例えば中間データ生成部２０４０は、上記所定個を n 個、所定サイズを b とした場合、対象データに含まれる全てのビットそれぞれを、そのビットから (n*b) 個離れたビットを用いて変換する。

【００９５】

図６は、実施形態２の暗号化装置２０００が中間データを生成する処理を擬似コードとして例示する図である。手続 f は、対象データから中間データを生成する手続である。図６において、a は対象データ、n は対象データのサイズ、m は繰り返し数を表す。繰り返し数については後述する。なお、図６の例において、対象データに含まれる各部分データのサイズは１バイトであるとする。

【００９６】

図６において、対象データは、１バイトの要素をｎ個持つ配列として表されている。そして、手続 f は、配列の i 番目の要素である a[i] を、a[i] 自身と、その１つ前の要素である a[i-1] との和として算出する。

【００９７】

繰り返し数 m は、手続 f において、「配列 a に含まれる全ての要素それぞれを変換する」という一連の処理を繰り返す回数である。

【００９８】

なお、図６に示す疑似コードで生成された中間データは、図７の疑似コードで示される手続 finv によって、対象データを復元することができる。つまり、中間データ生成部２０４０が図６に示す方法で中間データを生成する場合、復号装置３０００の対象データ復元部３０６０は、図７に示す方法で対象データを復元する。なお、図７において、a は中間データ、n は中間データのサイズ、m は繰り返し数を表す。finv が取得する m の値は、中間データ a を生成する際に手続 f に与えた繰り返し数と同じである必要がある。

【００９９】

なお、前述したように、手続 f は、a[i] を、a[i] と a[i-1] との和として算出する。しかし、手続 f が中間データを生成するための演算は、和を算出する演算に限定されない。例えば、手続 f は、a[i] を、a[i] と a[i-1] との排他的論理和として算出する。ただし、finv で行う演算は、f で行う演算に応じて変更する必要がある。

【０１００】

ここで、f と finv において、繰り返し数 m は引数として与えられているため、繰り返し数 m は可変な値である。しかし、繰り返し数 m は固定されている方が好ましい。これは、繰り返し数 m が任意の数でよい場合、インサイダー攻撃が行われる恐れがあるためである。以下、具体的に説明する。

【０１０１】
まず、暗号化処理と復号処理において、共通の秘密鍵が用いられると仮定する。そして、このインサイダー攻撃は、「暗号化部２０６０によって出力される暗号文の末尾 n ビットが、秘密鍵の末尾 n ビットを表すようにする」というものである。具体的には次のような流れで攻撃が行われる。１）m に１を設定する。２） f を実行して中間データを生成する。３）この中間データから暗号文を生成する。４）生成された暗号文の末尾 n ビットと秘密鍵の末尾 n ビットを比較する。５）生成された暗号文の末尾 n ビットと秘密鍵の末尾 n ビットとが同一でない場合、m に１を加算する。６）生成された暗号文の末尾 n ビットと秘密鍵の末尾 n ビットとが同一になるまで、２）から５）を繰り返す。７）生成された暗号文の末尾 n ビットと、秘密鍵の末尾 n ビットとが同一になった場合、この暗号文を暗号化部２０６０による出力とする。
【０１０２】
このようにして暗号文の一部に秘密鍵の一部が含まれるようにした攻撃者は、実施形態１で説明した方法で、秘密鍵全体を取得することができてしまう。なお、このように繰り返し数 m の値が可変である場合、例えば復号装置３０００は、暗号文と共に繰り返し数 m を取得する。
【０１０３】
このインサイダー攻撃は、繰り返し数 m を予め固定しておけば防ぐことができる。例えば、暗号化装置２０００の製造業者に、繰り返し数 m の公表を義務づけておく。このとき、暗号化装置２０００が公表した繰り返し数 m を無視して上述のインサイダー攻撃を行ったとする。この場合、復号装置３０００は、公表された繰り返し数 m で finv を実行しても、対象データを復元することができない。そのため、復号装置３０００は、暗号化装置２０００において上述のインサイダー攻撃が行われていることを検知することができる。
【０１０４】
＜作用・効果＞
本実施形態によれば、中間データ生成部２０４０において、各ブロックに含まれる部分データの変換に用いるブロックを、そのブロックから所定個離れたブロックにする。そのため、変換に用いるブロックの位置を乱数等によって決めるアルゴリズムと比較し、アルゴリズムの構築が容易である。よって、中間データ生成部２０４０の設計及び実装が容易である。
【０１０５】
[実施形態３]
図８は、実施形態３に係る暗号化装置２０００を示すブロック図である。ここで、図８において、矢印の流れは情報の流れを示している。また、図８において、各ブロックは、ハードウエア単位の構成ではなく、機能単位の構成を示している。なお、実施形態３の暗号化装置２０００は、以下で説明する点を除き、実施形態１に係る暗号化装置２０００と同様の機能を有する。
【０１０６】
実施形態３の暗号化装置２０００は、整数変換部２０８０を有する。整数変換部２０８０は、入力された整数とは異なる整数を出力する。
【０１０７】
実施形態３の中間データ生成部２０４０は、次のように動作する。まず、中間データ生成部２０４０は、対象データに含まれる各部分データについて、対象データにおけるその部分データの位置を整数変換部２０８０に入力する。そして、中間データ生成部２０４０は、その部分データを、整数変換部２０８０によって出力された整数によって表される位置にある部分データを用いて変換する。
【０１０８】
整数変換部２０８０は、例えば S-BOX を用いて実現される。S-BOX は、入力する整数と出力する整数とを対応付ける対応テーブルを有する。S-BOX は、この対応テーブルを参

照して、整数変換部２０８０に入力された整数に対応する整数を出力する。
【０１０９】
図９は、S-BOX が有する対応テーブル２００を例示する図である。対応テーブル２００は、入力２０２及び出力２０４を有する。S-BOX は、この対応テーブルの中から、入力２０２が S-BOX に入力された整数を示すレコードを割り出し、割り出したレコードの出力２０４に示されている整数を出力する。例えば、整数変換部２０８０に対して３が入力されたとする。対応テーブル２００において、入力２０２が３を示すレコードは、出力２０４として３０を示している。したがって、整数変換部２０８０は、３０を出力する。よって、中間データ生成部２０４０は、対象データの３番目の部分データの値を、対象データの３０番目の部分データの値を用いて変換を行う。
【０１１０】
整数変換部２０８０は、S-BOX を有する形態に限定されない。例えば整数変換部２０８０は、乱数を生成する乱数生成部を有する。整数変換部２０８０は、乱数生成部を用いて、整数変換部２０８０に対して入力された整数とは異なる整数であり、かつ対象データのサイズ以下の値を表す整数を、ランダムに出力する。また例えば、整数変換部２０８０は、入力された値のハッシュ値を算出するハッシュ値生成部を有していてもよい。
【０１１１】
ここで、整数変換部２０８０によって出力される整数の範囲の大きさは、対象データに含まれる上記部分データの数と同じである必要がある。例えば、部分データの数が n である場合、整数変換部２０８０によって出力される整数の範囲は、「0 以上 (n-1) 以下」や「1 以上 n 以下」などである必要がある。
【０１１２】
そこで、例えば整数変換部２０８０は、対象データに含まれる部分データの数に基づいて、出力する整数を決定する。
【０１１３】
整数変換部２０８０が S-BOX を利用する形態である場合、例えば整数変換部２０８０は、以下に示す方法で S-BOX を初期化する。まず、整数変換部２０８０は、データ取得部２０２０が対象データを取得した際に、データ取得部２０２０から、対象データのサイズを取得する。次に、整数変換部２０８０は、上述した対応テーブルのレコードを全て削除する。そして、整数変換部２０８０は、対象データのサイズの範囲内であり、かつ互いに異なる乱数を順に生成し、生成した乱数を順に対応テーブルのレコードとして登録していく。ここで、所定の範囲内の乱数を生成する技術は既知の技術であるため、説明を省略する。
【０１１４】
なお、中間データ生成部２０４０が対象データの先頭の部分データの位置を 0 として扱う場合に、S-BOX、乱数生成部、又はハッシュ値生成部から出力される整数の最小値が m であるとする。この場合、整数変換部２０８０は、S-BOX、乱数生成部、又にハッシュ値生成部によって出力される整数から m を引いた値を出力する。
【０１１５】
ここで、実施形態１で説明したように、中間データ生成部２０４０は、対象データから得られる所定サイズごとの各ブロックについて、そのブロックに含まれる部分データを、別のブロックに含まれる部分データを用いて変換する。したがって、あるブロックに含まれる複数の部分データそれぞれの位置を整数変換部２０８０に入力した場合、少なくとも１つの出力は、他のブロックに含まれる部分データの位置を示す必要がある。整数変換部２０８０は、このことを考慮して実現される必要がある。例えば S-BOX が有する対応テーブルを初期化する際には、このことを考慮して初期化する。そのために、例えば整数変換部２０８０は、S-BOX を初期化する際に、対象データのサイズ、上記所定サイズ、及び各部分データのサイズを取得する。
【０１１６】
＜作用・効果＞

本実施形態によれば、中間データ生成部２０４０において、各ブロックに含まれる部分データの変換に用いるブロックの位置を、整数変換部２０８０によって出力される整数によって決める。そのため、実施形態２の中間データ生成部２０４０による中間データの生成方法と比較し、より柔軟に中間データの生成方法を決めることができる。

【０１１７】

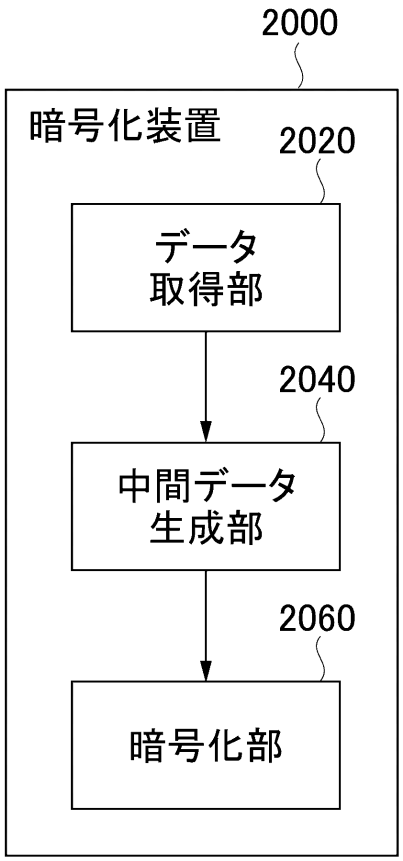
以上、図面を参照して本発明の実施形態について述べたが、これらは本発明の例示であり、上記実施形態の組み合わせ、及び上記実施形態以外の様々な構成を採用することもできる。

【符号の説明】

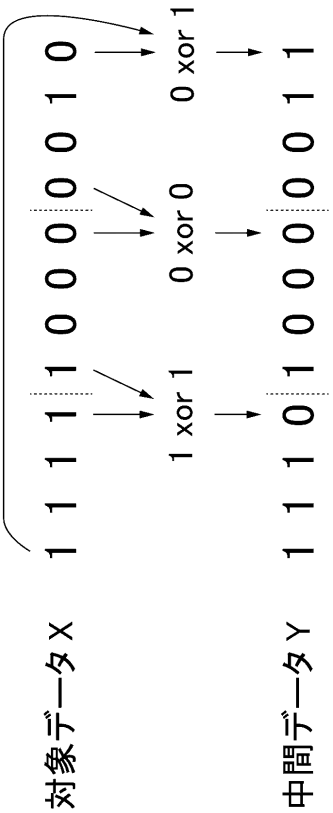
【０１１８】

２００ 対応テーブル
２０２ 入力
２０４ 出力
１０２０ バス
１０４０ プロセッサ
１０６０ メモリ
１０８０ ストレージ
１２２０ データ取得モジュール
１２４０ 中間データ生成モジュール
１２６０ 暗号化モジュール
２０００ 暗号化装置
２０２０ データ取得部
２０４０ 中間データ生成部
２０６０ 暗号化部
２０８０ 整数変換部
３０００ 復号装置
３０２０ 暗号文取得部
３０４０ 暗号文復号部
３０６０ 対象データ復元部

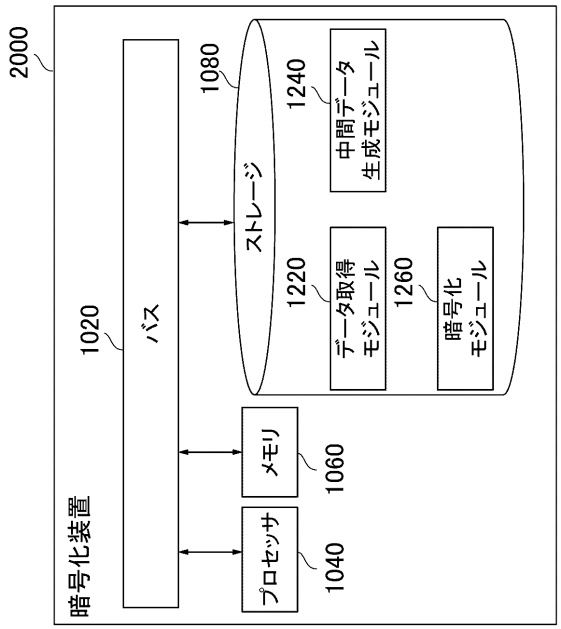
【図１】


2000
暗号化装置
2020
データ
取得部
2040
中間データ
生成部
2060
暗号化部

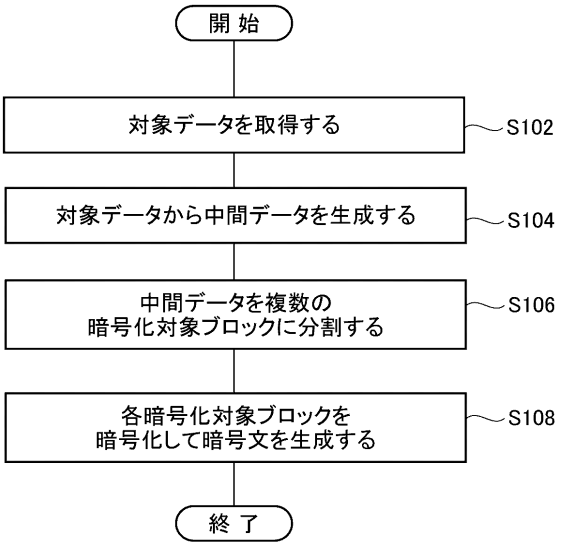
【図２】


1 1 1 1 1 1 0 0 0 0 0 0 1 0
1 xor 1
0 xor 0
0 xor 1
1 1 1 1 1 0 0 0 0 0 0 0 1 1
対象データＸ
中間データＹ

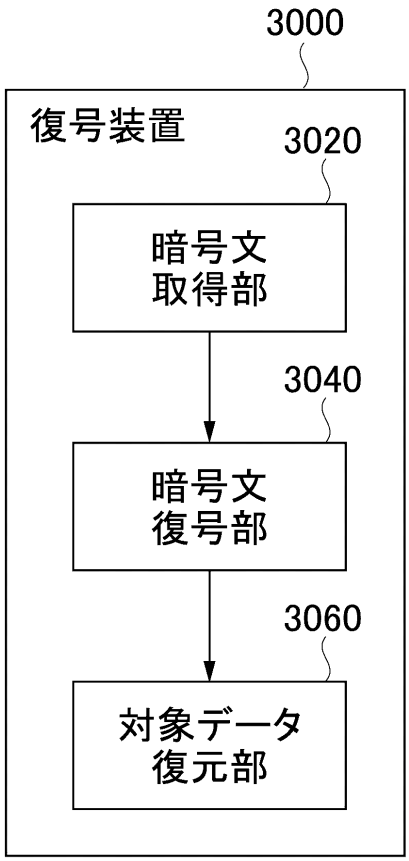
【図３】


2000
暗号化装置
1020
バス
1080
ストレージ
1240
中間データ
生成モジュール
1220
データ取得
モジュール
1260
暗号化
モジュール
メモリ
1060
プロセッサ
1040

【図４】


開始
対象データを取得する
S102
対象データから中間データを生成する
S104
中間データを複数の
暗号化対象ブロックに分割する
S106
各暗号化対象ブロックを
暗号化して暗号文を生成する
S108
終了

【図５】



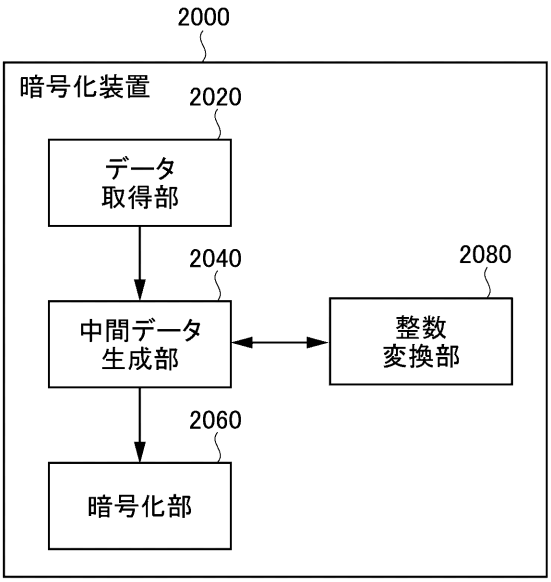
【図６】

```
procedure f (var a: array of byte; n, m: integer);
var
    i, j: integer;
begin
    for j := 1 to m do
    begin
        a[0] := a[0] + a[n-1];
        for i := 1 to n-1 do
        begin
            a[i] := a[i] + a[i-1];
        end;
    end;
end;
```

【図７】

```
procedure finv (var a: array of byte; n, m: integer);
var
    i, j: integer;
begin
    for j := 1 to m do
    begin
        for i := n-1 downto 1 do
        begin
            a[i] := a[i] - a[i-1];
        end;
        a[0] := a[0] - a[n-1];
    end;
end;
```

【図８】



【図９】

200

202 入力	204 出力
1	15
2	23
3	30
4	21
5	10
・ ・ ・	・ ・ ・