(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4737334号
(P4737334)
(45)発行日　平成23年7月27日(2011.7.27)　(24)登録日　平成23年5月13日(2011.5.13)

(51)Int.Cl.　F I
G09C 1/00 (2006.01)　G09C 1/00 610A

請求項の数 9 (全 28 頁)

(21)出願番号　特願2009-505246(P2009-505246)
(86)(22)出願日　平成20年3月19日(2008.3.19)
(86)国際出願番号　PCT/JP2008/055132
(87)国際公開番号　WO2008/114829
(87)国際公開日　平成20年9月25日(2008.9.25)
審査請求日　平成23年2月18日(2011.2.18)
(31)優先権主張番号　特願2007-71432(P2007-71432)
(32)優先日　平成19年3月19日(2007.3.19)
(33)優先権主張国　日本国(JP)

早期審査対象出願

(73)特許権者　800000068
学校法人東京電機大学
東京都千代田区神田錦町２－２
(74)代理人　100083806
弁理士　三好　秀和
(72)発明者　鈴木　秀一
東京都千代田区神田錦町２丁目２番地　学校法人東京電機大学内

審査官　西田　聡子

最終頁に続く

(54)【発明の名称】暗号装置、復号装置、暗号プログラム、復号プログラム、及び記録媒体

(57)【特許請求の範囲】

【請求項１】

平文を入力する入力部と、

第一の擬似乱数列を生成する擬似乱数生成部と、

前記第一の擬似乱数列を初期値として第二の擬似乱数列を生成する公開可能な擬似乱数生成部と、

前記平文と前記第二の擬似乱数列とを排他的論理和したデータに前記第一の擬似乱数列をヘッダとして付加したデータを再配置表を用いて一体化する変換処理部と、

前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理部と、

前記生成された暗号文を出力する出力部と、

を備えたことを特徴とする暗号装置。

【請求項２】

平文を入力する入力部と、

第一の擬似乱数列を生成する擬似乱数生成部と、

前記第一の擬似乱数列を鍵として前記平文をブロック暗号化するブロック暗号文生成部と、

前記ブロック暗号化されたデータに前記第一の擬似乱数列をヘッダとして付与したデータを再配置表を用いて一体化する変換処理部と、

前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理部と、
前記生成された暗号文を出力する出力部と、
を備えたことを特徴とする暗号装置。
【請求項３】
暗号文を入力する入力部と、
前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理部と、
前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理部と、
前記第１のデータを初期値として擬似乱数列を生成する公開可能な擬似乱数生成部と、
生成された擬似乱数列と前記第２のデータとを排他的論理和したデータを平文として出力する出力部と、
を備えたことを特徴とする復号装置。
【請求項４】
暗号文を入力する入力部と、
前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理部と、
前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理部と、
擬似乱数列を生成する擬似乱数生成部と、
生成された擬似乱数列を鍵として用いることにより前記第２のデータをブロック復号化したデータを平文として出力する出力部と、
を備えたことを特徴とする復号装置。
【請求項５】
コンピュータを、
平文を入力する入力手段と、
第一の擬似乱数列を生成する擬似乱数生成手段と、
前記第一の擬似乱数列を初期値として第二の擬似乱数列を生成する公開可能な擬似乱数生成手段と、
前記平文と前記第二の擬似乱数列とを排他的論理和したデータに前記第一の擬似乱数列をヘッダとして付加したデータを再配置表を用いて一体化する変換処理手段と、
前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理手段と、
前記生成された暗号文を出力する出力手段と、
して機能させることを特徴とする暗号プログラム。
【請求項６】
コンピュータを、
平文を入力する入力手段と、
第一の擬似乱数列を生成する擬似乱数生手段と、
前記第一の擬似乱数列を鍵として前記平文をブロック暗号化するブロック暗号文生成手段と、
前記ブロック暗号化されたデータに前記第一の擬似乱数列をヘッダとして付与したデータを再配置表を用いて一体化する変換処理手段と、
前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗

号文として生成する再配置処理手段と、
前記生成された暗号文を出力する出力手段と、
して機能させることを特徴とする暗号プログラム。
【請求項７】
コンピュータを、
暗号文を入力する入力手段と、
前記暗号文を所定の分割数に分割し、再配置表を用いて分割されたデータを元の配置に戻す逆再配置処理手段と、
前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理手段と、
前記第１のデータを初期値として擬似乱数列を生成する公開可能な擬似乱数生成手段と、
生成された擬似乱数列と前記第２のデータとを排他的論理和したデータを平文として出力する出力手段と、
して機能させることを特徴とする復号プログラム。
【請求項８】
コンピュータを、
暗号文を入力する入力手段と、
前記暗号文を所定の分割数に分割し、再配置表を用いて分割されたデータを元の配置に戻す逆再配置処理手段と、
前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理手段と、
擬似乱数列を生成する擬似乱数生成手段と、
生成された擬似乱数列を鍵として用いることにより前記第２のデータをブロック復号化したデータを平文として出力する出力手段と、
して機能させることを特徴とする復号プログラム。
【請求項９】
請求項５乃至８のいずれか１項に記載のプログラムが記録されたコンピュータが読み取り可能な記録媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、既知平文攻撃を排除可能とする暗号装置及び暗号プログラム、これらの暗号装置及び暗号プログラムによって作成された暗号文を復号する復号装置及び復号プログラム、及びこれらのプログラムを記録する記録媒体に関する。
【背景技術】
【０００２】
従来使用されてきた暗号方式（共通鍵暗号方式：ブロック暗号、ストリーム暗号）は、既知平文攻撃を許容する（非特許文献１）。既知平文攻撃とは、平文とその暗号文との組を多数用いて暗号鍵を特定する攻撃のことであり、線形攻撃や差分攻撃などがその代表的な手法である。ＤＥＳ（Data Encryption Standard）やＡＥＳ（Advanced Encryption Standard）などの標準的なブロック暗号方式では、鍵スケジュールと１０ラウンド以上のデータ攪拌とによって、これらの攻撃に対抗してきた。
【非特許文献１】“Cryptography: Theory and practice,3rd-Ed.”, Stinson, D.R., Chapman & Hall/CRC Press Inc. 2006.
【発明の開示】
【０００３】
上記の手法においてセキュリティレベルを引き上げるためには、暗号鍵の長さを長くし

たり、データ攪拌処理において使用されるＳ－ＢＯＸの非線形性を強くしたりする必要がある。しかし、その分、暗号化の速度が低下し、共通鍵暗号方式の従来の利点を損なう。また、既知平文攻撃に対して強い特別なＳ－ＢＯＸを設計する必要があり、そのための研究開発費や計算機資源も増大することになる。

【０００４】

本発明は、上記の実情を鑑みて為されたものであり、既知平文攻撃を排除可能とすると共に、高速度で暗号化を行うことのできる暗号装置及び暗号プログラム、これらの暗号装置及び暗号プログラムによって作成された暗号文を復号化する復号装置及び復号プログラム、及びこれらのプログラムを記憶する記憶媒体を提供することを目的とする。

【０００５】

本発明の第１の側面によれば、平文を入力する入力部と、第一の擬似乱数列を生成する擬似乱数生成部と、前記第一の擬似乱数列を初期値として第二の擬似乱数列を生成する公開可能な擬似乱数生成部と、前記平文と前記第二の擬似乱数列とを排他的論理和したデータに前記第一の擬似乱数列をヘッダとして付加したデータを再配置表を用いて一体化する変換処理部と、前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理部と、前記生成された暗号文を出力する出力部とを備えたことを特徴とする暗号装置が提供される。

【０００６】

本発明の第２の側面によれば、平文を入力する入力部と、第一の擬似乱数列を生成する擬似乱数生成部と、前記第一の擬似乱数列を鍵として前記平文をブロック暗号化するブロック暗号文生成部と、前記ブロック暗号化されたデータに前記第一の擬似乱数列をヘッダとして付与したデータを再配置表を用いて一体化する変換処理部と、前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理部と、前記生成された暗号文を出力する出力部とを備えたことを特徴とする暗号装置が提供される。

【０００７】

本発明の第３の側面によれば、暗号文を入力する入力部と、前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理部と、前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理部と、

前記第１のデータを初期値として擬似乱数列を生成する公開可能な擬似乱数生成部と、生成された擬似乱数列と前記第２のデータとを排他的論理和したデータを平文として出力する出力部とを備えたことを特徴とする復号装置が提供される。

【０００８】

本発明の第４の側面によれば、暗号文を入力する入力部と、前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理部と、前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理部と、擬似乱数列を生成する秘密の擬似乱数生成部と、生成された擬似乱数列を鍵として用いることにより前記第２のデータをブロック復号化したデータを平文として出力する出力部とを備えたことを特徴とする復号装置が提供される。

【０００９】

本発明の第５の側面によれば、コンピュータを、平文を入力する入力手段と、第一の擬似乱数列を生成する擬似乱数生成手段と、前記第一の擬似乱数列を初期値として第二の擬似乱数列を生成する公開可能な擬似乱数生成手段と、前記平文と前記第二の擬似乱数列とを排他的論理和したデータに前記第一の擬似乱数列をヘッダとして付加したデータを再配置表を用いて一体化する変換処理手段と、前記一体化されたデータを複数のブロックデー

タに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理手段と、前記生成された暗号文を出力する出力手段として機能させることを特徴とする暗号プログラムが提供される。

【００１０】

本発明の第６の側面によれば、平文を入力する入力手段と、第一の擬似乱数列を生成する擬似乱数生手段と、前記第一の擬似乱数列を鍵として前記平文をブロック暗号化するブロック暗号文生成手段と、前記ブロック暗号化されたデータに前記第一の擬似乱数列をヘッダとして付与したデータを再配置表を用いて一体化する変換処理手段と、前記一体化されたデータを複数のブロックデータに分割し、前記再配置表を秘密鍵として用いることにより前記複数のブロックデータを重複することなく再配置したデータを暗号文として生成する再配置処理手段と、前記生成された暗号文を出力する出力手段として機能させることを特徴とする暗号プログラムが提供される。

【００１１】

本発明の第７の側面によれば、コンピュータを、暗号文を入力する入力手段と、前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理手段と、前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理手段と、前記第１のデータを初期値として擬似乱数列を生成する公開可能な擬似乱数生成手段と、生成された擬似乱数列と前記第２のデータとを排他的論理和したデータを平文として出力する出力手段として機能させることを特徴とする復号プログラムが提供される。

【００１２】

本発明の第８の側面によれば、コンピュータを、暗号文を入力する入力手段と、前記暗号文を所定の分割数に分割し、再配置表を秘密鍵として用いて分割されたデータを元の配置に戻す逆再配置処理手段と、前記再配置表を用いて逆再配置されたデータを逆変換し、逆変換されたデータを先頭から所定のデータ長を有する第１のデータと残りの第２のデータと分離する逆変換処理手段と、擬似乱数列を生成する秘密の擬似乱数生成手段と、生成された擬似乱数列を鍵として用いることにより前記第２のデータをブロック復号化したデータを平文として出力する出力手段として機能させることを特徴とする復号プログラムが提供される。

【００１３】

本発明の第９の側面によれば、上記のプログラムのうちの少なくとも一つが記録されたコンピュータが読み取り可能な記録媒体が提供される。

【図面の簡単な説明】

【００１４】

【図１】図１は、本発明に係る暗号方式の特徴をストリーム暗号において示した概念図である。

【図２】図２は、本発明に係る暗号装置の概略的な構成を示したブロック図である。

【図３】図３は、本発明に係る暗号化処理の手順を示したフローチャートである。

【図４】図４は、図３に示した暗号化処理における変換処理の具体的な手順を示したフローチャートである。

【図５】図５は、図３に示した暗号化処理における再配置処理の具体的な手順を示したフローチャートである。

【図６】図６は、図４に示した変換処理の一例を示した図である。

【図７】図７は、本発明に係る復号装置の概略的な構成を示したブロック図である。

【図８】図８は、本発明に係る復号化処理の手順を示したフローチャートである。

【図９】図９は、図８に示した復号化処理における逆再配置処理の具体的な手順を示したフローチャートである。

【図１０】図１０は、図８に示した復号化処理における逆変換処理の具体的な手順を示し

たフローチャートである。
【図１１】図１１は、本発明のその他の暗号装置の概略的な構成を示したブロック図である。
【図１２】図１２は、図１１に示した暗号装置における暗号化の手順を示した概念図である。
【発明を実施するための最良の形態】
【００１５】
図１は、本発明に係る暗号方式の特徴をストリーム暗号を用いて示した概念図である。図１に示すように、本発明の暗号方式は、
（１）　秘密の擬似乱数発生器G_0で第一の擬似乱数列ｒと第ニの擬似乱数列Ｒとを生成するステップと、
（２）　第一の擬似乱数列ｒを初期値として公開可能な擬似乱数発生器G_1で擬似乱数列r_xを生成するステップと、
（３）　平文ｘと擬似乱数列r_xとの排他的論理和をとり平文ｘを暗号化するステップと、
（４）　第一の擬似乱数列ｒをヘッダとして暗号文c_xに付加したデータ（ｒ，c_x）をＳ－ＢＯＸ（Substitution Box）などを用いて変換（一体化）処理するステップと、
（５）　変換されたデータd_xをｎ個のブロックB_i（ｉ＝０，１，・・・，ｎ－１）に分割するステップと、
（６）　第二の擬似乱数列Ｒに基づき生成された再配置表Ｋを秘密鍵として用いることによりｎ個のブロックb_i（ｉ＝０，１，・・・，ｎ－１）を重複することなく再配置させたデータを最終的な暗号文fc_xとして生成するステップと、
を備える。
【００１６】
さらに、この暗号方式では、必要に応じて、ステップ（４）～（６）における変換処理から再配置処理までの処理を数回繰り返すステップを含めてもよい。
【００１７】
なお、再配置表Ｋは、０，１，・・・，ｎ－１までのｎ個の整数の重複のない十分にランダムな並び替えを表す表（再配置表）のことであり、以降では、Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）と表すことにする。再配置表Ｋは、暗号化処理を行うたびに、秘密の擬似乱数生成器G_0が生成する第二の擬似乱数に基づき作成されるものであり、ｎ！通りの可能性の中から一つが秘密鍵として選ばれる。
【００１８】
また、上記（３）では、平文ｘと疑似乱数r_xとの排他的論理和をとったが、正確には、図１に示すように、疑似乱数r_xとの排他的論理和をとる対象は、平文ｘと、平文ｘのデータ長などの情報を含んだヘッダ情報と、必要に応じてパディングｐとを合わせたものである。以降の実施例においては、ヘッダ情報ｕと平文ｘと必要に応じてパディングｐとを合わせたデータを改めて平文ｘと見なす。
【００１９】
本発明においては、この暗号方式を“再配置暗号”と称する。
【００２０】
以下に、本発明の実施形態を、図面を用いて詳細に説明する。
【００２１】
［暗号装置］
図２は、本発明に係る暗号装置の概略的な構成を示したブロック図である。暗号装置１０は、入力部２０と、第一擬似乱数生成部３０と、第二擬似乱数生成部４０と、変換処理部５０と、再配置処理部６０と、出力部７０と、記憶部８０と、制御部９０とを備える。このうち、記憶部８０と制御部９０とを除く部分を暗号文生成部１０Ａと称することにする。
【００２２】

入力部２０は、送信者が平文ｘを入力するための入力インターフェースである。

【００２３】

第一擬似乱数生成部３０は、予測困難な擬似乱数列を生成する擬似乱数生成器であり、暗号装置１０のシステムクロックや入力部２０からの入力タイミングなどを利用することができるが、より乱数に近いものとして熱雑音などを用いることもできる。第一擬似乱数生成部３０として熱雑音による擬似乱数生成器を用いた場合、その他の場合と比べてコストを低減できる。第一擬似乱数生成部３０としては、毎回異なる擬似乱数を生成することが重要であるので、使い捨て擬似乱数生成器を用いてもよい。第一擬似乱数生成部３０に使用される擬似乱数生成器は送信者と受信者の間で秘密にされる。第一擬似乱数生成部が生成する擬似乱数については、送信者、受信者を含め誰も一切知っている必要がない。

【００２４】

第二擬似乱数生成部４０は、統計的に偏りのない擬似乱数列を生成する擬似乱数生成器であり、メルセンヌ・ツイスターを用いることができる。メルセンヌ・ツイスターは、統計学的に優れた擬似乱数列を生成できるが、暗号学的には安全ではない。しかし、本発明においては、第一擬似乱数生成部３０で生成した第一の擬似乱数列ｒを後述する変換処理部５０による非線形変換と後述する再配置処理部６０による再配置により秘匿にすることができるので、メルセンヌ・ツイスターの使用が可能である。第二擬似乱数生成器４０に使用される擬似乱数生成器は送信者と受信者以外に対して公開してもよい。

【００２５】

もちろん、第一擬似乱数生成部３０としてメルセンヌ・ツイスターを使用してもよい。

【００２６】

変換処理部５０は、データを非線形的に変換する処理を行うＳ－ＢＯＸなどを含めた関数系から構成されるものであり、後述する変換処理を行う。

【００２７】

再配置処理部６０は、データをブロックに分割し、当該ブロックを再配置するための再配置表を含めた関数系から構成されるものであり、後述する再配置処理を行う。

【００２８】

出力部７０は、最終的に生成された暗号文を受信者へ出力するための出力インターフェースである。

【００２９】

記憶部８０は、入力部２０～出力部７０から成る暗号生成部１０Ａが生成した各種のデータの格納を行うサブメモリと、後述する暗号化処理の各ステップを実行するためのコンピュータに読み取り可能な暗号プログラムを格納するメインメモリとから構成される。記憶手段８０は、ＲＡＭ（Random Access Memory）やＲＯＭ（Read Only Memory）などから構成される。さらに、記憶部８０のサブメモリとメインメモリとを別体として構成し、メインメモリ部分を磁気ハードディスク、フロッピー（登録商標）ディスク、CD-ROMなどの光ディスク、磁気テープ、メモリチップ等に記憶させてもよい。

【００３０】

制御部９０は、記憶部８０から読み出した暗号プログラムに従って、入力部２０～記憶部８０を制御するＣＰＵ(Central Processing Unit)を備える。

【００３１】

本実施例では、暗号装置１０を、暗号文生成部１０Ａ及び制御部９０と、記憶部８０とを一体化した構成としたが、記憶部８０を独立した記憶装置として暗号文生成部１０Ａ及び制御部９０とから切り離した構成としてもよい。いずれの構成においても、暗号装置１０はコンピュータによって実現されるものであり、入力部２０～出力部７０は、制御部９０により記憶部８０から読み出された暗号プログラムに従って制御される。

【００３２】

ここで、コンピュータとは、構造化された入力を所定の規則に従って処理し、処理した結果を構造化して出力する装置のことを指し、例えば、汎用コンピュータ、スーパーコンピュータ、メインフレーム、ワークステーション、マイクロコンピュータ、サーバ等が含

まれる。また、通信ネットワーク（例えば、イントラネット、ローカルエリアネットワーク（LAN）、ワイドエリアネットワーク（WAN）、及びこれらの組み合わせから成る通信ネットワーク）を介して接続された２つ以上のコンピュータから成る構成（例えば、分散コンピュータシステム）であってもよい。

【００３３】

また、ここでのコンピュータには、携帯電話やモバイル端末、家電製品や自動車などの制御チップ、コントローラ、ＩＣカードに組み込まれた演算装置なども含まれる。

【００３４】

［暗号化処理］

以上を前提として、図２に示した暗号装置１０によって行われる再配置暗号について詳細に説明する。図３は、図２に示した暗号装置１０によって行われる再配置暗号の処理手順を示したフローチャートである。

【００３５】

送信者により入力部２０から平文ｘ（長さ：ｇワード）が入力されると、制御部９０は、これを記憶部８０に記憶させ、記憶部８０に格納された暗号プログラムに従い、第一擬似乱数生成部３０～生成部７０に対して以下に示す処理を行うように促す。

【００３６】

ステップＳ１０において、制御部９０は、第一擬似乱数生成部３０に対して、第一の擬似乱数列ｒ（長さ：ａワード）を生成させ、これを記憶部８０に記憶させる。なお、第一の擬似乱数列ｒの長さａは任意に設定することができる。

【００３７】

＜ヘッダｒの生成アルゴリズムの実施例＞

ステップＳ１０における第一擬似乱数生成部３０によるヘッダｒの生成アルゴリズムは、以下のように記述される。

```
r: array[0..a-1] of the word;
Randomize; //initialize G0 by the clock;
for i:=0 to a-1 do
  r[i]:=G0;
```

【００３８】

次に、ステップＳ２０において、制御部９０は、記憶部８０から予め格納された分割数ｎを読み出し、第一擬似乱数生成部３０に対して、０，１，・・・，ｎ－１までのｎ個の整数の疑似乱数列（第二の擬似乱数列）を生成させ、これを再配置表Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）として記憶部８０に記憶させる。

【００３９】

＜再配置表Ｋの生成アルゴリズムの実施例＞

ステップＳ２０における第一擬似乱数生成部３０による再配置表Ｋの生成アルゴリズムは、以下のように記述される。

```
k: array[0..n-1] of the word;
Randomize; //initialize G0 by the clock;
for i:=0 to a-1 do
  k[i]:=i;
for i:=0 to N do
begin
  s:=G0 mod n;
  t:=G0 mod n;
  x:=k[s];
  k[s]:=k[t];
  k[t]:=x;
end;
```

ここで、１ワードは８ビット、１６ビット、または３２ビットの符号なし整数を表す。ま

た、ここでは、第一擬似乱数生成部３０は、毎回１ワードの擬似乱数を出力するものとしている。

【００４０】

なお、一般には、第一擬似乱数部３０が生成する擬似乱数列は同じ数字の重複を含む。このとき、第二の擬似乱数列から生成される再配置表Ｋにおけるｎ個の数字（０，１，・・・，ｎ－１）の並びには同じ数字の重複が生じてしまう。具体的には、分割数ｎが十分に大きければ、第一擬似乱数生成部３０が生成する第二の擬似乱数列には同じ数字の重複が存在することはないと考えられるが、分割数ｎが小さい場合には、その可能性が生じる。そのような場合には、ｎ個の数字の並びが重複せず、かつ十分ランダムに配列されるように工夫して再配置表Ｋを作成する必要がある。

【００４１】

その場合、後述する［具体的な実装］における擬似乱数生成器Ｇ１を使い、生成した擬似乱数列を用いて数万回並べ替えることが考えられる。しかしながら、それよりも効率的な方法として、第一擬似乱数生成部３０が生成する擬似乱数列を周期的に拡張して、拡張した擬似乱数列から再配置表Ｋを生成し、さらに、生成された再配置表Ｋの統計的な偏りを補正することにより最終的に所望の再配置表Ｋを得るという方法が考えられる。

【００４２】

この方法は、次の３つのステップから構成される。

【００４３】

｛ステップ１｝

第一擬似乱数生成部３０において、短い擬似乱数列から２５６バイトの擬似乱数列を作成する。例えば、３２バイトの擬似乱数列を２５６バイトの擬似乱数列に拡張するアルゴリズムは、以下のように記述される。

＜実施例＞

```
procedure set32byte;
var
  i, j, a: integer
  d: array[0..255] of byte;
begin
  Randomize;
  for i:=0 to 31 do
  begin
     j:=random(256);
     d[i]:=j;
  end;
  for i:=32 to 255 do
  begin
    a:= i mod 32;
    d[i]:=d[a];
  end;
end;
```

｛ステップ２｝

【００４４】

次に、記憶部８０の擬似乱数列のメモリ領域に２５６バイトの擬似乱数列を順次格納していき、同じ数字が出たら次のメモリに格納するといった処理を繰り返すことにより再配置表Ｋを生成する。例えば、２５６バイトの擬似乱数列から２５６バイトの再配置表Ｋを生成するアルゴリズムは、以下のように記述される。

＜実施例＞

```
procedure generate256k;
var
```

```
  d, st, k: array[0..255] of byte;
  i: integer;
  procedure seti; //dからkを作成する。
  var
     j: integer;
  begin
     if st[d[i]]=0 then
     begin
       k[d[i]]:=i;
       st[d[i]]:=1;
     end
     else
     begin
       j:=(d[i]+19) and 255;
       while st[j]=1 do
        begin
          j:=(j+67) and 255;
        end;
        k[j]:=i;
        st[j]:=1;
     end;
  end;
begin
  read_d;
  clear_st;
    for i:=0 to 255 do
  begin
    seti;
  end;
end;
```

{ステップ３}

【００４５】

次に、ステップ２で作成された再配置表Ｋの統計的な偏りを、例えば、以下に示すようなアルゴリズムを用いて補正する。

＜実施例＞

```
procedure revise;
var
  k, k2: array[0..255] of byte;
  i: integer;
begin
  read_k;
  for I:=0 to 255 do
  begin
    k2[i]:=k[255-k[i]];
  end;
end;
```

【００４６】

次に、ステップＳ３０において、制御部９０は、送信者により入力部２０から入力された平文ｘを読み込み、これを平文ｘのデータ長ｇなどの情報を含んだヘッダ情報ｕ（長さ：ｑワード）と共にこれを記憶部８０に記憶させる。ここで、ヘッダ情報ｕの長さｑは４

ワードの符号なし整数として設定することができる。

【００４７】

次に、ステップＳ４０において、制御部９０は、第一の擬似乱数列ｒを記憶部８０から読み出し、第二擬似乱数生成部４０に対して、第一の擬似乱数列ｒを初期値として、平文ｘと同じ長さ（ｎｍ－ａ－１ワード）の第二の擬似乱数列$r_x=(r_0, r_1, \cdots, r_{nm-a-1})$を生成させ、これを記憶部８０に記憶させる。

【００４８】

なお、このステップにおいて、入力部２０から入力される平文ｘの長さｇが分割数ｎの倍数ではない場合には、制御部９０は、ｖ≡－ａ－ｑ－ｇ（ｍｏｄ　ｎ）となるような最小の非負整数ｖを算出し、パディングｐとしてｖワードの長さの擬似乱数列ｚを第一擬似乱数生成部３０に生成させ、平文ｘの最後に付加する処理を行う。そして、制御部９０は、ヘッダ情報ｕ、平文ｘ、パディングｐを合わせたデータを改めて平文$x=(u, x, z)=(x_0, x_1, \cdots, x_{nm-a-1})$として記憶部８０に記憶させる。また、この場合、再配置処理において分割される各ブロックの長さを表す整数“ｍ”は、ｍ＝（ａ＋ｑ＋ｇ＋ｖ）／ｎとして算出される。

【００４９】

次に、ステップＳ５０において、制御部９０は、記憶部８０から平文$x=(x_0, x_1, \cdots, x_{nm-a-1})$と第二の擬似乱数列$r_x=(r_0, r_1, \cdots, r_{nm-a-1})$とを読み出し、両者の排他的論理和をとる（$c_i=x_i$ ＸＯＲ r_i（ｉ＝０，１，・・・，ｎｍ－ａ－１））ことにより暗号文$c_x=(c_0, c_1, \cdots, c_{nm-a-1})$を生成し、これを記憶部８０に記憶させる。

【００５０】

なお、本実施例では、このステップにおいて第二の擬似乱数列r_xと平文ｘとの間で一度に排他的論理和する構成としたが、その代わりに、制御部９０は、ステップＳ４０において第二の擬似乱数生成部４０に対して１ワードづつ擬似乱数を生成させ、そのつど、ステップＳ５０において平文ｘの１ワードと逐次的に排他的論理和する構成としてもよい。

【００５１】

次に、ステップＳ６０において、制御部９０は、記憶部８０から第一の擬似乱数列ｒと暗号文$c_x=(c_0, c_1, \cdots, c_{nm-a-1})$とを読み出し、第一の擬似乱数列ｒを暗号文$c_x$のヘッダとして付加したデータ$(r, c_x)=(c_0, c_1, \cdots, c_{nm-1})$を生成させ、これを改めて$c_x=(r, c_x)=(c_0, c_1, \cdots, c_{nm-1})$として記憶部８０に記憶させる。

【００５２】

次に、ステップＳ７０において、制御部９０は、ステップＳ８０～Ｓ１００で行われる変換処理と分割処理と再配置処理とを１セットとした処理のラウンド数を表すＣｔを立て（Ｃｔ＝１）、ステップＳ８０へ処理を進める。

【００５３】

ステップＳ８０において、制御部９０は、記憶部８０からデータ$c_x=(c_0, c_1, \cdots, c_{nm-1})$を読み出し、変換処理部５０に対して、例えば、図４に示すように、$c_{(dm+i) \bmod nm}=c_{(dm+i) \bmod nm}+c_{i \bmod nm}$（ｉ＝０，１，・・・，ｎｍ－１）として加算した値（左辺の$c_{(dm+i) \bmod nm}$）をＳ－ＢＯＸを用いてバイト単位で変換する変換処理を行わせ、変換されたデータを$d_x=F(r, c_x)=(d_0, d_1, \cdots, d_{nm-1})$として記憶部８０に記憶させる。ここで、“ｄ”はブロック差分と呼ばれる整数であり、０＜ｄ＜ｎの範囲で予め決められているものとする。また、図４に示した変換処理のフローチャートの中の関数ｗは、ワードとバイト配列の共用体であり、次の形式で記憶部８０に記憶される。

```
Tunion=record
  case integer of
  1: (d:Word);
  2: (h:array[0..3] of byte);
```

```
  end;
  w: Tunion;
```

ここでは、１ワードを４バイトとして扱っている。

【００５４】

図６は、図４に示した変換処理のフローチャートにおいて、ｎ＝３，ｍ＝２，ｄ＝１の場合を視覚化したものである。この例では、記号Ａ１～Ａ３の処理で示したように、長さ２ワードのブロックデータを縦方向に順次積層し、記号Ａ４の処理において積層した値をブロック単位で足し合わせ、最後に記号Ａ５の処理において、ブロック単位でＳ－ＢＯＸを用いて変換することにより変換処理を行っている。

【００５５】

なお、この変換処理は、可逆な変換（１対１対応の変換）であればどのような処理を行ってもよい。例えば、ここで用いた加算の変わりに減算を用いてよい。また、Ｓ－ＢＯＸについても、非線形な変換であればどのような関数系を用いてもよい。しかしながら、本発明の主旨からは、再配置表Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）から次のようにして生成されるＳ－ＢＯＸを用いることが好ましい。

【００５６】

＜再配置表ＫによるＳ－ＢＯＸの生成アルゴリズムの実施例＞

再配置表Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）からは様々な方法でＳ－ＢＯＸを生成することができるが、ここでは簡単な実施例をあげる。

```
（n<=256のとき）
e:=256-n;
for i:=0 to e-1 do
  s[i]:=n+i;
for i:=e to 255 do
  s[i]:=k[i-e];
（n>256のとき）
```

n-256≧e≧0となる整数eを一つ決める。

```
ct:=0;
for i:=0 to n-1 do
begin
  if (k[i]-e>=0) and (k[i]-e<256) then do
  begin
    s[ct]:=k[i]-e;
   ct:=ct+1
  end;
end;
```

ここで、ｎ＝２５６のときはｓ＝Ｋになり、変換処理における非線形な変換を施すために、再配置表Ｋそのものが利用できる。つまり、このときには、Ｓ－ＢＯＸは、０，１，・・・，２５５をランダムに並び替えたｓ＝（ｋ［０］，ｋ［１］，・・・，ｋ［２５５］）となる。

【００５７】

次に、ステップＳ９０において、制御部９０は、記憶部８０から変換されたデータ$d_x=(d_0, d_1, \cdots, d_{nm-1})$を読み出し、再配置処理部６０に対して、これを長さｍワードのｎ個のブロックデータ$b_i=(d_{mi}, d_{mi+1}, \cdots, d_{mi+m-1})$（ｉ＝０，１，・・・，ｎ－１）に分割させ、分割されたデータを$d_x=(b_0, b_1, \cdots, b_{n-1})$として記憶部８０に記憶させる。

【００５８】

次に、ステップＳ１００において、制御部９０は、記憶部８０から分割されたデータ$d_x=(b_0, b_1, \cdots, b_{n-1})$と再配置表Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）とを読み出し、再配置処理部６０に対して、図５に示すように、分割され

たデータ$d_x=(b_0, b_1, \cdots, b_{n-1})$を、再配置表K＝（k［0］，k［1］，・・・，k［n－1］）に基づき、$d_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$のように再配置させ、これを記憶部８０に記憶させる。なお、図５において、命令Ｍｏｖｅ（ｘ［ｉ］，ｙ［ｊ］，ｚ）は、ｘ［ｉ］のアドレスからｙ［ｊ］のアドレスへzバイトの記憶内容をコピーする処理を表す。

【００５９】

次に、ステップＳ１１０において、制御部９０は、フラグＣｔの値をインクリメントし（Ｃｔ＝２）、ステップＳ１２０において、インクリメントされた値が所定のラウンド回数ｈを越えたか否かを判定する。

【００６０】

ステップＳ１２０において、制御部９０によりインクリメントされた値が所定のラウンド回数ｈを越えたと判定された場合は、ステップＳ１３０に処理を進める。この場合、制御部９０は、記憶部８０から再配置されたデータ$d_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$を読み出し、最終的な暗号文$fc_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$として出力部７０に出力する。そして、出力部７０は、後述する復号装置に対して暗号文$fc_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$を送信してすべての処理を終了する。

【００６１】

一方、ステップＳ１２０において、制御部９０によりインクリメントされた値が所定のラウンド回数ｈを越えていないと判定された場合は、ステップＳ８０に戻って、インクリメントされた値が所定のラウンド回数ｈを越えるまで、ステップＳ９０～Ｓ１００までの処理を繰り返した後、すべての処理を終了する。

【００６２】

［暗号装置１０の効果］

暗号装置１０は、実質的に二つの鍵を用いて平文ｘを暗号化処理している。第一の鍵は、第一擬似乱数生成部（秘密の擬似乱数生成器）３０により生成され、平文ｘにヘッダとして付加される第一の擬似乱数列ｒである。第二の鍵は、同じく第一擬似乱数生成部３０により生成され、再配置処理の際に使用される第二の疑似乱数列から生成される再配置表Ｋである。これらの鍵を用いることによって、暗号装置１０は、以下のような効果をもたらす。

【００６３】

（１）第一擬似乱数生成部３０は、平文を暗号化するたびに異なる第一の擬似乱数列を生成することができる。異なる第一の擬似乱数列を初期値として第二擬似乱数生成部（公開可能な擬似乱数生成器）４０で生成される第三の擬似乱数列の間には相関がほとんどないので、本暗号を既知平文攻撃することは極めて難しい。

【００６４】

（２）第一擬似乱数生成部３０が生成する第二の擬似乱数列に同じ数の重複する配列がない場合、第一擬似乱数生成部３０はｎ！通りの可能性の中から秘密鍵として一つの再配列表Ｋを生成することができる。例えば、ｎ＞４０の場合、その組み合わせは2^{159}よりも大きくなり、さらに、最も実用的なｎ＝２５６の場合、その組み合わせの数は2^{1683}を越えることになるので、鍵の全探索は事実上不可能になる。

【００６５】

（３）分割数ｎを大きくすることに計算上のコストはかからない。また、本暗号は、擬似乱数の生成、整数の加算、メモリ内容のコピーといった高速処理が可能な演算のみから構成されているので、暗号化の実現速度は、現在標準の共通鍵方式であるＡＥＳと比較して極めて高速である。また、全体の演算回数も十分の一程度以下になる。

【００６６】

（４）上記（２）で述べたように、本暗号の安全性は、データを分割し再配置したものを元に戻すことの計算量的困難さに基づいている。このことを考慮すると、本発明は長期間同じ鍵を使用しても高いセキュリティレベルが維持できる。

【００６７】
（５）上記（４）の利点は、本暗号が長期間のデータ保存に適していることを意味する。このことから、本暗号は、従来の暗号化方式では対応できなかった分野、例えば、医療データ等の個人情報の長期保存、にも適用できる。
【００６８】
（６）また、暗号装置１０は、再配置処理部６０を設けたことにより、同じ平文ｘと同じ第一の鍵（第一の擬似乱数列）ｒ（長さ：ｋビット）とから、2^k通りの暗号文を生成できる。このことは、同じ平文ｘと同じ第一の鍵ｒとから、ｋの値を不定とした場合には毎回異なる暗号文が無数にできることを意味する。

【００６９】
（７）さらに、暗号装置１０においては、変換処理部５０で用いられるＳ－ＢＯＸとして再配置表Ｋを利用することにより、その構成は０，１，・・・，ｎ－１のｎ個の整数をランダムに配置するだけのものになるので、従来の暗号化方式におけるＳ－ＢＯＸの構成よりも簡単であり、Ｓ－ＢＯＸの研究開発にコストがかからない。
【００７０】
［復号装置］
図７は、本発明に係る復号装置の概略的な構成を示したブロック図である。復号装置１００は、入力部１２０と、公開可能な第二擬似乱数生成部１４０と、逆変換処理部１５０と、逆再配置処理部１６０と、出力部１７０と、記憶部１８０と、制御部１９０とを備える。このうち、記憶部１８０と制御部１９０とを除く部分を復号文生成部１００Ａと称することにする。
【００７１】
入力部１２０は、送信者から送られてきた暗号文ｆｃｘを受信者が受け取るための入力インターフェースである。第二擬似乱数生成部１４０は、暗号装置１０の第二疑似乱数生成部４０と同様の構成を有する。逆再配置処理部１６０は、暗号装置１０の再配置処理部６０と同様の構成を有し、後述する逆再配置処理を行う。逆変換処理部１５０は、暗号装置１０の変換処理部５０と同様の構成を有し、後述する逆変換処理を行う。出力部１７０は、最終的に復号された復号文（平文ｘ）を出力するための出力インターフェースである。
【００７２】
記憶部１８０は、入力部１２０～出力部１７０から成る復号文生成部１００Ａが生成した各種のデータの格納を行うサブメモリと、後述する復号化処理の各ステップを実行するためのコンピュータに読み取り可能な暗号プログラムを格納するメインメモリとから構成される。
【００７３】
制御部１９０は、記憶部１８０から読み出した復号プログラムに従って、入力部１２０～記憶部１８０を制御するＣＰＵを備える。
【００７４】
本実施例では、復号装置１００を、復号文生成部１００Ａ及び制御部１９０と、記憶部１８０とを一体化した構成としたが、記憶部１８０を独立した記憶装置として復号文生成部１００Ａ及び制御部１９０とから切り離した構成としてもよい。いずれの構成においても、復号装置１００はコンピュータによって実現されるものであり、入力部１２０～出力部１７０は、制御部１９０により記憶部１８０から読み出された暗号プログラムに従って制御される。
【００７５】
［復号化処理］
以上を前提として、図７に示した復号装置１００によって行われる復号化処理について詳細に説明する。図８は、図７に示した復号装置１００によって行われる復号化の処理手

順を示したフローチャートである。

【００７６】

暗号装置１０から送信された暗号文$fc_x=(f_0, f_1, \cdots, f_{nm-1})$が入力部１２０から入力されると、制御部１９０は、これを記憶部１８０に記憶させ、記憶部１８０に格納された暗号プログラムに従い、第ニ擬似乱数生成部１４０～出力部１７０に対して以下に示す処理を行うように促す。

【００７７】

ステップＳ２００において、制御部１９０は、暗号装置１０から送信された暗号文の長さｎｍを入力部１２０から読み込ませ、これを記憶部１８０に記憶させる。

【００７８】

次に、ステップＳ２１０において、制御部１９０は、記憶部１８０から暗号文fc_xの長さｎｍと予め格納された分割数ｎとを読み出す。

【００７９】

次に、ステップＳ２２０において、制御部１９０は、読み出した暗号文ｆｃｘの長さｎｍと分割数ｎとから、ブロックデータの長さｍを$m=nm/n$として算出する。

【００８０】

次に、ステップＳ２３０において、制御部１９０は、ステップＳ２４０～Ｓ２６０で行われる分割処理と逆再配置処理と逆変換処理とを１セットとした処理のラウンド数を表すフラグＣｔを立て（Ｃｔ＝１）、ステップＳ２４０へ処理を進める。

【００８１】

ステップＳ２４０において、制御部１９０は、記憶部１８０から暗号文$fc_x=(f_0, f_1, \cdots, f_{nm-1})$を読み出し、これをｎ個のブロックデータに分割して、分割されたデータを$d_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$として記憶部１８０に記憶させる。

【００８２】

次に、ステップＳ２５０において、制御部１９０は、記憶部１８０からデータ$d_x=(b_{k[0]}, b_{k[1]}, \cdots, b_{k[n-1]})$と秘密鍵Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）とを読み出し、逆再配置処理１６０に対して、図９に示すように、ｋ［ｉ］番目のブロックデータ$b_{k[i]}$をb_iへと逆配置させ、逆配置されたデータを$d_x=(b_0, b_1, \cdots, b_{n-1})$として記憶部１８０に記憶させる。

【００８３】

次に、ステップＳ２６０において、制御部１９０は、記憶部１８０から逆配置されたデータ$d_x=(b_0, b_1, \cdots, b_{n-1})$を読み出し、逆変換処理部１５０に対して、例えば、図１０に示すように、これをＳ－ＢＯＸを用いて逆変換させ、さらに、逆変換されたデータを$d_x=(d_0, d_1, \cdots, d_{nm-1})$を$c_{(dm+i) \bmod nm}=c_{(dm+i) \bmod nm}-c_{i \bmod nm}$（$i=nm-1, nm-2, \cdots, 1, 0$）として減算する逆変換処理を行わせ、逆変換されたデータを$c_x=(c_0, c_1, \cdots, c_{nm-1})$として記憶部１８０に記憶させる。

【００８４】

ここで、逆変換とは、暗号装置１０のＳ－ＢＯＸの非線形変換を表す関数ｓの逆関数ｉｓを用いた逆変換のことであり、次のプログラムで実現される。

```
for i:=0 to 255 do
  is[s[i]]:=i;
```

【００８５】

次に、ステップＳ２７０において、制御部１９０は、フラグＣｔの値をインクリメントし（Ｃｔ＝２）、ステップＳ２８０において、インクリメントされた値が所定のラウンド数ｈを越えたか否かを判定する。

【００８６】

ステップＳ２８０において、制御部１９０によりインクリメントされた値が所定のラウンド数ｈを越えたと判定された場合には、ステップＳ２９０に処理を進める。

【００８７】
ステップＳ２９０において、制御部１９０は、記憶部１８０から逆変換されたデータc_x＝（c_0，c_1，・・・，c_{nm-1}）と予め格納された数値“ａ”とを読み出し、逆変換されたデータc_x＝（c_0，c_1，・・・，c_{nm-1}）の先頭からａワードをヘッダｒとして規定し、逆変換されたデータｃｘを改めてc_x＝（ｒ（ヘッダ），c_x（残りのデータ））として記憶部１８０に記憶させる。
【００８８】
次に、ステップＳ３００において、制御部１９０は、記憶部１８０から逆一体化されたc_x＝（ｒ，c_x）を読み出し、第二擬似乱数生成部１４０に対して、ヘッダｒを初期値としたｎｍ－ａワードの擬似乱数列r_x＝（r_0，r_1，・・・，r_{nm-a-1}）を生成させる。
【００８９】
そして、ステップＳ３１０において、制御部１９０は、逆一体化された残りのデータc_x＝（c_{a+1}，c_{a+2}，・・・，c_{nm-a-1}）と生成された擬似乱数列r_x＝（r_0，r_1，・・・，r_{nm-a-1}）とを排他的論理和する（$x_{a+i}=c_{a+i}$ ＸＯＲ r_i（ｉ＝０，１，・・・，ｎｍ－ａ－１））ことにより、データｘ＝（x_a，x_{a+1}，・・・，x_{nm-1}）を算出し、次いで、データｘの先頭からｑワードを平文ｘのデータ長ｇなどの情報を含んだヘッダ情報ｕとして切り取り、さらに、残りのデータの先頭からｇワードのみを平文ｘとして復号し、これを記憶部１８０に記憶させる。
【００９０】
なお、最後に切り捨てられたｎｍ－ｑ－ｇワードのデータはパディングである。
【００９１】
そして最後に、ステップＳ３２０において、制御部１９０は、記憶部１８０から平文ｘを読み出し、出力部１７０に出力させて、全ての処理を終了する。
【００９２】
一方、ステップＳ２８０において、制御部１９０によりインクリメントされた値が所定のラウンド数ｈを越えていないと判定された場合には、ステップＳ２４０に戻って、インクリメントされた値が所定のラウンド回数ｈを越えるまで、ステップＳ２４０～Ｓ２６０までの処理を繰り返した後、すべての処理を終了する。
【００９３】
このように、復号装置１００は、暗号装置１０から送信されてきた暗号文ｆｃｘを復号するための情報として、秘密鍵としての再配置表Ｋ＝（ｋ［０］，ｋ［１］，・・・，ｋ［ｎ－１］）と、分割数ｎと、ヘッダの長さａと、ブロック差分ｄとを暗号装置１０と共有している。これにより、復号装置１００は、復号の過程で暗号文cf_xの本当の鍵とも言える擬似乱数列のヘッダｒが入手でき、平文ｘの長さｇも同様に入手できるので、擬似乱数列のパディングｐを切り捨てることができる。
【００９４】
［暗号装置と復号装置のその他の構成］
上記においては暗号装置と復号装置とを別体として説明したが、これはあくまで各装置の構成及び機能の説明を容易にするためである。上記の説明から明らかなように、本発明においても、従来の共通鍵方式と同様、暗号化処理と復号化処理とは可逆の関係にあるので、暗号装置と復号装置とを一体の構成とした装置に適用できることは明らかである。
【００９５】
［具体的な実装］
本発明に係る暗号装置及び復号装置のプログラム上の実装例を示す。ここでは、１ ワードを１バイトとし、疑似乱数のヘッダｒは２５６ビットを使用する。従って、ａ＝３２である。第一疑似乱数生成部３０で生成する疑似乱数としては、コンピュータプログラミング環境で使用できる疑似乱数を用いる。具体的には、暗号装置のシステムクロックや送信者による入力部からの入力のタイミングなどを使用して再現しにくい疑似乱数をヘッダｒとして使用する。また、分割数ｎはｎ＝２５６、平文のデータ長ｇなどの情報を含むヘ

ッダ情報ｕはｕ＝４、ブロック差分ｄ＝１とする。このとき、秘密鍵ＫはＫ＝（ｋ［０］，ｋ［１］，・・・，ｋ［２５５］）として表される再配置表であり、ｋ［ｉ］（ｉ＝０，１，・・・，２５５）には、０，１，・・・，２５５を並べ替えた値が格納されている。

【００９６】

平文ｘ、ヘッダ情報ｕ、疑似乱数のヘッダｒをあわせた全データが１０２４バイトの倍数になるように、平文ｘの末尾に適当な長さｖの疑似乱数をパディングする。そして、これを改めてｘ＝（x_0，x_1，・・・，x_{256m-1}）とする。ここで、x_i（ｉ＝０，１，・・・，２５５）は、ＬｏｎｇＷｏｒｄ（３２ビット符号なし整数）である。

【００９７】

なお、以降のプログラムの変数の中には、上記の実施例で使用した変数名が異なるものもあるが、混乱することはないはずである。

【００９８】

＜公開可能な疑似乱数生成部４０の実装例＞

```
d: array[0..7] of LongWord;
i; integer;
i:=0;
function g1: LongWord;
begin
  w.d:=d[i and 7]+d[(i-1) and 7];
  d[i and 7]:=w.d;
  w.b[0]:=k[w,b[0]]; w.b[1]:=k[w.b[1]];
  w.b[2]:=k[w,b[2]]; w.b[3]:=k[w.b[3]];
  g1:=w.d; //va;ue of g1
  i:=(i+1) and 7;
end;
```

ここで、ｗはＬｏｎｇＷｏｒｄ（３２ビット符号なし整数）ｄと４バイトの配列ｂ［０］，ｂ［１］，ｂ［３］の共用体である。

【００９９】

このような疑似乱数生成部４０から生成される疑似乱数r_xと平文の疑似乱数のヘッダｒ以降の部分を排他的論理和して暗号化したものを、ＬｏｎｇＷｏｒｄの配列として、改めてｘ＝（ｘ［０］，ｘ［１］，・・・，ｘ［ｖ－１］）（ｖ＝２５６ｍ）とする。ここで、ｍは、ブロックデータの長さをＬｏｎｇＷｏｒｄの個数で表したものである。

【０１００】

＜変換処理の実装例＞

```
  i; integer;
  begin
    for i:=0 to v-m-1 do
    begin
      x[i+m]:=x[i+m];x[i];
      w.d:=x[i+m];
      w.b[0]:=k[w,b[0]]; w.b[1]:=k[w.b[1]];
      w.b[2]:=k[w,b[2]]; w.b[3]:=k[w.b[3]];
      x[i+m]:=w.d;
    end;
    for i:=v-m to v-1 do
    begin
      x[i+m-v]:=x[i+m-v]+x[i];
      w.d:=x[i+m-v];
      w.b[0]:=k[w,b[0]]; w.b[1]:=k[w.b[1]];
```

```
      w.b[2]:=k[w,b[2]]; w.b[3]:=k[w.b[3]];
      x[i+m-v]:=w.d;
    end;
  end;
```

【０１０１】

この処理の後、ｘをｘ＝（y_0，y_1，・・・，y_{255}）と２５６分割する。ここで、y_i＝（ｘ［ｍｉ］，ｘ［ｍｉ＋１］，・・・，ｘ［ｍｉ＋ｍ－１］）（ｉ＝０，１，・・・，２５５）である。

【０１０２】

＜逆変換処理の実装例＞

まず、逆置換ｉｋを次のように計算しておく。

```
for i:=0 to 255 do
  ik[k[i]]:=i;
```

逆置換ｉｋを用いて逆変換を次のように計算する。

```
  i; integer;
  begin
    for i:=v-1 downto v-m do
    begin
      w.d:=x[i+m-v];
      w.b[0]:=k[w,b[0]]; w.b[1]:=k[w.b[1]];
      w.b[2]:=k[w,b[2]]; w.b[3]:=k[w.b[3]];
      x[i+m-v]:=w.d;
      x[i+m-v]:=x[i+m-v]-x[i];
    end;
    for i:=v-m-1 downto 0 do
    begin
      w.d:=x[i+m]
      w.b[0]:=k[w,b[0]]; w.b[1]:=k[w.b[1]];
      w.b[2]:=k[w,b[2]]; w.b[3]:=k[w.b[3]];
      x[i+m]:=w.d;
      x[i+m]:=x[i+m]-x[i];
    end;
  end;
```

【０１０３】

この処理の後、ｘをｘ＝（y_0，y_1，・・・，y_{255}）と２５６分割する。ここで、y_i＝（ｘ［ｍｉ］，ｘ［ｍｉ＋１］，・・・，ｘ［ｍｉ＋ｍ－１］）（ｉ＝０，１，・・・，２５５）である。

【０１０４】

＜再配置処理の実装例＞

ｘと同じ長さの配列ｙを準備し、以下のように再配置処理する。ここで、命令Ｍｏｖｅ（ｘ［ｉ］，ｙ［ｊ］，ｚ）は、ｘ［ｉ］のアドレスからｙ［ｊ］のアドレスへzバイトの記憶内容をコピーする処理を表す。この処理によって、配列ｘの内容を再配置表Ｋによってブロック単位で並べ替えることができる。

```
  i; integer;
  begin
    for i:=0 to 255 do
    begin
      Move(x[i*m], y[k[i]*m], m);
    end;
    Move(y[0], x[0], 4*v);
```

```
  end;
```

【０１０５】

なお、復号時に使用する逆再配置処理は以下のようにすればよい。

```
  i; integer;
  begin
    for i:=0 to 255 do
    begin
      Move(x[k[i]*m], y[i*m], m);
    end;
    Move(y[0], x[0], 4);
  end;
```

【０１０６】

この実装例は、１６８３ビットの鍵長のブロック暗号程度の安全性を持ち、ＡＥＳの１０倍程度の暗号化の実行速度が達成できる。

【０１０７】

［その他の実施例］

上記の実施例では、第二擬似乱数生成部４０を用いたストリーム暗号によって高速な暗号装置１０をデザインした。その他の実施例として、ストリーム暗号をブロック暗号に変えた構成を有する暗号装置が考えられる。

【０１０８】

図１１は、その他の暗号装置の概略的な構成を示したブロック図である。暗号装置２００は、入力部２２０と、第一擬似乱数生成部２３０と、ブロック暗号文生成部２９０と、変換処理部２５０と、再配置処理部２６０と、出力部２７０と、記憶部２８０と、制御部３００とを備える。このうち、記憶部２８０と制御部３００とを除く部分を暗号文生成部２００Ａと称することにする。

【０１０９】

このうち、入力部２２０と、第一擬似乱数生成部２３０と、変換処理部２５０と、再配置処理部２６０と、出力部２７０とは、それぞれ、入力部２０と、第一擬似乱数生成部３０と、変換処理部５０と、再配置処理部６０と、出力部７０と同様の機能を有するので、説明を省略する。

【０１１０】

記憶部２８０は、入力部２２０～出力部２７０から成る暗号生成部２００Ａが生成した各種のデータの格納を行うサブメモリと、後述する暗号化処理の各ステップを実行するためのコンピュータに読み取り可能な暗号プログラムを格納するメインメモリとから構成される。

【０１１１】

また、制御部３００は、記憶部２８０から読み出した暗号プログラムに従って、入力部２２０～記憶部２８０を制御するＣＰＵを備える。

【０１１２】

なお、記憶部２８０及び制御部３００のハードとして構成は、上記実施例と同様の構成なので、説明を省略する。

【０１１３】

［暗号化処理］

以上を前提として、図１１に示した暗号装置２００によって行われる再配置暗号について上記の実施例と異なる部分のみ説明する。図１２は、図１１に示した暗号装置２００によって行われる再配置暗号の処理手順を示した概念図である。

【０１１４】

暗号装置２００は、上記の実施例におけるストリーム暗号の代わりにブロック暗号を用いて本発明の再配置暗号を実現するものである。

【０１１５】

従って、ブロック暗号文生成部２９０において平文ｘをブロック暗号化する際に、平文ｘの長さがブロック長の倍数とならない場合には平文の最後にパディングをする必要が生じる。このとき、制御部３００は、ｖ≡－ａ－ｇ－ｑ（ｍｏｄ　ｎ）となるような最小の非負整数ｖを算出し、パディングｐとしてｖワードの長さの擬似乱数列ｚを第一擬似乱数生成部２３０に生成させ、平文ｘの最後に付加する処理を行う。そして、平文ｘのデータ長ｇなどの情報を含んだヘッダ情報ｕ（長さ：ｑワード）、平文ｘ、パディングｑを合わせたデータを改めて平文ｘ＝（ｘ、ｚ）とした後、ブロック暗号化のステップへと処理を進める。この場合、再配置処理において分割される各ブロックの長さを表す整数“ｍ”は、ｍ＝（ａ＋ｑ＋ｇ＋ｐ）／ｎとして算出される。

【０１１６】

次のステップとして、制御部３００は、記憶部２８０から平文ｘ（ｇワード）、平文ｘのデータ長などの情報を含んだヘッダ情報（ｑワード）、パディング（ｐワード）を読み出し、ブロック暗号文生成部２９０に対して、これらのデータを第一の擬似乱数列ｒを鍵として、公知のブロック暗号化を行わせる。

【０１１７】

以降の処理は、ストリーム暗号を用いた上記の実施例と同様なので、説明を省略する。

【０１１８】

このように、本発明の暗号をブロック暗号に用いた場合、ストリーム暗号に用いた場合よりも暗号化の実行速度は遅くなるが、その代わりに、従来使用されているＡＥＳなどの共通鍵方式を用いた暗号装置への実装が容易であるといった利点がある。

【０１１９】

［再配置暗号のＮＰ完全性］

現在、世界標準の暗号として公開鍵暗号が広く採用されている。公開鍵暗号は、巨大数の素因数分解が現在のコンピュータ（ノイマン型コンピュータ）の能力では現実的な時間では行えないこと(素因数分解問題)などを安全性の根拠としている。しかし、近年急速に研究開発が進められている量子コンピュータを使うと、公開鍵暗号を解くために必要な素因数分解問題と離散対数問題を高速に解くことができることが証明されている（Peter W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", In Proceeding of 35th IEEE FOCS, pp.124-134, Santa Fe, NM, Nov 20-22, 1994.）。このことは、将来、量子コンピュータが実用化されると、公開鍵暗号は、標準的な暗号方式としては実質的に使用できなくなることを意味する。

【０１２０】

しかし、当業者の間では、ＮＰ完全性を有する問題であれば量子コンピュータでも解く事ができないと考えられている。これに関し、本発明の再配置暗号による暗号文の解読問題はＮＰ完全であることが以下のようにして示される。

【０１２１】

証明に当たって、まず次の３つの問題を設定する。

【０１２２】

（Ｐ１）ナップサック問題（部分和問題）Z_0：決定問題としてのナップサック問題とは、容量ＣのナップサックとN個の品物A_i（容量c_i，価値v_i）がある場合（$i=1, 2, \cdots, N$）に、このナップサックに詰め込める品物の組み合わせの中で価値の総計が所定値Ｖとなる組み合わせがあるか否かを判定する問題である。ここで、Ｃ，Ｎ，c_i，v_i，Ｖはすべて自然数である。この問題において、すべての品物について$c_i=v_i$が成り立つ場合、これを部分和問題といい、以下のように定式化できる。

＜部分和問題＞

与えられた自然数$x_1, x_2, \cdots, x_N$，ｙに対し、ある部分集合$I \subset \{1, 2, \cdots, N\}$が存在し、$y=\Sigma_{i \in I} x_i$とできるか？

【０１２３】

（Ｐ２）和ジグソーパズル問題Z_1：和ジグソーパズル問題と称する問題を新たに設定する。

＜和ジグソーパズル問題＞
与えられた自然数x_1，x_2，・・・，x_N，yに対し、ある置換（Ｓ－ＢＯＸ）$s \in S_N$と自然数mが存在し、$y = \Sigma_{i=I}^{m} x_{s(i)}$とできるか？
【０１２４】
（Ｐ３）再配置暗号ジグソーパズル問題Z_2：再配置暗号ジグソーパズル問題と称する問題を新たに設定する。
＜再配置暗号ジグソーパズル問題＞
与えられた自然数の配列$X = (x_1, x_2, \cdots, x_N)$と自然数（平文）Wに対して、再配置暗号のある秘密鍵K（Ｓ－ＢＯＸ又は再配置表）が存在し、$D_K(X) = W$とできるか？ここで、D_Kは秘密鍵Kを用いた再配置暗号の復号化関数である。
【０１２５】
このとき、本願発明の再配置暗号による暗号文の解読問題がＮＰ完全であることは、上記の３つの問題を多項式時間に帰着させることで次のように証明される。
【０１２６】
まず、ナップサック問題（部分和問題）Z_0はＮＰ完全であることが既に知られている。そして、明らかに、Z_0はZ_1に多項式時間帰着できる。つまり、

$$Z_0 <_p Z_1$$

。
【０１２７】
次に、関数$f(X, y, s, m) = (E_s(X, y, m), (X, \Sigma_{i=1}^{m} x_{s(i)}, m), s)$を定義する。ここで、$E_s$は、sを用いた再配置暗号の暗号化関数である。関数fは、再配置関数sと暗号化関数E_sで計算されるわけだが、その計算時間はO（n）程度であるので、Z_1はZ_2に多項式時間帰着されることになる。つまり、関数fは多項式時間計算可能関数であり、

$$Z_1 <_p Z_2$$

。ここで、$X = (x_1, x_2, \cdots, x_N)$へのsの作用を$s(X) = (x_{s(1)}, x_{s(2)}, \cdots, x_{s(N)})$とし、nを入力のサイズ（バイト数）とする（$N \ll n$）。
【０１２８】
以上のことより、Z_0，Z_1，Z_2の各問題は、

$$Z_0 <_p Z_1 <_p Z_2$$

の順番に多項式時間帰着させることできることがわかる。上述したように、Z_0及びZ_1はＮＰ完全であるので、結論として、Z_2もＮＰ完全であることが示されたことになる。
【０１２９】
このことより、統計的に偏りのない擬似乱数発生器を用いた場合、本発明の再配置暗号は量子コンピュータに対する耐性を持つと考えられる。実際に使用される擬似乱数発生器は統計的な偏りを持つわけだが、例えば、２５６個に分割された再配置暗号のブロックの順番を統計的に割り出すことは困難であると考えられる。
【０１３０】
［再配置暗号を用いたメッセージ認証］
上記で説明した実施形態においては、暗号文が改竄された場合、復号した平文から改竄を検知する機能は高くない。しかし、本暗号をハッシュ関数と組み合わせて使用することにより、この機能を補完することができる。
【０１３１】
これに関し、上述したように公開鍵暗号が量子コンピュータによって攻撃されるのと同様に、メッセージ認証に使われる鍵付きハッシュ関数（ＭＡＣ：Message Authentication Code）も攻撃されると考えられる。

【０１３２】
これに対抗するために、本発明においては、次に示すように幾つかの応用例が考えられる。
【０１３３】
＜応用例１＞
ｈをＭＤ５やＳＨＡ１などのハッシュ関数とし、送信するメッセージｍをｎバイトの配列Ｘ＝（x_0，x_1，・・・，x_{n-1}）に格納することを考える。このとき、次の手順で認証機能付きメッセージＭを作成する。
【０１３４】
（Ｓ１）再配置暗号の秘密鍵Ｋ（Ｓ－ＢＯＸ又は再配置表）を用いて、配列Ｘ＝（x_0，x_1，・・・，x_{n-1}）を（Ｋ［x_0］，Ｋ［x_1］，・・・，Ｋ［x_{n-1}］）と置換する。
【０１３５】
（Ｓ２）$m' = E_K(h(X)) = h_K(m)$を計算する。
【０１３６】
（Ｓ３）オリジナルのメッセージｍに対して、認証機能付きメッセージＭをＭ＝（ｍ，ｍ’）とする。
【０１３７】
このようにして作成された認証機能付きメッセージＭが、共通鍵として秘密鍵Ｋを共有する通信において、攻撃者によって改竄されたか否かを、次のようにして確認することができる。
【０１３８】
メッセージＭを受信した受信者は、ｈ（Ｘ）と$D_K(m')$を計算して、それらが一致するか否かを調べる。このとき、一致すればメッセージＭは改竄されていないことが分かる。
【０１３９】
この応用例の利点として、通常、ハッシュ関数は必ず衝突があるので、小さい確率ではあるが，メッセージを改竄することができる。しかし、本応用例では、オリジナルのメッセージｍ及び秘密鍵Ｋが同じでも、それから作成されたｍ’は無数にあるので、攻撃者はメッセージの改竄に成功したか否かを確認することはできない。
【０１４０】
＜応用例２＞
応用例１の方向とは異なり、本発明の再配置暗号を利用して全く新しいＭＡＣを作成することも可能である。いま、再配置暗号の秘密鍵Ｋを、０，１，２，・・・，２５５を並べ替えた２５６バイトのＳ－ＢＯＸとする。そして、メッセージｍをｎバイトの配列Ｘ＝（x_0，x_1，・・・，x_{n-1}）に格納することを考える。このとき、次の手順で認証機能付きメッセージＭを作成する。
【０１４１】
（Ｓ１）ｉ＝０，ｊ＝０とする。
【０１４２】
（Ｓ２）秘密鍵Ｋを用いて、配列Ｘ＝（x_0，x_1，・・・，x_{n-1}）を（Ｋ［x_0］，Ｋ［x_1］，・・・，Ｋ［x_{n-1}］）と置換する。
【０１４３】
（Ｓ３）$x_{(j+1) \bmod n} = x_{(j+i) \bmod n} + x_j$とし、その後、ｊをインクリメントする。
【０１４４】
（Ｓ４）ｊ＜ｎならば、（Ｓ３）へ行く。
【０１４５】
（Ｓ５）ｉをインクリメントする。
【０１４６】

（Ｓ６）ｉ＜Ｒならば、ｊ＝０として（Ｓ３）へ行く。ここで、Ｒは攪拌の回数を表す所定値であり、通常３回ぐらいに設定する。

【０１４７】

（Ｓ７）ｍ’＝E_K（x_0，x_1，・・・，$x_{n'-1}$）＝h_K（ｍ）を計算する。

【０１４８】

ここで、ｎ’はｎよりも小さい自然数である。つまり、本応用例においては、一体化されたデータの一部を暗号化し、ハッシュ表として使用するので、このハッシュ表は平文の長さよりもずっと短い。

【０１４９】

この場合も、秘密鍵Ｋを共有する通信で、メッセージＭ＝（ｍ，ｍ’）を受信した受信者は、（Ｓ１）～（Ｓ６）を実行し、（x_0，x_1，・・・，x_{n-1}）とD_K（ｍ’）が一致するか否かを調べることでメッセージ認証を行うことができる。この応用例はかなり高速で行うことができ、やはり、攻撃者がメッセージの改竄に成功したか否かを確認することはできない。

【０１５０】

＜応用例３＞

また、別の応用例として、メッセージの受信者は、ランダムな再配置暗号の秘密鍵Ｋ’を用いて、次の手順に従うことによって、送信者が送信内容を否認できないメッセージ認証を行うことができる。

【０１５１】

（Ｓ１）送信者は、＜応用例２＞のＭＡＣを計算し、Ｍ＝（ｍ，ｍ’）を受信者に送信する。

【０１５２】

（Ｓ２）受信者は、ランダムな再配置暗号の秘密鍵Ｋ’を作成し、ｙ＝E_K（ｍ，$h_{K'}$（ｍ））を作成して、送信者に送り返す。

【０１５３】

（Ｓ３）送信者は、D_K（ｙ）＝（ｍ，$h_{K'}$（ｍ））を計算し、正当な通信であることを確認した後、ｍ”＝E_K（$h_{K'}$（ｍ））を計算し、（ｍ，ｍ”）を受信者に送り返す。

【０１５４】

（Ｓ４）受信者は、D_K（ｍ”）を計算し、$h_{K'}$（ｍ）と一致するか否かを調べ、一致した場合には（ｍ，Ｋ’，ｍ”）を保存する。

【０１５５】

この応用例におけるＭＡＣの衝突は秘密鍵Ｋ’に依存する。送信者は、Ｋ’を知らないので、ｍ”の衝突を計算できない。従って、送信者は、メッセージｍの送信を否認できない。もちろん、第三者がこの通信内容を改竄することもできない。

【０１５６】

以上、幾つかの例を挙げて説明したように、本発明の再配置暗号を用いると、量子コンピュータによっても攻撃されないメッセージ認証を行うことができるので、これを電子決済の基本として電子商取引システムに適用すれば、将来的に量子コンピュータに対して耐性を有する電子商取引システムを開発することができる。その意味において、本発明の再配置暗号は、今後さらに活発化するであろう電子商取引システムの基盤と十分に成り得る暗号方式であると考えられる。

【０１５７】

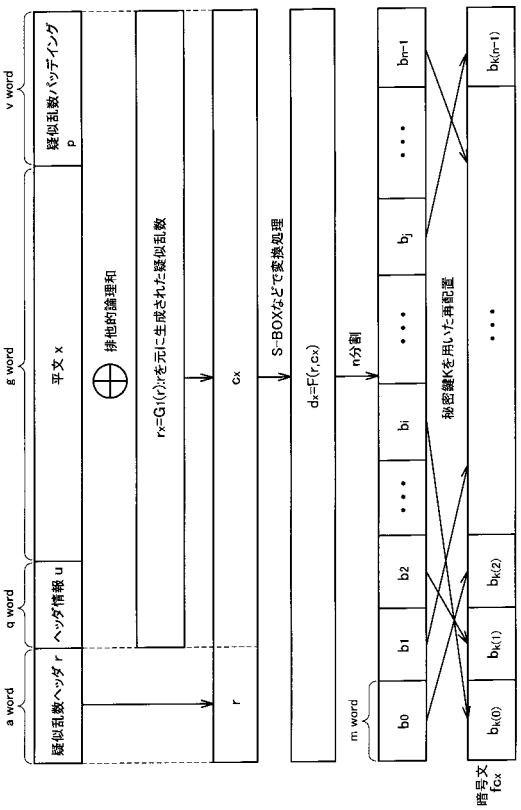
なお、本発明は、上述した実施形態に限定されるものではなく、その要旨を逸脱しない範囲で構成要素を変形して具現化することができる。また、上記した実施形態に開示された構成要素を適宜組み合わせることにより各種の発明を形成することができる。例えば、各実施形態に示される構成要素から幾つかの構成要素を削除してもよい。また、ある実施形態に示される構成要素を別の実施形態に追加してもよい。さらに、異なる実施形態に亘って構成要素を適宜組み合わせてもよい。

【産業上の利用可能性】

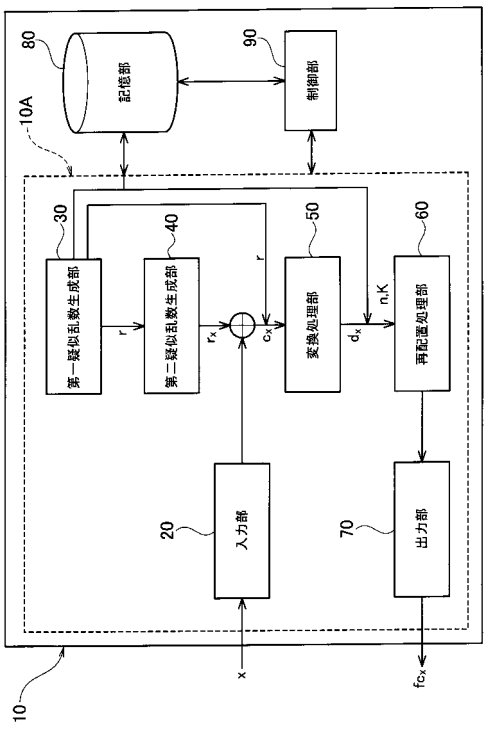
【０１５８】

本発明によれば、既知平文攻撃を極めて困難にすると共に、暗号化の実行速度を極めて速くすることができる暗号装置及び暗号プログラム、これらの暗号装置及び暗号プログラムによって作成された暗号文を復号化する復号装置及び復号プログラム、及びこれらのプログラムを記憶する記憶媒体を提供することができる。

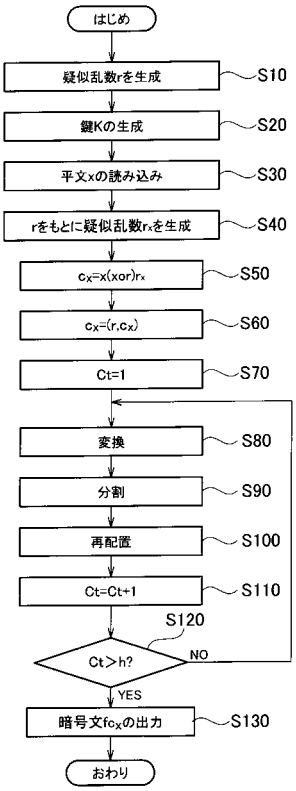
【図１】



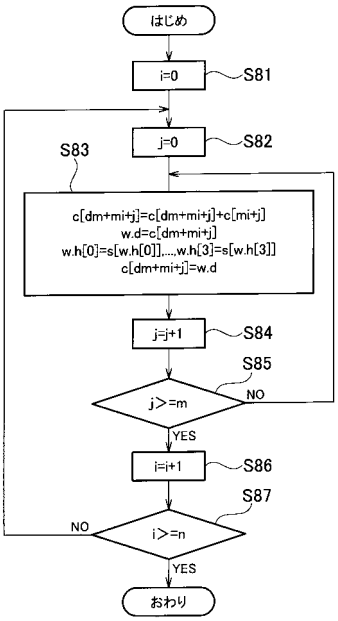
【図２】



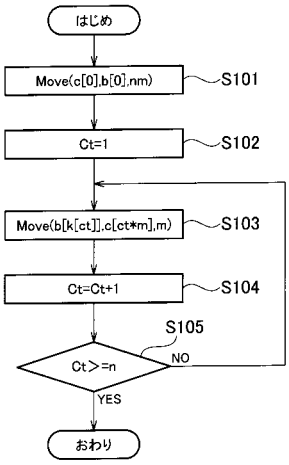
【図３】


はじめ
疑似乱数rを生成
S10
鍵Kの生成
S20
平文xの読み込み
S30
rをもとに疑似乱数r_xを生成
S40
$c_x=x(xor)r_x$
S50
$c_x=(r,c_x)$
S60
Ct=1
S70
変換
S80
分割
S90
再配置
S100
Ct=Ct+1
S110
S120
Ct＞h?
NO
YES
暗号文fc_xの出力
S130
おわり

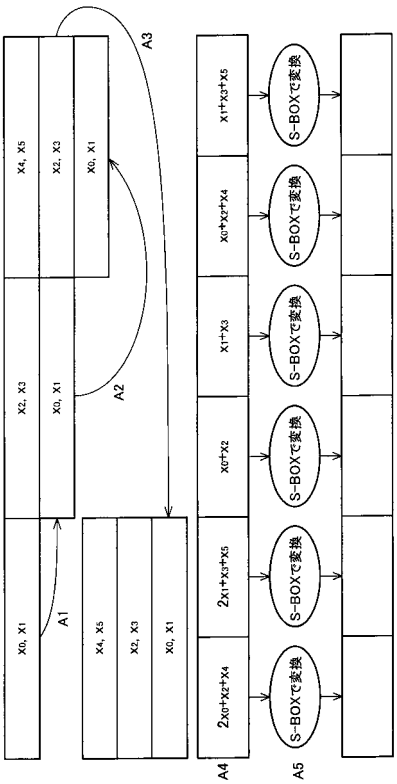
【図４】


はじめ
i=0
S81
j=0
S82
S83
c[dm+mi+j]=c[dm+mi+j]+c[mi+j]
w.d=c[dm+mi+j]
w.h[0]=s[w.h[0]],...,w.h[3]=s[w.h[3]]
c[dm+mi+j]=w.d
j=j+1
S84
S85
j＞=m
NO
YES
i=i+1
S86
S87
NO
i＞=n
YES
おわり

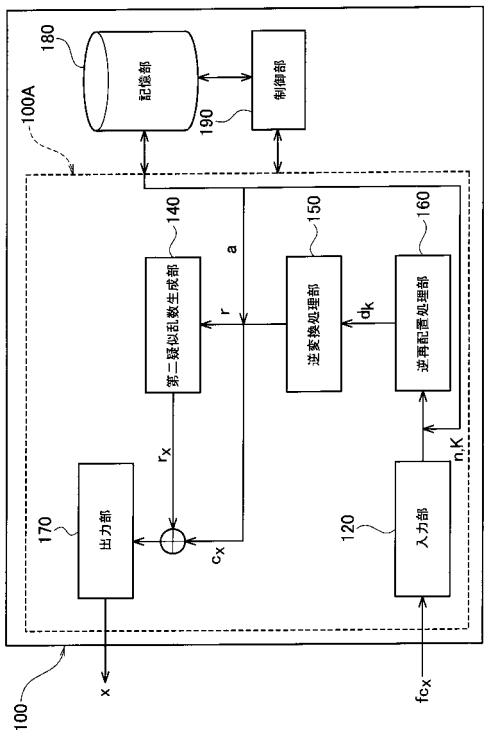
【図５】


はじめ
Move(c[0],b[0],nm)
S101
Ct=1
S102
Move(b[k[ct]],c[ct*m],m)
S103
Ct=Ct+1
S104
S105
Ct＞=n
NO
YES
おわり

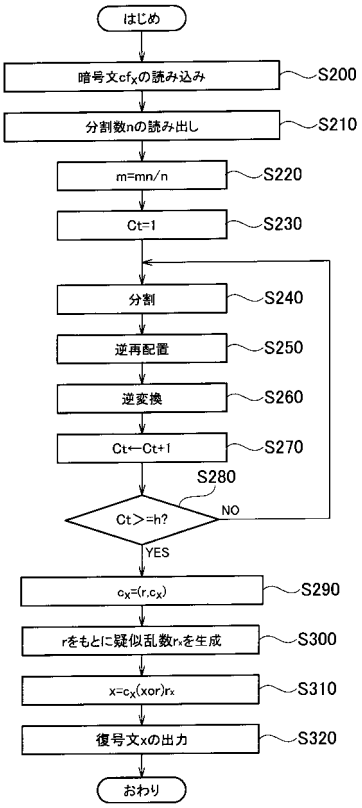
【図６】


A3
A2
A1
A4
A5
S-BOXで変換

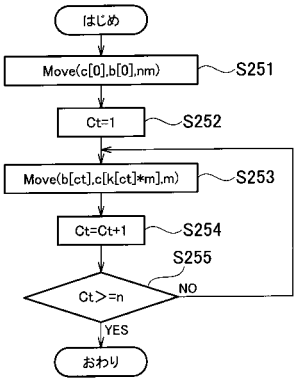
【図７】



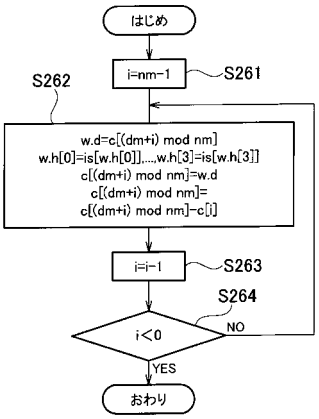
【図８】



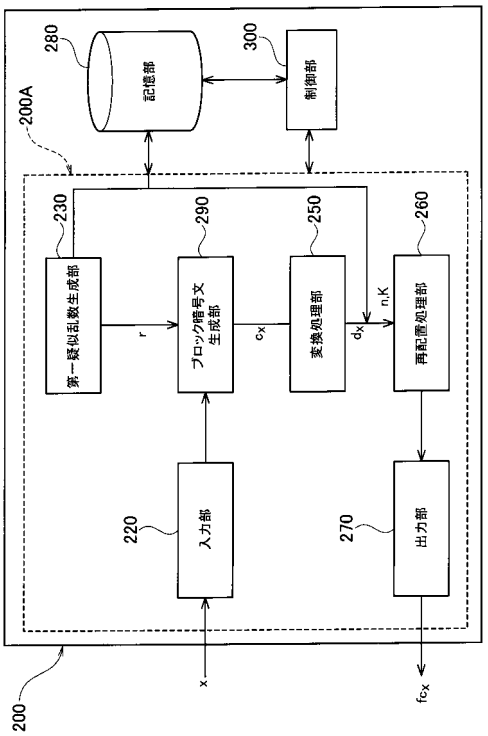
【図９】



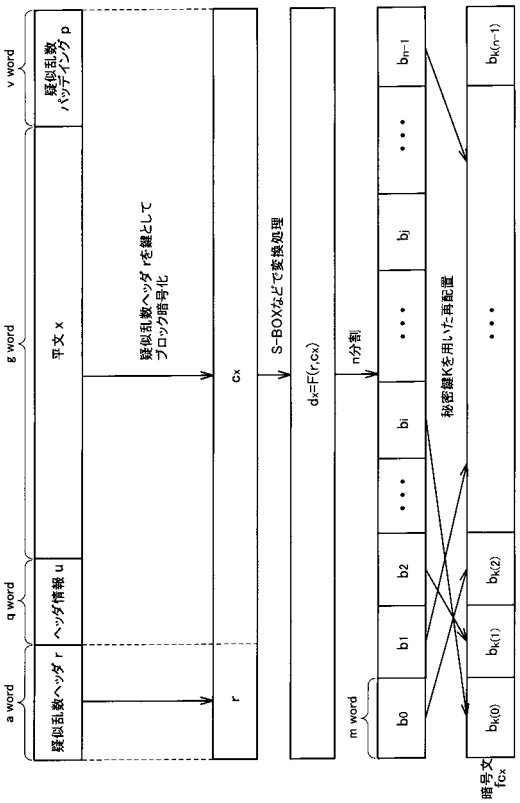
【図１０】



【図１１】



【図１２】


a word
q word
g word
v word
疑似乱数ヘッダ r
ヘッダ情報 u
平文 x
疑似乱数パッディング p
疑似乱数ヘッダ r を鍵として
ブロック暗号化
r
cx
S-BOXなどで変換処理
dx=F(r,cx)
n分割
m word
b0
b1
b2
bi
bj
bm-1
秘密鍵Kを用いた再配置
bK(0)
bK(1)
bK(2)
bK(n-1)
暗号文
fcx

フロントページの続き

(56)参考文献 特開２００８－１２４９３６（ＪＰ，Ａ）
特開２００６－１９１６２６（ＪＰ，Ａ）
特開平１０－１７３６４６（ＪＰ，Ａ）
特開昭６２－２３７８３４（ＪＰ，Ａ）
特開昭６２－８１１４５（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G09C 1/00