(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2008 年1 月24 日 (24.01.2008)

PCT

(10) 国際公開番号
WO 2008/010441 A1

(51) 国際特許分類:
H04L 9/22 (2006.01)

(21) 国際出願番号: PCT/JP2007/063797

(22) 国際出願日: 2007 年7 月11 日 (11.07.2007)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2006-199681 2006 年7 月21 日 (21.07.2006) JP

(71) 出願人 (米国を除く全ての指定国について): 日本電気株式会社 (NEC CORPORATION) [JP/JP]; 〒1088001 東京都港区芝五丁目７番１号 Tokyo (JP). 北陸日本電気ソフトウェア株式会社 (NEC SOFTWARE HOKURIKU, LTD.) [JP/JP]; 〒9202141 石川県白山市安養寺１番地 Ishikawa (JP).

(72) 発明者; および

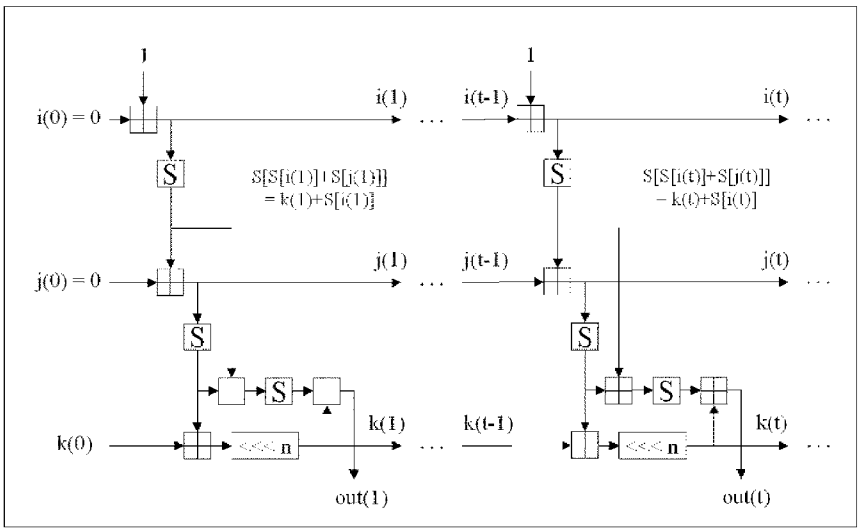
(75) 発明者/出願人 (米国についてのみ): 角尾 幸保 (TSUNOO, Yukiyasu) [JP/JP]; 〒1088001 東京都港区芝五丁目７番１号 日本電気株式会社内 Tokyo (JP). 齊藤 照夫 (SAITO, Teruo) [JP/JP]; 〒9202141 石川県白山市安養寺１番地 北陸日本電気ソフトウェア株式会社内 Ishikawa (JP). 久保 博靖 (KUBO, Hiroyasu) [JP/JP]; 〒9202141 石川県白山市安養寺１番地 北陸日本電気ソフトウェア株式会社内 Ishikawa (JP). 洲崎 智保 (SUZAKI, Tomoyasu) [JP/JP]; 〒9202141 石川県白山市安養寺１番地 北陸日本電気ソフトウェア株式会社内 Ishikawa (JP).

(74) 代理人: 加藤 朝道 (KATO, Asamichi); 〒2220033 神奈川県横浜市港北区新横浜３丁目２０番１２号 ダヴィンチ望星７階 加藤内外特許事務所 Kanagawa (JP).

/続葉有/

(54) Title: ENCRYPTION DEVICE, PROGRAM, AND METHOD

(54) 発明の名称: 暗号装置及びプログラムと方法



(57) Abstract: Provided is an encryption device having a high safety for confidential data when performing data communication and accumulation.The encryption device generates a pseudo-random number according to a secret key and applies the pseudo-random number sequence to a plain text so as to generate an encrypted text. An internal state used for generating the pseudo-random number sequence is an internal state based on the state in accordance with rearrangement of a finite number of numeric values. At least one of the temporary variable used for generation of the pseudo-random number sequence is a temporary variable using the result of execution of a predetermined leftward or rightward rotate shift and depending on a number smaller than the number of internal state in accordance with the result of the linear or non-linear, or a combination of linear and non-linear using one or more numeric values of the internal state. The pseudo-random number generated is generated by performing calculation by using one or more numeric values among the internal states and the temporary variable.

(57) 要約: データの通信や蓄積の際にデータを秘匿するための安全性の高い暗号装置を提供する。秘密鍵に基づき擬似乱数列を生成し、前記擬似乱数列を平文に作用させることにより暗号文を生成する暗号装置であって、前記擬似乱数列の生成に用いる内

/続葉有/

WO 2008/010441 A1

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書

２文字コード及び他の略語については、定期発行される各*PCT*ガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、前記擬似乱数列の生成に用いる一時変数の少なくとも一つが、前記内部状態のうちの一つ、又は、複数個の数値を使用した、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数であり、前記生成される擬似乱数を、前記内部状態のうちの１つ又は複数個の数値と、前記一時変数との演算によって生成する。

明 細 書

暗号装置及びプログラムと方法

技術分野

[0001] [関連出願の記載]

本発明は、日本国特許出願:特願2006－199681号(平成18年7月21日出願)の優先権主張に基づくものであり、同出願の全記載内容は引用をもって本書に組み込み記載されているものとする。

本発明は、データの通信や蓄積の際にデータを秘匿するための暗号装置及びコンピュータ・プログラムと方法に関する。

背景技術

[0002] ＜ストリーム暗号概説＞

データを秘匿するための手法に暗号がある。暗号の中で、高速な暗号化及び復号を行うために、平文をビット単位あるいはバイト単位などで逐次、暗号化するストリーム暗号がある。典型的なストリーム暗号は、鍵ストリームを生成する鍵ストリーム生成部と、鍵ストリームと平文を結合する結合部と、からなる。例えば、暗号処理では、秘密鍵をシードとして擬似乱数を生成し(鍵ストリーム生成部)、その擬似乱数と平文をXORし(結合部)、暗号文を生成する。ここで、XORはビット毎の排他的論理和を意味する。

[0003] 平文をP、擬似乱数をR、暗号文をCとすると、

P XOR R = C

の関係となる。

[0004] 復号処理では、同じシードから擬似乱数を生成して暗号文とXORすることで、平文を得ることができる。

[0005] C XOR R = (P XOR R) XOR R

= P XOR (R XOR R)

= P XOR 0

= P

の関係となり、

C XOR R = P

が得られる。

[0006] ＜ストリーム暗号の安全性＞

ストリーム暗号は、結合部がXORなど簡単な処理で実現されることが多い。そのためストリーム暗号の安全性は、鍵ストリーム生成部が生成する擬似乱数の安全性に依存する。

[0007] ここでいう、擬似乱数の安全性は、観測済みの擬似乱数列から、今後発生する擬似乱数列が予測できないことに依存している。

[0008] 例えば、鍵ストリーム生成部が、a, b, c, a, b, c, ... のように繰り返す鍵ストリームを生成したならば、観測済みの擬似乱数から、今後発生する擬似乱数の予測がつくために、暗号文が解読されてしまう。それは、結合部の逆演算を実行できるからである。

[0009] すなわち、擬似乱数Rの予測がつけば、暗号文Cを観測して、

C XOR R = P

により、平文Pを得ることができる。

[0010] 前述のように考えると、鍵ストリーム生成部が生成する擬似乱数が予測不可能な乱数（真の乱数）を生成するならば、そのストリーム暗号は安全であるといえる。逆にいえば、鍵ストリーム生成部が生成する擬似乱数が真の乱数と区別できるような情報を見つけることができれば、そのストリーム暗号は何がしかの安全性が低下していると考えることが出来る。

[0011] ＜ストリーム暗号の安全性評価手法の一例＞

前述ような考え方に基づいて、ストリーム暗号の安全性を評価する手法がある。出力された暗号文や擬似乱数列が、真の乱数列と区別がつくことを示す暗号の攻撃手法を、「Distinguishing Attack」とよぶ。

[0012] 真の乱数との区別は、出力された暗号文や擬似乱数列が、何がしかの偏りや、特徴を持っていることが示せれば、区別できたと判断する。

[0013] Distinguishing Attackでは、このような偏りや特徴を示す手段を、「Distinguisher 」と呼び、このような偏りや特徴を示す手段を発見したり作成したりすることを、「Distinguisherを構成する」と言う。Distinguisher が構成できるならば、Distinguishing Attackが

適用できることになる。

[0014] ある暗号に対して、Distinguishing Attackが適用できるならば、その暗号は平文または鍵の情報を漏らしている可能性があるので、安全な暗号であると保証することが出来ない。

[0015] 従って、Distinguishing Attackが適用できる暗号に修正を加え、Distinguishing Attackが適用出来なくなれば、暗号の安全性が向上したと考えることが出来る。

[0016] ＜ストリーム暗号の具体例＞

RC4は、Ron Rivest氏によって開発された暗号方式であり、RFC2246(TLS), WEP, WPA など暗号標準としてよく利用されるストリーム暗号である。RC4の仕様は、RFC2246(TLS)などで公開されていると言ってよい。

[0017] RC4は、処理単位nビットが可変という特徴を持ちながらも、処理単位nを大きくするとメモリが2n必要となり、鍵スケジュールも極端に低速になる。

[0018] そのため、現実的に、nは8ビットを越える実装は少なく、32ビットを越える実装は不可能であった。

[0019] よって、RC4は、32bit/64bitプロセッサといった最近のプロセッサ・サイズに適した実装が出来なかった。

[0020] 32ビットRC4ではこれらの制限が生じないよう、32bit/64bitプロセッサで高速かつ小メモリで実装可能なアルゴリズムに改良された。2005年にG. Gongらが発表した論文（非特許文献1）である。

[0021] 非特許文献1の論文では、処理単位が32bitの場合、速度は、RC4の約3.1倍高速、メモリはRC4の約2^{-22}倍に抑えることに成功している。

[0022] また、内部変数kを追加することにより、過去に報告されたRC4の脆弱性（統計的な偏り）が生じないアルゴリズムに改良されている。

[0023] ＜発明が対象とする攻撃手法の要点＞

G. Gongらのアルゴリズムの場合、連続する出力の最下位ビットが必ず一致するというdistinguisherの構成が可能となる。

[0024] このdistinguisherにより、約2^{30}のデータ量で真の乱数系列と区別可能となる。

[0025] ＜攻撃手法の例示のための、G. Gongら提案した改良アルゴリズムの説明＞

図2は、非特許文献1において、G. Gongらによって提案されたRC4の改良アルゴリズム(32bit RC4)を示す図である。G. Gongらによって提案されたRC4型ストリーム暗号は、配列Sのエントリ数が2^{n}個、配列Sのエントリサイズがmビットである。

[0026] また、非特許文献1の提案では、n=8, m=32のモデルについてのみKSAの初期定数aiを定義しているので、本明細書における解析でも、n=8, m=32のモデルについて、詳しく解析を行う。以下では、基本処理単位n, mを元に便宜上GGHN(n,m)と記述する。

[0027] 図2に示されるように、G. Gongらによって提案されたRC4型ストリーム暗号GGHN(n, m)は、KSA(K,S)とPRGA(S)の二つの処理から構成される。

[0028] KSA(K,S)は、いわゆる初期設定であり、40ビットから256ビットの鍵Kを基にして、32ビット256個の配列を並べ替え、初期状態Sを作り出す処理を行う。

[0029] PRGA(S)は、鍵ストーリームを生成する処理であり、状態Sに基づいて毎時刻、擬似乱数を生成する。

[0030] ここで、+はmod N上またはmod M上の算術加算を表し、N=2^{8}、M=2^{32}である。またLは秘密鍵のバイト数を表している。

[0031] まず、KSA(K,S)の動作を説明する。

[0032] KSAでは、配列Sの初期値として、初期変数aiを代入し(S[i]=a_i)、Sエントリどうしのswap(Swap[S[i],S[j]])と算術加算(S[i]＝S[i]＋S[j] mod M)を繰り返すことで、Sエントリを攪拌している。

[0033] KSAにおいて、内部変数kもSエントリ(k＝k＋S[i] mod M)を使って初期化されるため、PRGAにおけるkの初期値は未知である。

[0034] Sエントリの攪拌において、ループ回数rが可変であるが、Sエントリの出現確率がランダムになるように、m=32のときr=20を選択するよう定められている。G. Gongらによる提案では、m=64のときは、r=40を設定するよう定められている。

[0035] KSA(K,S)が終了した直後でPRGA(S)が開始されていない状態を時刻t＝0とする。時刻t＝0のとき、KSA(K,S)の動作が終了し、秘密鍵Kによって配列Sの状態が十分攪拌されていることが期待されている。

[0036] 次に、PRGA(S)の動作を説明する。

[0037] PRGAでは、インデックスi、jに基づく配列Sの参照結果(S[(S[i]+S[j])mod N])と、変数kの算術加算を行って、1ワード(1ワード=32ビット)を鍵ストリーム(out=S[(S[i]+S[j]) mod N]+k) mod M)として出力する。また鍵ストリームを生成するために参照したSエントリ(S[(S[i]+S[j])mod N])を、鍵ストリーム出力直後にkを用いて更新している(S[(S[i]+S[j])mod N]=k+S[i] mod M)。図2における out(=S[(S[i]+S[j])mod N]=k+S[i] mod M)が出力となる鍵ストリームである。

[0038] 図3では、時刻t＝1における動作(PRGAの状態遷移)を示している。配列Sにおいて、アドレス1の値S[1]がA、アドレスAの値S[A]がBの時に、アドレスA+Bの値S[A＋B]は、k0+A+Bになる。

[0039] 図4では、時刻t＝2における動作(PRGAの状態遷移)を示している。配列Sにおいて、アドレス1の値S[1]がA、アドレスAの値S[A]がB、アドレス2の値S[2]がC、アドレスA+Cの値S[A+C]がD、アドレスA+Bの値S[A+B]がk0+A+B、の時に、アドレスC+Dの値S[C+D]が、k0+B+C+Dになる。

[0040] G. Gongらによって提案されたRC4型ストリーム暗号の安全性は、彼らの提案論文(非特許文献1)のなかでで報告されている。

[0041] それによれば、鍵ストリームはSエントリに変数kを算術加算することでマスクされるため、KSAによってkが一様分布に従うとするならば出力系列に偏りが生じないことを示している。

[0042] また内部メモリのサイズはRC4の4倍となり、Sエントリが算術加算によって更新されるため、内部メモリを求める攻撃に対しても安全性が向上していると報告されている。

[0043] ただし、全てのS-boxエントリ(配列Sの要素)と変数kが同時に偶数になると、以降常に偶数が続くという「weak state」が存在する。しかし、内部メモリのサイズからみて、weak stateの存在確率は起こりえないくらいに十分小さいので、安全性に問題がないとされている。

[0044] 非特許文献1:G. Gong, K.C. Gupta, M. Hell, and Y. Nawaz,"Towards a General RC4-Like Keystream Generator,"SKLOIS Conference on Information Security and Cryptology, CISC 2005,LNCS 3822, pp.162-174, Springer Verlag, 2005.

非特許文献2:I. Mantin, and A. Shamir: "A Practical Attack on Broadcast RC4,"Fas

t Software Encryption, FSE 2001, LNCS 2355, pp.152-164,Springer-Verlag, 2001
非特許文献3:S. Paul, B. Preneel, and G. Sekar: "Distinguishing Attacks on the StreamCipher Py," eSTREAM, the ECRYPT Stream Cipher Project, Report 2005/081, 2005.

発明の開示

発明が解決しようとする課題

[0045] 非特許文献1～3の開示事項は、本書に引用をもって繰り込み記載されているものとする。以下の分析は、本発明によって与えられたものである。

[0046] ＜発明が対象とするGGHN(n,m)攻撃手法 Distinguishing Attackの説明＞

GGHN(8,32)を解析するにあたり、変数の表記や定義について説明する。

[0047] ・は、算術乗算を表す。

‖は、データの連結(concatenation)を表す。

X <<< nは、データXの左nビットローテイトを表す。

またlsb(X)はデータXの最下位ビット、LSB(X)はデータXの最下位バイトとし、

lsb(X) = X mod 2

LSB(X) = X mod 2^{8}

となる。

時刻tにおける変数i, j, kをit, jt, ktと表記する。

また時刻tにおけるx番目のS-boxエントリをSt[x]と表記する。

時刻tで出力される鍵ストリームをOtとし、最初の鍵ストリームを出力する時刻をt=1とする。

[0048] ここで、PRGAの初期値をi0 = 0, j0 = 0と定義し、k0は未知となる。

[0049] また解析を行う上で、攻撃者は鍵ストリームを自由に入手可能であるものとする。

[0050] ＜1番目の出力ワードと2番目の出力ワード間の偏り＞

まずGGHN(8,32)の1番目の出力ワードと2番目の出力ワード間に偏りが生じることを説明するため、以下の＜ケース1＞の条件が成り立つ場合について考察する。

[0051] ＜ケース1＞

1. LSB(S1[i1] + S1[j1]) = LSB(S1[i1])、ただし、LSB(S1[i1]) ≠ 1

2. LSB(S2[i2] + S2[j2]) = i2

[0052] ケース1の時、t=1、2における配列Sの最下位バイトの状態遷移について図5及び図6に示す。図5には、配列Sにおいて、アドレス1の値S[1]がAであり、アドレスAの値S[A]がk0+Aであるべきところが0となり、矛盾することが示されている。図6には、配列Sにおいて、アドレス1の値S[1]がA、アドレスAの値S[A]がk0+A、アドレスA＋Cの値S[A+C]が2-C、アドレス2の値S[2]がk0+2であるべきところが、Cとなり、矛盾することが示されている。

[0053] 図2より、t=1のとき、i1=1であり、LSB(S1[i1]) = Aとおくと、j1 = Aとなる。

[0054] ここで、ケース1の条件1を満たす時、

LSB(S1[1] + S1[A]) = LSB(S1[1])

LSB(S1[A]) = 0 ・・・ (1)

となるため、

LSB(k1) = LSB(k0 + S1[j1]) = LSB(k0)

となる。

[0055] ただし、A = 1のとき、

LSB(S1[1] + S1[1]) = LSB(S1[1])

LSB(S1[1]) = 0 ≠ 1

となり、式(1)と矛盾するため、

LSB(S1[i1]) ≠ 1の条件が導かれる。

[0056] t=1で出力される鍵ストリームにおいて以下の関係が成り立つ。

LSB(O1) = LSB(k0) ・・・ (2)

[0057] 同様にt=2のとき、i2=2であり、LSB(S2[i2]) = Cとおくと、j2 = A + Cとなる。

[0058] ここで、ケース1の条件2を満たす時、

LSB(S2[2] + S2[A+C]) = 2

LSB(S2[A+C]) = 2 - C

となる。

[0059] t=2で出力される鍵ストリームにおいて、以下の関係が成り立つ。

LSB(O2) = LSB(k0 + 2) ・・・ (3)

[0060] よって、式(2)、(3)より、1番目と2番目の出力ワードO1、O2について以下の関係式が必ず成り立つ。

[0061] lsb(O1) = lsb(O2) ・・・ (4)

[0062] また、ケース2についても同様に考える。

[0063] ＜ケース2＞

1. LSB(S1[i1] + S1[j1]) = LSB(S1[i1])、ただし、LSB(S1[i1]) ≠ 1

2. LSB(S2[i2] + S2[j2]) = j2

[0064] ケース2の場合、t=1、2における配列Sの最下位バイトの状態遷移について、図5及び図7に示す。図7には、アドレス1の値S[1]がA、アドレス2の値S[2]がC、アドレスAの値S[A]がk0+A、アドレスA＋Cの値S[A+C]がk0+A＋Cであるべきところがとなり、矛盾することが示されている。

[0065] t=1における内部変数kはケース1と同じため、配列Sの状態遷移や鍵ストリームにおける関係式(2)も同じとなる。

[0066] t=2のときケース2の条件2を満たす時、

LSB(S2[2] + S2[A+C]) = A + C

LSB(S2[A+C]) = A

となる。

[0067] t=2で出力される鍵ストリームにおいて以下の関係が成り立つ。

[0068] LSB(O2) = LSB(k0 + 2･S1[1]) ・・・ (5)

[0069] よって、ケース2の条件1、2を満たす時も式(2)、(5)より、式(4)が必ず成り立つ。

[0070] このように、ケース1、2ともに、1番目の出力ワードO1と2番目の出力ワードO2間で同じ関係式(4)が成り立つ。

[0071] 次に、この式がdistinguisherとして利用できることを説明する。

[0072] ＜Distinguisherが成り立つ確率と必要なデータ量＞

ここでは、distinguisherとして利用する式(4)の成立確率について説明する。

[0073] もしGGHN(8,32)の出力する系列が真の乱数系列であれば、distinguisherである式(1)が成り立つ確率は2^{-1}となる。

[0074] 式(4)の成立確率は、PRGAの構造に依存したものであり、KSAの構造には依存しな

い。

[0075] よって、以下の検討では、KSA終了後の変数kと配列Sがそれぞれ独立で一様分布に従うものとする。

[0076] まず、ケース1、2の条件1、2が成り立つ確率p1、p2は以下のとおりである。ここで条件2が成り立つ確率p2はケース1、2を考慮した確率である。

[0077] $p1 = 1/256 \cdot 255/256$

$p2 = 1/256 \cdot 1/256 + 255/256 \cdot 2/256$

[0078] ここで条件1、2のいずれかの条件を満たさない時、式(1)が成り立つ確率を、理想的に1/2と仮定すると、GGHN(8,32)の出力系列において、式(4)が成り立つ確率Pdは、以下で与えられる。

[0079] $pd = 1 \cdot p1 \cdot p2 + 1/2 \cdot (1 - p1 \cdot p2)$

$\fallingdotseq 1/2 \cdot (1 + 2^{-15.01})$

[0080] よって、真の乱数系列における確率1/2と比べて大きくなる。

[0081] 次に、式(4)をdistinguisherとしたとき、GGHN(8,32)の出力系列と真の乱数系列を区別するために必要なデータ量を検討する。

[0082] 非特許文献2によると、2つの分布を区別するのに必要なデータ量は、以下のようになることが示されている。

[0083] 確率pで発生する事象分布Xと、確率p(q+1)で発生する事象分布Yにおいてあるイベントeが生じる時、無視できない成功確率でXとYを区別するには、$O(1/pq^{2})$のサンプルが必要である。

[0084] ただし、上記定理は、$p \ll 1$のときに成り立つ。

[0085] 非特許文献3には、p=1/2のとき、2つの分布を区別するのに必要なデータ量は以下のようになることが示されている。

[0086] 確率p = 1/2で発生する事象分布Xと確率1/2(q+1)で発生する事象分布Yにおいてあるイベントeが生じる時、無視できない成功確率でXとYを区別するには$O(1/q^{2})$のサンプルが必要である。

[0087] 本解読におけるイベントeは、式(4)が成り立つ事象であり、乱数におけるイベントeの分布をX、GGHN(8,32)の出力系列におけるイベントeの分布をYとみなすことができる

。

[0088] よって、p = 2^{-1}、q = 2^{-15.01}と考えることができるため、解読に必要なデータ量はO(2^{30.02)となる。

[0089] ここで求まるデータ量は、GGHN(8,32)のKSAが完全にランダムなpermutationという仮定の元での値であり、PRGAの構造的な偏りから求まる理論的なデータ量である。

[0090] よって、GGHN(8,32)は、理論的に約2^{30}の秘密鍵における鍵ストリームの先頭2ワードを利用することで、真の乱数列との区別が可能である。

[0091] ＜1番目の出力ワードと2番目の出力ワード間の偏り＞の説明（段落0050から0070）では、鍵ストリームの先頭2ワードについてdistinguisherの構成方法を説明したが、ケース1については任意の時刻tにおいて連続する鍵ストリーム2ワードについて同様の関係が成り立つ。

[0092] よって、鍵ストリームの先頭数ワードを捨てるという対策は効果が無い。

[0093] 前述した＜発明が対象とするGGHN(n,m)攻撃手法 Distinguishing Attackの説明＞は、図5乃至図9に、まとめることができる。

[0094] 図9は、解読に必要なデータ量（理論値）をあらわしており、初期処理によってS-boxエントリが一様にランダムで出力均等と仮定した場合の、解読に必要なデータ量の理論値を求める流れを説明する図である。

[0095] 図9では、初期処理によってS-boxエントリが一様にランダムで出力均等と仮定している。これは、攻撃する時刻において、Sの値256通りがすべて出現する可能性があるということである。図9における、確率p1(=1/256)、確率p2(=255/256)は、図5における、条件1が発生する確率、と、条件2が発生する確率にそれぞれ対応する。また、図9における、確率p3 (=(1/256)・(1/256)+(255/256)(2/256)=512/256^2)は、図7における条件3、または、図6における条件4が発生する確率に対応する。解読に必要なデータ量（理論値）は、O(q^{-2})=O($2^{30.02}$)となる。

[0096] 図8に示すように、G. Gongら提案した改良アルゴリズムの distinguisher を構成することが出来る。図8は、

lsb(O1) = lsb(O2)

の式を、distinguisherとすることを説明する図である。

[0097] 連続する出力の下位8bitは以下のようになる。

O1=k0

O2=k0＋2A （条件3）、

O2＝k0+2 （条件4）

[0098] よって、式(4)をdistinguisherとすれば、GGHN(8,32)の出力系列と真の乱数系列との区別が可能となる。

[0099] 本発明者等は、このことを、確めるための実験を行ったので、以下に説明する。図10と図11は、実験の結果をまとめたものである。図10には、計算機実験によって得られた確率と解読に必要なデータ量が示されている。図11には、計算機実験によって、与えるデータ量Nを変化させながら、図8のdistinguisherが機能するか確かめた結果が示されている。

[0100] すなわち、図11では、与えるデータ量Nを変化させながら、図8のdistinguisherについて成立回数Xを求めた。100通りの秘密鍵について実験を行い棄却率を求めた。

$X - 2^{N-1} > (1/2)\cdot\sqrt{(2^{N-1} - 2^{N-2})}$

を満たすとき、乱数でないと棄却する(乱数であれば30.5%となる)。以下に、該実験の説明をする。

[0101] ＜実験結果の説明＞

式(4)をdistinguisherとしたとき、図9のように、GGHN(8,32)の出力系列と真の乱数系列が区別可能であるかどうか確認した。実験の手順は以下のとおりである。

[0102] 1. 秘密鍵を2^{w}回ランダムに変更し、GGHN(8,32)の鍵ストリームをそれぞれ2ワード生成する。

[0103] 2. 1.で生成した2^{w}個の鍵ストリームについて式(1)の成立回数をカウントする。

[0104] 3. 2.でカウントされた回数xが以下の関係式を満たした時、出力系列が乱数でないと棄却する。ここでμは平均値、σは標準偏差を表す。

[0105] $\mu - x > \sigma/2$

[0106] よって本実験においては、

$2^{w-1} - x > 1/2\cdot(2^{w-1} - 2^{w-2})^{-1/2}$

の関係式を満たした時、出力系列が乱数でないとして棄却する。

[0107] 4. 1.で与えた2^{w}個の秘密鍵グループを独立に100通り与え、1～3を繰り返し、棄却率を求める。

[0108] 実験結果を表として示した図11によると、2^{28}のデータを与えたとき、棄却率が85%となり、乱数における棄却率と比べて、50%以上のアドバンテージが得られた。

[0109] よって、式(4)を、distinguisherとしたときの攻撃手法によって、GGHN(8,32)の出力系列は、約2^{30}ワードの鍵ストリームを用いることによって、極めて高い確率で真の乱数列との区別が可能である事が実験的にも確認できた。

[0110] このように、従来のGGHN(8,32)の出力系列は、式(4)

{lsb(O1) = lsb(O2)}

を、distinguisherとしたときの攻撃手法によって、鍵ストリームが高い確率で真の乱数列との区別が可能であり、安全性が低いという課題がある。

[0111] 本発明は、本発明者等による上記課題の認識に基づき創案されたものであって、その目的は、データを秘匿するための安全性の高い暗号装置、プログラム、方法を提供することにある。

課題を解決するための手段

[0112] 本願で開示される発明は、前記課題を解決するため、概略以下の構成とされる。

[0113] 本発明は、上記課題となった解析手法に、耐性を持つ対策を提供する。また、対策の実装において、暗号の設計者らの主張する安全性と実装性を損なうことのないよう考慮している。

[0114] 本発明の1つのアスペクト(側面)に係る暗号装置は、秘密鍵に基づき擬似乱数列を生成し、前記擬似乱数列を平文に作用させることにより暗号文を生成する暗号装置であって、前記擬似乱数列の生成に用いる内部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、前記擬似乱数列の生成に用いる一時変数の少なくとも一つが、前記内部状態のうちの一つ、又は、複数個の数値を使用した、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数であり、前記生成される擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数との演算によって生成する。

[0115] 本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新を、並び替え以外の、線形演算、及び、非線形演算を用いて行うようにしてもよい。

[0116] 本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新として、並び替え以外の、線形演算及び非線形演算を用いることにより、前記内部状態の状態数が、単調増加する、ようにしてもよい。あるいは、前記内部状態の状態数が、単調減少する、ようにしてもよい。

[0117] 本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新として、並び替え以外の線形演算及び非線形演算を用いることにより、内部状態の状態数を振動させる、ようにしてもよい。

[0118] 本発明においては、前記内部状態の更新を、前記擬似乱数列の出力ごとに行うようにしてもよい。あるいは、前記擬似乱数列の出力よりも多く行うようにしてもよい。あるいは、前記擬似乱数列の出力よりも少なく行うようにしてもよい。

[0119] 本発明においては、前記ローテイトシフトの方向及び／又は数値(シフト数)を、前記内部状態の数値に依存して動的に変更するようにしてもよい。

[0120] 本発明においては、前記ローテイトシフトの方向及び数値(シフト数)を、あらかじめ定められたテーブルの値に従って変更するようにしてもよい。

[0121] 本発明の別のアスペクトに係る装置は、配列Sの要素の並び替えと算術加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、内部変数kの初期値を配列Sの要素から求める第1の処理部(KSA)と、第1、第2のインデックス変数iとjに関して配列Sの要素S[j]と内部変数kの加算結果をローテイトシフトした値で内部変数kの値を更新し、S[i]＋S[j]による配設Sの参照結果S[(S[i]＋S[j])]と内部変数kとの加算結果に基づき、鍵ストリームを出力し、前記鍵ストリームを生成するために参照した配列Sの要素S[(S[i]＋S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理部(PRGA)と、を備えている。

[0122] 本発明の別のアスペクトに係る装置は、配列Sの要素の並び替えと算術加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、内部変数kの初期値を配列Sの要素から求める第1の処理部(KSA)と、第1のインデックス変数iの

配列要素S[i]を第1のシフト数ローテイトシフトした値と、第2のインデックス変数jとを算術加算した結果に基づき、第2のインデックス変数jの値を更新し、第2のインデックス変数jの配列要素S[j]を第2のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき内部変数kの値を更新し、S[i]＋S[j]による配設要素S[(S[i]＋S[j])]を第3のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき、鍵ストリームを出力し、前記鍵ストリームを生成するために参照した配列要素S[(S[i]＋S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理部(PRGA)と、を備えている。

[0123] 本発明においては、初期設定配列aを用い、配列aの並べ替えを行って攪拌し、内部変数kを対応する配列要素aをローテイトシフトしたものと算術加算して求め、配列Sを、配列aの要素をローテイトシフトしたものと配列Sの要素との所定の演算によって求める第1の処理部(KSA)と、第1、第2のインデックス変数i、jに関して配列Sの参照結果S[j]と内部変数kとを算術加算し、S[i]＋S[j]による配設Sの参照結果と内部変数kに基づき、鍵ストリームを出力し、鍵ストリームを生成するために参照したSのエントリを、鍵ストリーム出力直後に内部変数kを用いて更新する第2の処理部(PRGA)と、を備えた構成としてもよい。上記第1の処理部(KSA)と第2の処理部(PRGA)は、コンピュータ・プログラム(ソフトウェア)として実装してもよい。

[0124] また、本発明に係る方法においては、擬似乱数列の生成に用いる内部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、前記擬似乱数列の生成に用いる一時変数の少なくとも一つを、前記内部状態のうちの一つ、又は、複数個の数値を用いた、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数とし、前記擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数との予め定められた所定の演算によって生成する。本発明によれば、上記第1の処理部(KSA)と第2の処理部(PRGA)の各手順を含む方法が提供される。

発明の効果

[0125] 本発明によれば、GGHN(n,m)におけるdistinguisherの構成を困難とし、GGHN(n,m)

がもつ、速度性能の低下を回避することができる。このため、本発明は、データの通信や蓄積の際にデータを秘匿するための安全性の高い暗号装置を提供することができる。

図面の簡単な説明

[0126] [図1]本発明の一実施形態を示すブロック図である。

[図2]32bit RC4 (CISC 2005)で提案されたRC4の改良アルゴリズムを示す図である。

[図3]PRGAの状態遷移 (1)を示す図である。

[図4]PRGAの状態遷移 (2)を示す図である。

[図5]PRGAの解析 (1)を示す図である。

[図6]PRGAの解析 (3)を示す図である。

[図7]PRGAの解析 (2)を示す図である。

[図8]PRGAの解析 (4)を示す図である。

[図9]解読に必要なデータ量(理論値)を示す図である。

[図10]解読に必要なデータ量(実験値)を示す図である。

[図11]実験結果 (distinguisher)を示す図である。

[図12]G. Gong らによってしめされたストリーム暗号のアルゴリズムを示す図である。

[図13]本発明の変更アルゴリズムを示す図である。

[図14]解決しようとする課題の具体例1を示す図である。

[図15]解決しようとする課題の具体例2を示す図である。

[図16]対策案 (1)を示す図である。

[図17]対策案 (2)を示す図である。

発明を実施するための最良の形態

[0127] 上記した本発明についてさらに詳細に説明する。図13は、本発明の変更アルゴリズムを説明するための図である。図13には、本発明において、元のアルゴリズムの変更すべき箇所が示されている。

[0128] 上記課題となった解析手法は、PRGAの構造的な脆弱性を利用したものである。

[0129] しかも、解析手法では、KSAによって全ての内部記憶が一様分布に従うとみなしていた。

[0130] よって、本発明の1つの側面においては、KSAに変更は施さず、PRGAのみ改良をおこなった。

[0131] 図13に示す対策手段におけるアルゴリズムの変更(図2のPRGAのアルゴリズムからの変更)は、変数kの更新の際に、左ローテイト処理 k=((k+S[j])<<<n) mod M

を行っている点である(ただし、N=2^{n}、M=2^{m})。

[0132] ローテイト数は、GGHN(n,m)としたとき、nビットとする。

[0133] 本発明によれば、擬似乱数列の生成に用いる一時変数の少なくとも一つである内部変数kを、前記内部状態のうちの一つ又は、複数個の数値を使用した、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、nビットローテイトシフト((k+S[j])<<<n)を実行した結果とし、前記生成される擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数との演算によって生成する(out=(S[(S(i)+S[j]) mod N] +k) mod M)。

[0134] ここで、本発明による対策手段の効果を説明するため、ケース1における内部変数k0とS1[A]、S2[A+C]をバイト単位の変数を用いて以下のように表現する。

[0135] ここで、右側が下位ビットであり、LSB(k0) = k00である。

ただし、式(1)より、LSB(S1[A]) = B0 = 0である。

[0136] k0 = k03 || k02 || k01 || k00

S1[A] = B3 || B2 || B1 || B0

S2[A+C] = A3 || A2 || A1 || A0

[0137] このとき、算術加算で生じるバイトを跨ぐ桁あがりの影響を無視すると、図13に示したPRGAは、最下位バイトにおいて、一様分布に従う独立な変数が必ず2個以上挿入されるため、式(1)に、偏りが生じなくなる。

[0138] 同様に、どのバイト間で比較しても、一様な分布に従う独立な変数が必ず2個以上挿入されることから、偏りは生じないと考えられる。

[0139] 実際には、桁上がりがバイトを超えて影響するが、基本的な考え方に影響は無い。

[0140] また、図13の対策手段における実装性を考慮すると、処理の増加は、ローテイト処理1回分である。

[0141] 図13の比較例として、元のアルゴリズムを図12に示す。変数kの更新は、

k=(k+S[j]) mod M

にて行っている。

[0142] また、本発明において、実装ターゲットとして、32ビット／64ビットプロセッサ上でのソフトウェアの実装を想定した場合、nビットの左ローテイト処理は速度の劣化が小さいものと思料される。

[0143] また、本発明によれば、新たな内部メモリを用いることなく、改良が可能であるため、メモリの増加も無い。

[0144] よって、本発明による、図13の対策手段は、

・設計者らの設計方針を崩すことなく、

・実装性の利点を損なうこともない、

暗号モデルを実現できるものと期待される。

[0145] 次に、図16に、本発明による、KSA(K, S)の変更による対策手段の例を示す。a[i]を左8xrビットローテイトしたものと変数kとを算術加算した値で、変数kを更新している。S[i]と、a[i]を左8x(3-r)ビットローテイトしたものとの演算結果(排他的論理和演算)にて、S[i]を更新している。

[0146] 図16の、KSA(K, S)の変更による対策手段でも、S-boxはある程度、ランダムになることが実験により確認できており、かつ、各バイトで見たとき出力均等となる。

[0147] そして、KSAの速度は、約2.5倍高速(RC4のKSAと比べて約8倍低速)となり、初期値テーブルを使用しないので、例えば1Kbyteのメモリを削減することができる。

[0148] なお、図16の対策手段は、ローテイトシフトの方向及び／又はシフト数を、前記内部状態の数値に依存して動的に変更することに対応する。ローテイトシフトの方向及びシフト数を、あらかじめ定められたテーブルの値に従って変更するようにしてもよい。

[0149] 図17に、PRGA(S)の変更による対策手段の別の例を示した。S[i]を右24ビットローテイトシフトしたもの(S[i]>>>24)とjとの算術加算結果でjの値を更新し、S[j]を右16ビットローテイトシフトしたもの(S[j]>>>16)とkとの算術加算結果でkの値を更新し、S[(S[i]+S[j]) mod N]を右8ビットローテイトしたものとkとの算術加算結果をoutとしている。

[0150] この対策手段の場合、鍵ストリーム間の関係(1番目と2番目の出力バイトO1とO2)

が、

O1 = k0 + XO2 = k0 + Y

となり、Xの値に応じて、Yの値が決定する関係となる。

[0151] よって、インデックスの仮定で参照されるS-boxエントリを選択できたとしても。仮定で用いたインデックス情報を、上記の関係式に持ち込めない(X, Yの情報によらない恒等式に持ち込めない)ため、Distinguishing Attackの適用が困難になる。

[0152] なお、本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新を、並び替え以外の、線形演算、及び、非線形演算を用いて行うようにしてもよい。

[0153] 本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新として、列の並び替え以外の、線形演算及び非線形演算を用いることにより、前記内部状態の状態数を、単調増加させるようにしてもよい。あるいは、前記内部状態の更新として、列の並び替え以外の、線形演算及び非線形演算を用いることにより、前記内部状態の状態数を、単調減少させるようにしてもよい。

[0154] 本発明において、前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態であって、前記内部状態の更新として、並び替え以外の線形演算及び非線形演算を用いることにより、内部状態の状態数を振動させる、ようにしてもよい。

[0155] 本発明においては、前記内部状態の更新を、前記擬似乱数列の出力ごとに行うようにしてもよい。あるいは、前記擬似乱数列の出力よりも多く行うようにしてもよい。あるいは、前記擬似乱数列の出力よりも少なく行うようにしてもよい。以下実施例に即して説明する。

実施例

[0156] 図1は、本発明の一実施例を説明するための図である。図14、図15は、比較例として、本発明が解決しようとする課題の具体例1、2を示すブロック図である。以下、本発明の一実施例を説明する。図1に示すように、<<<nの処理では、k(0)に対して、左ローテイト処理(k+S[j])<<<nを行ってk(1)としている。(S[(S[i]+S[j]) mod N]+k(1))mod Mをout(1)としている。

[0157] 図14は、本発明が解決しようとする課題の具体例1として、図5から図7（ケース2の条件でのt=1、2における配列Sの最下位バイトの状態遷移）への処理の過程を示した図である。状態変化は、Sの箱で示されており、図には表れていない。はじめに、図14の比較例を参照して説明する（アルゴリズムは図12に示される）。

[0158] 時刻1において、インデックスiが1となる（j(0)の矢印の先の箱から出力される1参照）。インデックスi(0)と1を入力する十字ボックスは、算術加算を表しており、i＝（i＋1）mod N＝1を出力する。

[0159] アドレスiの値S[1]（下位8ビット）がAであり、インデックスjがAとなる。なお、図1、図14、図15において、Sを四角で囲むボックスはS－boxであり、例えばインデックス（例えば1）を入力し、S[1]がAの場合、ボックスからは、Aが出力される図式とされる。j(0)とAを入力とするボックス（算術加算器）は、j(0)=（j(0)+A） mod N＝Aを出力している。

[0160] アドレスjの値S[j]（下位8ビット）が0であり、

変数kの値（下位8ビット）は

0＋k0＝k0

となる。

[0161] アドレスiの値S[i]（下位8ビット）がAであり、アドレスjの値S[j]（下位8ビット）が0であるため、出力O1を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋0＝A

となる。

[0162] アドレスAの値S[A]（下位8ビット）が0であり、変数kの値（下位8ビット）がk0であるため、出力O1の値（下位8ビット）は、

0＋k0＝k0

となる。

[0163] 出力O1を生成した後、アドレスAの値S[A]（下位8ビット）は、

k0＋A

に更新される。

[0164] 時刻2において、インデックスiは2となる。

[0165] アドレスiの値S[i]（下位8ビット）がCであり、インデックスjは、A＋Cとなる。

[0166] アドレスjの値S[j]（下位8ビット）がAであり、変数kの値（下位8ビット）は、

k0＋A

となる。

[0167] アドレスiの値S[i]（下位8ビット）がCであり、アドレスjの値S[j]（下位8ビット）がAであるため、出力O2を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋C

となる。

[0168] アドレスA＋Cの値S[A+C]（下位8ビット）がAであり、変数kの値（下位8ビット）がk0＋Aであるため、出力O2の値（下位8ビット）は、

A＋k0＋A＝k0＋2A

となる。

[0169] 出力O2を生成した後、アドレスA＋Cの値S[A+C]（下位8ビット）は、

k0＋A＋C

に更新される。

[0170] 図15は、図1の比較例として、本発明が解決しようとする課題の具体例2を示す図であり、図5から図6（ケース1の条件でのt=1、2における配列Sの最下位バイトの状態遷移）への処理の過程を示した図である。状態変化はSの箱で示されており、図には表れていない。

[0171] 図15を参照すると、時刻1において、インデックスiが1となる。

[0172] アドレスiの値S[i]（下位8ビット）がAであり、インデックスjがAとなる。

[0173] アドレスjの値S[j]（下位8ビット）が0であり、変数kの値（下位8ビット）は、 0＋k0＝k0

となる。

[0174] アドレスiの値S[i]（下位8ビット）がAであり、アドレスjの値S[j]（下位8ビット）が0であるため、出力O1を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋0＝Aとなる。

[0175] アドレスAの値S[A]（下位8ビット）が0であり、変数kの値（下位8ビット）がk0であるため、出力O1の値（下位8ビット）は、

0＋k0＝k0

となる。

[0176] 出力O1を生成した後、アドレスAの値S[A]（下位8ビット）は、

k0＋A

に更新される。

[0177] 時刻2において、インデックスiは2となる。

[0178] アドレスiの値S[i]（下位8ビット）がCであり、インデックスjはA＋Cとなる。

[0179] アドレスjの値S[j]（下位8ビット）が2－Cであり、変数kの値（下位8ビット）は、

k0＋2－C

となる。

[0180] アドレスiの値S[i]（下位8ビット）がCであり、アドレスjの値S[j]（下位8ビット）が2－Cであるため、出力O2を生成するために必要なアドレスは、

S[i]＋S[j]＝C＋2－C＝2

となる。

[0181] アドレス2の値S[2]（下位8ビット）がCであり、変数kの値（下位8ビット）がk0＋2－Cであるため、出力O2の値（下位8ビット）は、

C＋k0＋2－C＝k0＋2

となる。

[0182] 出力O2を生成した後、アドレス2の値S[2]（下位8ビット）は、

k0＋2－C＋C＝k0＋2

に更新される。

[0183] 図14、図15の比較例と相違して、本発明の一の実施例では、図1に示すように、処理の過程にローテイトシフト（<<<n）が追加されている。k(0)=ローテイトシフト（<<<n）は、((k+S[j])<<<n) mod M

[0184] 本実施例においても、図14と同様、時刻1において、インデックスiが1となる。アドレスiの値S[i]（下位8ビット）がAであり、インデックスjがAの下位8ビットとなる。

[0185] アドレスjの値S[j]（下位8ビット）がB（下位8ビットが0）であり、変数kの値はRoln（k0＋B）となる。なお、Rolnは、Rotate-Left-Shift by n-bitを表している。すなわち、Rol

n(k0＋B)は、図13のk＝((k＋S[j]<<<n) mod Mに対応し、k0＝((k0＋B)<<<n)m od Mが実行される。

[0186] アドレスiの値S[i](下位8ビット)がAであり、アドレスjの値S[j]がB(下位8ビットが0)であるため、出力O1を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋0＝A

となる。

[0187] アドレスAの値S[A]がB(下位8ビットが0)であり、変数kの値がRoln(k0＋B)であるため、出力O1の値(下位8ビット)は、

0＋Roln(k0＋B)＝Roln(k0＋B)

となる。

[0188] 出力O1を生成した後、アドレスAの値S[A](下位8ビット)は、

Roln(k0＋B)＋A

に更新される。

[0189] 時刻2において、インデックスiは2となる。

[0190] アドレスiの値S[i](下位8ビット)がCであり、インデックスjはA＋Cの下位8ビットとなる。

[0191] アドレスjの値S[j](下位8ビット)がAであり、変数kの値は、

Roln(Roln(k0＋B)＋A)

となる。

[0192] アドレスiの値S[i](下位8ビット)がCであり、アドレスjの値S[j](下位8ビット)がAであるため、出力O2を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋C

となる。

[0193] アドレスA＋Cの値S[A+C](下位8ビット)がAであり、変数kの値がRoln(Roln(k0＋B)＋A)であるため、出力O2の値(下位8ビット)は、

A＋Roln(Roln(k0＋B)＋A)

となる。

[0194] 出力O2を生成した後、アドレスA＋Cの値S[A+C](下位8ビット)は、

Roln(Roln(k0＋B)＋A)＋C

に更新される。

[0195] 次に、図1の本実施例において、図15に基づくと、時刻1において、インデックスiが1となる。

[0196] アドレスiの値S[i](下位8ビット)はAであり、インデックスjはAの下位8ビットとなる。

[0197] アドレスjの値S[j](下位8ビット)はB(下位8ビットが0)であり、変数kの値は、

Roln(k0＋B)

となる。

[0198] アドレスiの値S[i]がAであり、アドレスjの値S[j]がB(下位8ビットが0)であるため、出力O1を生成するために必要なアドレスは、

S[i]＋S[j]＝A＋0＝A

となる。

[0199] アドレスAの値S[A](下位8ビット)がB(下位8ビットが0)であり、変数kの値がRoln(k0＋B)であるため、出力O1の値(下位8ビット)は、

0＋Roln(k0＋B)＝Roln(k0＋B)

となる。

[0200] 出力O1を生成した後、アドレスAの値S[A](下位8ビット)は、

Roln(k0＋B)＋A

に更新される。

[0201] 時刻2において、インデックスiは2となる。

[0202] アドレスiの値S[i](下位8ビット)がCであり、インデックスjは、A＋Cの下位8ビットとなる。

[0203] アドレスjの値S[j](下位8ビット)が2－Cであり、変数kの値は、

Roln(Roln(k0＋B)＋2－C)

となる。

[0204] アドレスiの値S[i]がCであり、アドレスjの値S[j]が2－Cであるため、出力O2を生成するために必要なアドレスは、

S[i]＋S[j]＝C＋2－C＝2

となる。

[0205] アドレス2の値S[2]（下位8ビット）がCであり、変数kの値が、

Roln(Roln(k0＋B)＋2－C)

であるため、出力O2の値（下位8ビット）は、

C＋Roln(Roln(k0＋B)＋2－C)

となる。

[0206] 出力O2を生成した後、アドレス2の値S[2]（下位8ビット）は、

Roln(Roln(k0＋B)＋2－C)＋C

に更新される。

[0207] 本発明によれば、データの通信や蓄積の際にデータを秘匿するための安全性の高い暗号装置を得ることができる。

[0208] なお、図13、図16、図17を参照して説明した本発明に係るプログラムは、鍵ストリームを生成する任意のアプリケーションに適用いることができる。本発明に係る暗号装置は、例えばサーバ装置のCPU、記憶装置、ネットワーク等を含む構成とされる。秘密鍵情報は、サーバ装置の記憶装置に記憶される。図1に示した構成において、ローテイトシフト演算は、CPUのALU（演算装置）で行われる。

[0209] 以上、本発明を上記実施例に即して説明したが、本発明は上記実施例の構成にのみ制限されるものでなく、本発明の範囲内で当業者であればなし得るであろう各種変形、修正を含むことは勿論である。

請求の範囲

[1] 秘密鍵に基づき擬似乱数列を生成し、前記擬似乱数列を平文に作用させることにより暗号文を生成する暗号装置であって、

前記擬似乱数列の生成に用いる内部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、

前記擬似乱数列の生成に用いる一時変数の少なくとも一つを、

前記内部状態のうちの一つ、又は、複数個の数値を用いた、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数とし、

前記擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数との予め定められた所定の演算によって生成する手段を備えている、

ことを特徴とする暗号装置。

[2] 前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新を、並び替え以外の、線形演算、および、非線形演算を用いて行う、ことを特徴とする請求項1に記載の暗号装置。

[3] 前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新を、並び替え以外の、線形演算及び非線形演算を用いて行い、前記内部状態の状態数を、単調増加と単調減少のうちの一方にしたがって変化させる、ことを特徴とする請求項1に記載の暗号装置。

[4] 前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新を、並び替え以外の、線形演算及び非線形演算を用いて行い、前記内部状態の状態数を振動させる、ことを特徴とする請求項1に記載の暗号装置。

[5] 前記内部状態の更新を、

前記擬似乱数列の出力ごとに行う、

前記擬似乱数列の出力よりも多く行う、

前記擬似乱数列の出力よりも少なく行う、

のうち選択されたいずれか一つで行う、ことを特徴とする、請求項1乃至4のいずれか一項に記載の暗号装置。

[6] 前記ローテイトシフトの方向及び／又はシフト数を、前記内部状態の数値に依存して動的に変更する、ことを特徴とする、請求項1乃至5のいずれか一項に記載の暗号装置。

[7] 前記ローテイトシフトの方向及びシフト数を、あらかじめ定められたテーブルの値に従って変更する、ことを特徴とする、請求項1乃至5のいずれか一項に記載の暗号装置。

[8] 前記擬似乱数列の生成に用いる内部状態として、配列Sの要素の並び替えと算術加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、前記一時変数をなす内部変数kの初期値を配列Sの要素から求める第1の処理部と、

前記擬似乱数（以下、「鍵ストリーム」と称する）を生成するにあたり、第1、第2のインデックス変数iとjに関して配列Sの要素S[j]と内部変数kの加算結果をローテイトシフトした値で内部変数kの値を更新し、

S[i]＋S[j]による配設Sの参照結果S[(S[i]＋S[j])]と内部変数kとの加算結果に基づき、鍵ストリームを出力し、

前記鍵ストリームを生成するために参照した配列Sの要素S[(S[i]＋S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理部と、

を備えている、ことを特徴とする請求項1記載の暗号装置。

[9] 配列Sの要素数Nとローテイトシフトのシフト数nとが$N=2^n$の関係とされる、ことを特徴とする請求項8記載の暗号装置。

[10] 前記擬似乱数列の生成に用いる内部状態として、配列Sの要素の並び替えと算術加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、前記一時変数をなす内部変数kの初期値を配列Sの要素から求める第1の処理部と、

前記擬似乱数（以下、「鍵ストリーム」と称する）を生成するにあたり、

第1のインデックス変数iの配列要素S[i]を第1のシフト数ローテイトシフトした値と、第2のインデックス変数jとを算術加算した結果に基づき、第2のインデックス変数jの

値を更新し、

第2のインデックス変数jの配列要素S[j]を第2のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき内部変数kの値を更新し、

S[i]＋S[j]による配設要素S[(S[i]＋S[j])]を第3のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき、鍵ストリームを出力し、

前記鍵ストリームを生成するために参照した配列要素S[(S[i]＋S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理部と、

を備えている、ことを特徴とする請求項1記載の暗号装置。

[11] 初期設定配列aを用い、配列aの並べ替えを行って攪拌し、前記一時変数をなす内部変数kを、対応する配列要素aをローテイトシフトしたものと算術加算して求め、前記擬似乱数列の生成に用いる内部状態として配列Sを、配列aの要素をローテイトシフトしたものと配列Sの要素との所定の演算によって求める第1の処理部と、

前記擬似乱数(以下、「鍵ストリーム」と称する)を生成するにあたり、第1、第2のインデックス変数i、jに関して配列Sの参照結果S[j]と内部変数kとを算術加算し、S[i]＋S[j]による配設Sの参照結果と内部変数kに基づき、鍵ストリームを出力し、

鍵ストリームを生成するために参照したSのエントリを、鍵ストリーム出力直後に内部変数kを用いて更新する第2の処理部と、

を備えている、ことを特徴とする請求項1記載の暗号装置。

[12] 秘密鍵に基づき擬似乱数列を生成し、前記擬似乱数列を平文に作用させることにより暗号文を生成する暗号装置を構成するコンピュータに、

前記擬似乱数列の生成に用いる内部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、

前記擬似乱数列の生成に用いる一時変数の少なくとも一つを、

前記内部状態のうちの一つ、又は、複数個の数値を使用した、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数とし、

前記擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数

との予め定められた所定の演算によって生成する、処理

を実行させるプログラム。

[13] 請求項12記載のプログラムにおいて、

前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新を、並び替え以外の、線形演算、および、非線形演算を用いて行う、ことを特徴とするプログラム。

[14] 請求項12記載のプログラムにおいて、

前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新として、並び替え以外の、線形演算及び非線形演算を用いて行い、前記内部状態の状態数を、単調増加と単調減少のうちの一方にしたがって変化させる、ことを特徴とするプログラム。

[15] 請求項12記載のプログラムにおいて、

前記有限個の数値の列の並び替えを基礎とする状態に基づく内部状態に関して、

前記内部状態の更新として、並び替え以外の線形演算及び非線形演算を用いて行い、内部状態の状態数を振動させる、ことを特徴とするプログラム。

[16] 請求項12記載のプログラムにおいて、

前記内部状態の更新を、

前記擬似乱数列の出力ごとに行う、

前記擬似乱数列の出力よりも多く行う、

前記擬似乱数列の出力よりも少なく行う、

のうちの選択されたいずれか一つで行う、ことを特徴とする、プログラム。

[17] 請求項12記載のプログラムにおいて、

前記ローテイトシフトの方向及び／又はシフト数を、前記内部状態の数値に依存して動的に変更する、ことを特徴とする、プログラム。

[18] 請求項12記載のプログラムにおいて、

前記ローテイトシフトの方向及びシフト数を、あらかじめ定められたテーブルの値に従って変更する、ことを特徴とする、プログラム。

[19] 前記擬似乱数列の生成に用いる内部状態として、配列Sの要素の並び替えと算術

加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、前記一時変数をなす内部変数kの初期値を配列Sの要素から求める第1の処理と、

前記擬似乱数(以下、「鍵ストリーム」と称する)を生成する処理として、

第1、第2のインデックス変数iとjに関して配列Sの要素S[j]と内部変数kの算術加算結果をローテイトシフトした値で内部変数kの値を更新し、

S[i]+S[j]による配設Sの参照結果S[(S[i]+S[j])]と内部変数kとの算術加算結果に基づき、鍵ストリームを出力し、

前記鍵ストリームを生成するために参照した配列Sの要素S[(S[i]+S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理と、

を前記コンピュータに実行させる請求項12記載のプログラム。

[20] 前記擬似乱数列の生成に用いる内部状態として、配列Sの要素の並び替えと算術加算を繰り返すことで配列Sの要素を攪拌し配列Sの初期状態を作成し、その際、前記一時変数をなす内部変数kの初期値を配列Sの要素から求める第1の処理と、

前記擬似乱数(以下、「鍵ストリーム」と称する)を生成する処理として、

第1のインデックス変数iの配列要素S[i]を第1のシフト数ローテイトシフトした値と、第2のインデックス変数jとを算術加算した結果に基づき、第2のインデックス変数jの値を更新し、

第2のインデックス変数jの配列要素S[j]を第2のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき内部変数kの値を更新し、

S[i]+S[j]による配設要素S[(S[i]+S[j])]を第3のシフト数ローテイトシフトした値と内部変数kとの算術加算結果に基づき、鍵ストリームを出力し、

前記鍵ストリームを生成するために参照した配列要素S[(S[i]+S[j])]を、鍵ストリーム出力の直後に内部変数kと配列要素S[i]を用いて更新する第2の処理と、

を前記コンピュータに実行させる請求項12記載のプログラム。

[21] 初期設定配列aを用い、配列aの並べ替えを行って攪拌し、前記一時変数をなす内部変数kを、対応する配列要素aをローテイトシフトしたものと算術加算して求め、前記擬似乱数列の生成に用いる内部状態として配列Sを、配列aの要素をローテイトシフトしたものと配列Sの要素との所定の演算によって求める第1の処理と、

前記擬似乱数(以下、「鍵ストリーム」と称する)を生成する処理として、

第1、第2のインデックス変数i、jに関して配列Sの参照結果S[j]と内部変数kとを算術加算し、

S[i]+S[j]による配設Sの参照結果と内部変数kに基づき、鍵ストリームを出力し、

鍵ストリームを生成するために参照したSのエントリを、鍵ストリーム出力直後に内部変数kを用いて更新する第2の処理と、

を前記コンピュータに実行させる請求項12記載のプログラム。

[22] コンピュータを用いて擬似乱数列を生成する方法であって、

擬似乱数列の生成に用いる内部状態として、有限個の数値の列の並び替えを基礎とする状態に基づく内部状態を用い、

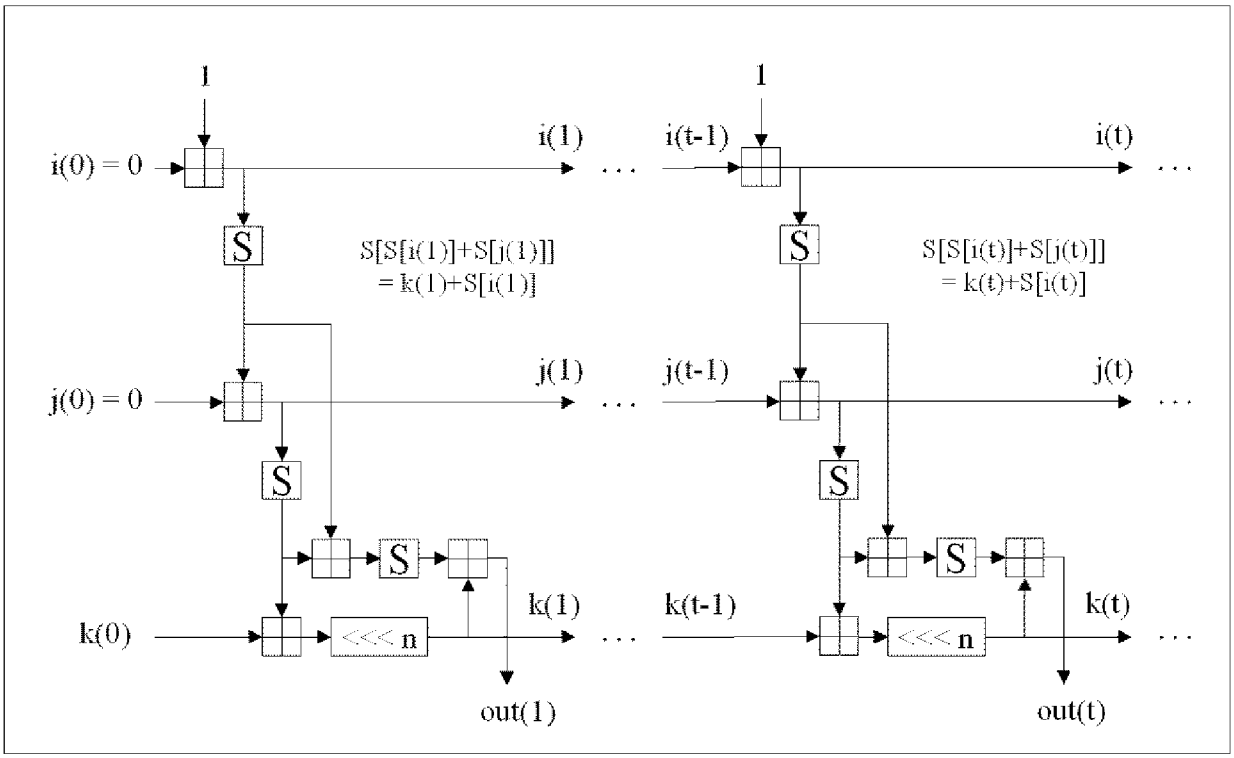
前記擬似乱数列の生成に用いる一時変数の少なくとも一つを、

前記内部状態のうちの一つ、又は、複数個の数値を用いた、線形、又は、非線形、線形及び非線形の組み合わせの結果に基づき、内部状態の数より小さい数に依存し、あらかじめ定められた、左又は右ローテイトシフトを実行した結果を値とする一時変数とし、

前記擬似乱数を、前記内部状態のうちの1つ又は複数個の数値と、前記一時変数との予め定められた所定の演算によって生成する、

ことを特徴とする擬似乱数列の生成方法。

[図1]



[図2]

32bit RC4 (CISC 2005)

KSA(K, S)

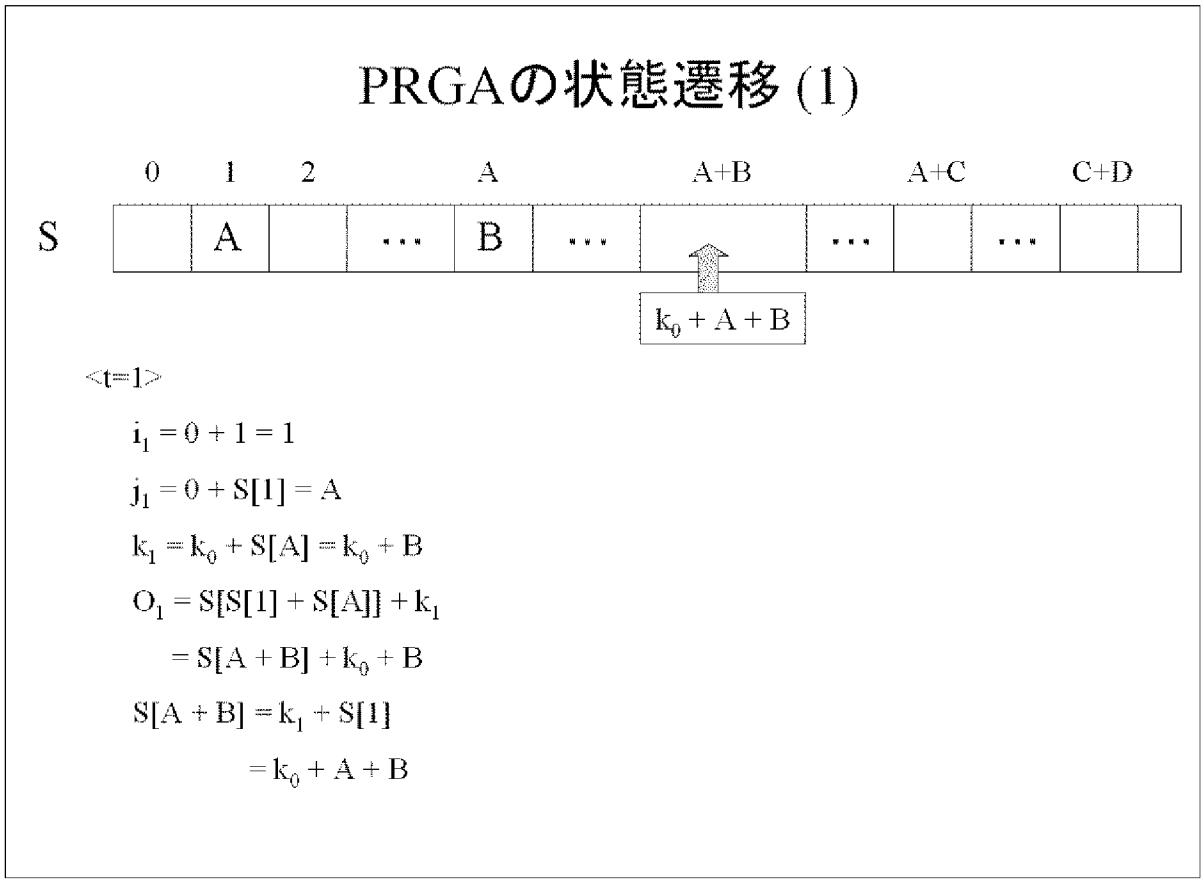
```
for i=0 to N-1
    S[i] = a_i
j = k = 0
Repeat r times
    for i = 0 to N-1
        j = (j + S[i] + K[i mod l]) mod N
        Swap(S[i], S[j])
        S[i] = S[i] + S[j] mod M
        k = k + S[i] mod M
```

t=0

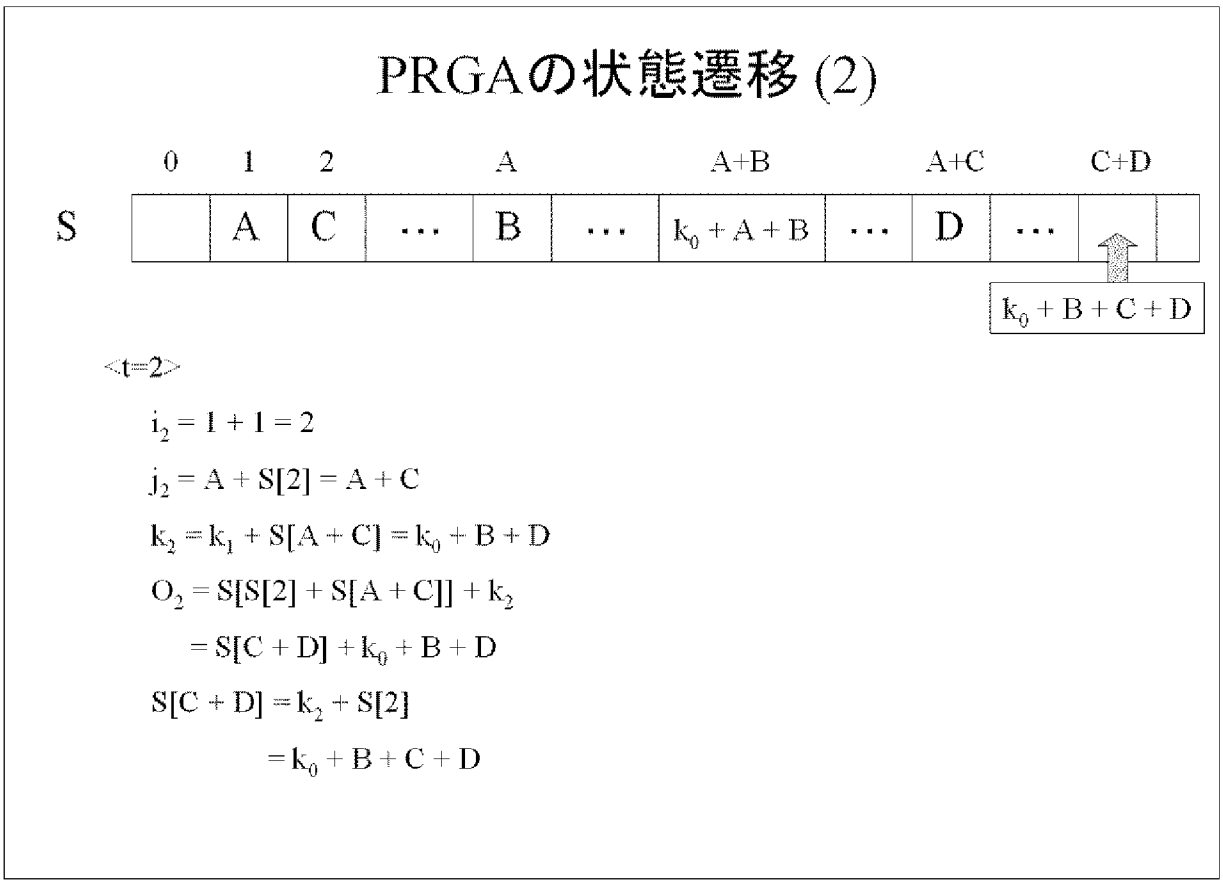
PRGA(S)

```
i = 0
j = 0
while (1)
    i = (i + 1) mod N
    j = (j + S[i]) mod N
    k = (k + S[j]) mod M
    out = (S[(S[i] + S[j]) mod N] + k) mod M
    S[(S[i] + S[j]) mod N] = k + S[i] mod M
```

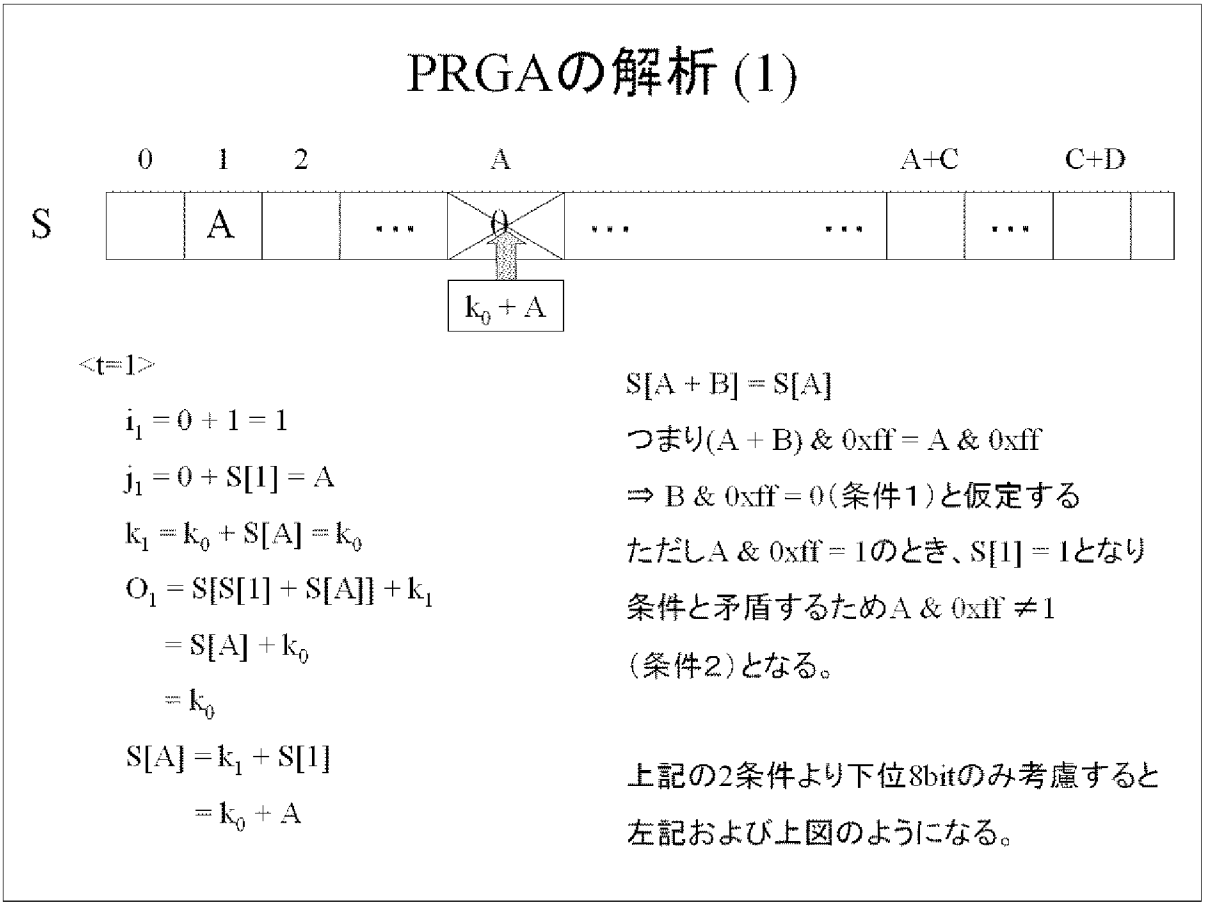
[図3]


PRGAの状態遷移 (1)
0
1
2
A
A+B
A+C
C+D
S
A
B
$k_0 + A + B$
<t=1>
$i_1 = 0 + 1 = 1$
$j_1 = 0 + S[1] = A$
$k_1 = k_0 + S[A] = k_0 + B$
$O_1 = S[S[1] + S[A]] + k_1$
$= S[A + B] + k_0 + B$
$S[A + B] = k_1 + S[1]$
$= k_0 + A + B$

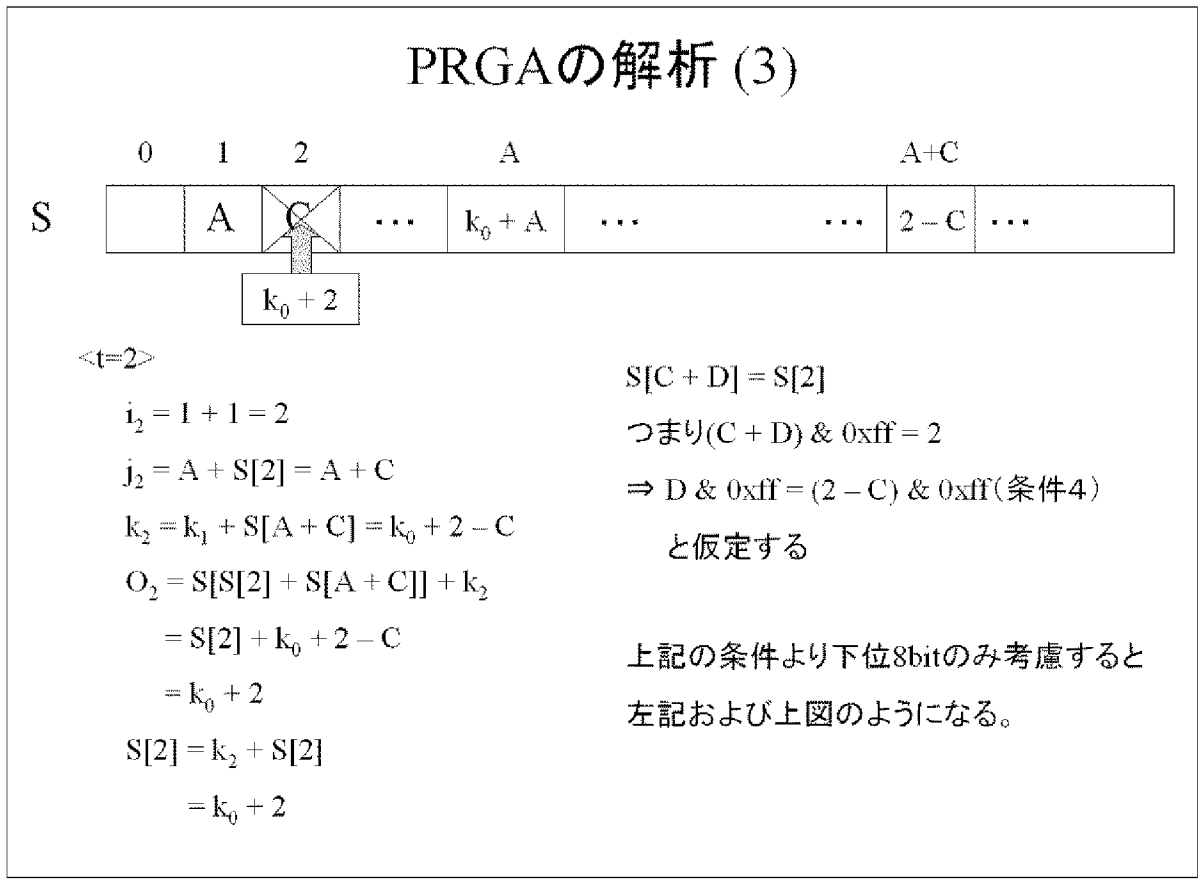
[図4]


PRGAの状態遷移 (2)
0
1
2
A
A+B
A+C
C+D
S
A
C
B
$k_0 + A + B$
D
$k_0 + B + C + D$
<t=2>
$i_2 = 1 + 1 = 2$
$j_2 = A + S[2] = A + C$
$k_2 = k_1 + S[A + C] = k_0 + B + D$
$O_2 = S[S[2] + S[A + C]] + k_2$
$= S[C + D] + k_0 + B + D$
$S[C + D] = k_2 + S[2]$
$= k_0 + B + C + D$

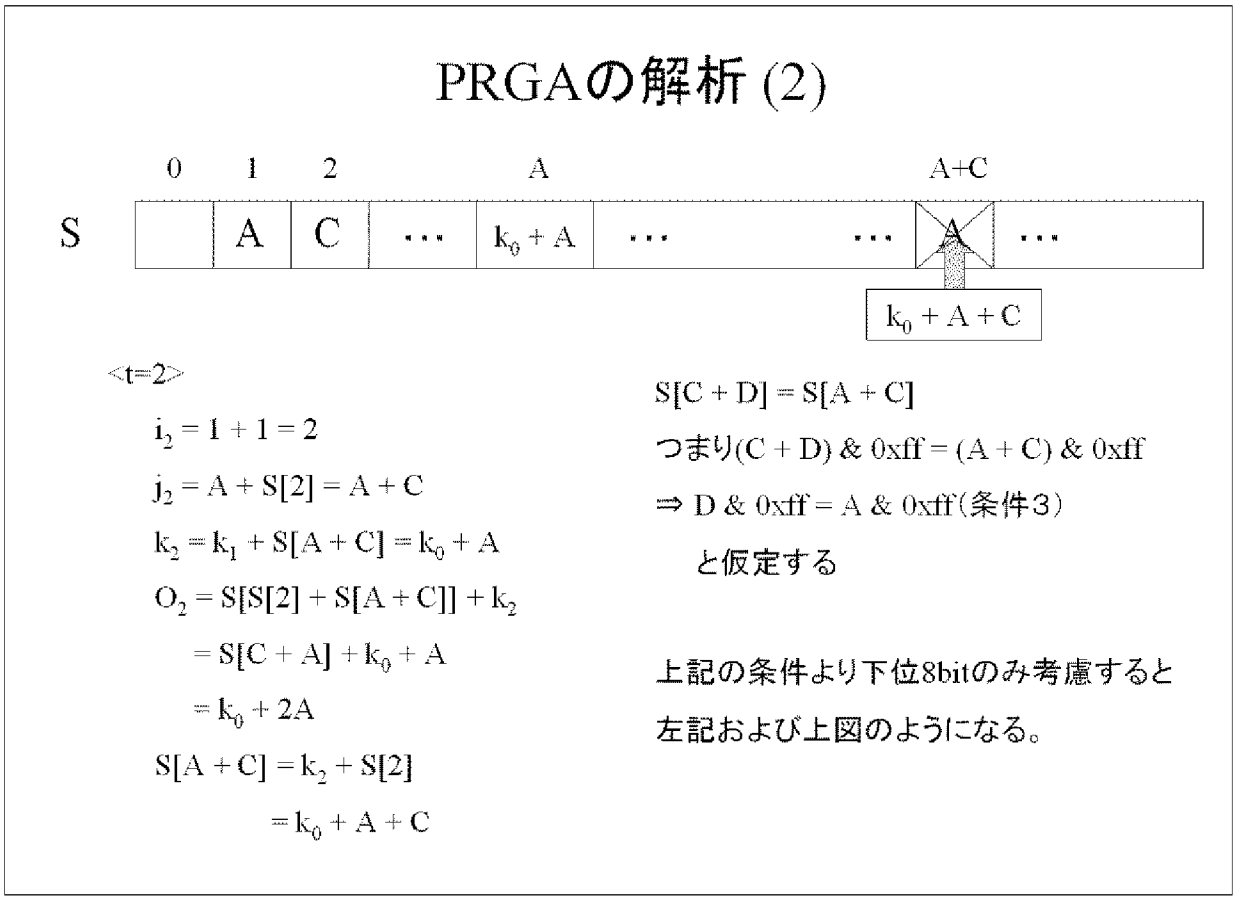
[図5]


PRGAの解析 (1)
0
1
2
A
A+C
C+D
S
A
0
$k_0 + A$
<t=1>
$i_1 = 0 + 1 = 1$
$j_1 = 0 + S[1] = A$
$k_1 = k_0 + S[A] = k_0$
$O_1 = S[S[1] + S[A]] + k_1$
$= S[A] + k_0$
$= k_0$
$S[A] = k_1 + S[1]$
$= k_0 + A$
S[A + B] = S[A]
つまり(A + B) & 0xff = A & 0xff
⇒ B & 0xff = 0（条件1）と仮定する
ただしA & 0xff = 1のとき、S[1] = 1となり
条件と矛盾するためA & 0xff ≠1
（条件2）となる。
上記の2条件より下位8bitのみ考慮すると
左記および上図のようになる。

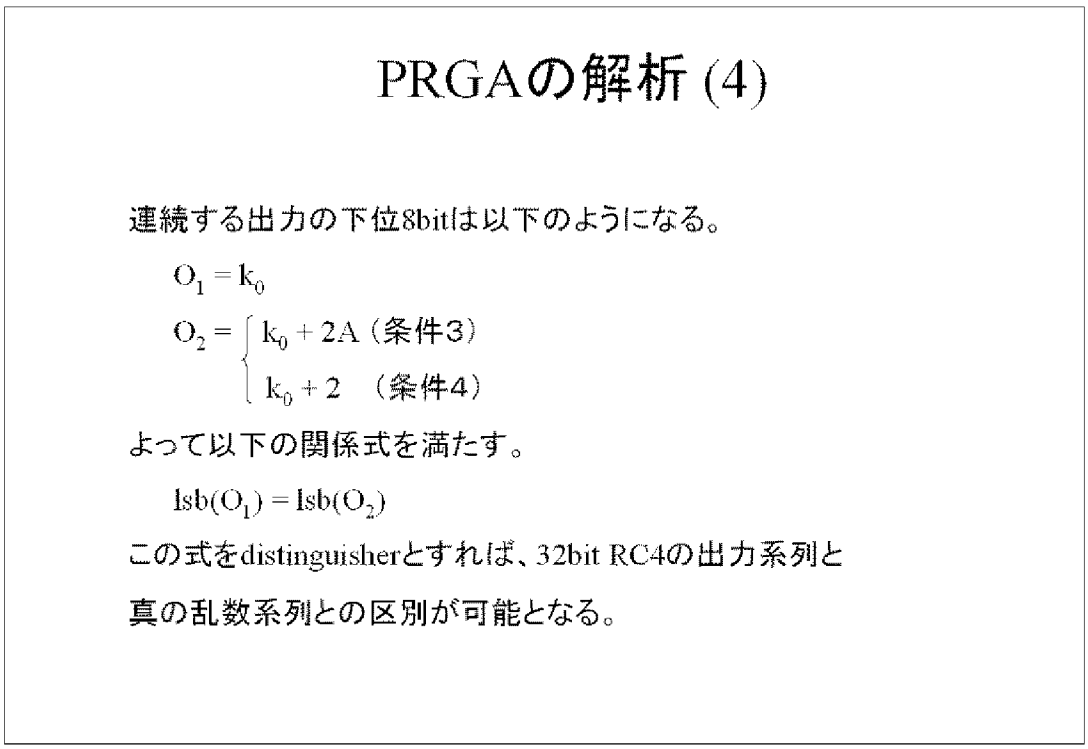
[図6]


PRGAの解析 (3)
0
1
2
A
A+C
S
A
C
$k_0 + A$
2 – C
$k_0 + 2$
<t=2>
$i_2 = 1 + 1 = 2$
$j_2 = A + S[2] = A + C$
$k_2 = k_1 + S[A + C] = k_0 + 2 – C$
$O_2 = S[S[2] + S[A + C]] + k_2$
$= S[2] + k_0 + 2 – C$
$= k_0 + 2$
$S[2] = k_2 + S[2]$
$= k_0 + 2$
S[C + D] = S[2]
つまり(C + D) & 0xff = 2
⇒ D & 0xff = (2 – C) & 0xff（条件4）
と仮定する
上記の条件より下位8bitのみ考慮すると
左記および上図のようになる。

[図7]


PRGAの解析 (2)
0
1
2
A
A+C
S
A
C
・・・
$k_0 + A$
・・・
・・・
A
・・・
$k_0 + A + C$
<t=2>
$i_2 = 1 + 1 = 2$
$j_2 = A + S[2] = A + C$
$k_2 = k_1 + S[A + C] = k_0 + A$
$O_2 = S[S[2] + S[A + C]] + k_2$
$= S[C + A] + k_0 + A$
$= k_0 + 2A$
$S[A + C] = k_2 + S[2]$
$= k_0 + A + C$
S[C + D] = S[A + C]
つまり(C + D) & 0xff = (A + C) & 0xff
⇒ D & 0xff = A & 0xff（条件3）
と仮定する
上記の条件より下位8bitのみ考慮すると
左記および上図のようになる。

[図8]


PRGAの解析 (4)
連続する出力の下位8bitは以下のようになる。
$O_1 = k_0$
$O_2 = \begin{cases} k_0 + 2A \text{（条件3）} \\ k_0 + 2 \quad \text{（条件4）} \end{cases}$
よって以下の関係式を満たす。
$lsb(O_1) = lsb(O_2)$
この式をdistinguisherとすれば、32bit RC4の出力系列と
真の乱数系列との区別が可能となる。

[図9]

解読に必要なデータ量（理論値）

初期処理によってS-boxエントリが一様にランダムで出力均等と仮定すると、

$p1 = 1/256$

$p2 = 255/256$

$p3 = 1/256 \cdot 1/256 + 255/256 \cdot 2/256 = 511/256^2$

となる。条件を満たさない場合にdistinguisherが成り立つ確率は1/2とすると

任意にdistinguisherが成り立つ確率は、

$$\Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2)\}$$
$$= \Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2) \mid \text{条件OK}\} + \Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2) \mid \text{条件NG}\}$$
$$= p1 \cdot p2 \cdot p3 \cdot 1 + \{1 - p1 \cdot p2 \cdot p3\} \cdot 2^{-1}$$
$$= 2^{-1} \cdot (1 + 1/256 \times 255/256 \times 511/256^2)$$
$$\fallingdotseq 2^{-1} \cdot (1 + 2^{-15.008})$$

解読に必要なデータ量は理論的に以下のようになる。

$O(q^{-2}) = O(2^{30.02})$

[図10]

解読に必要なデータ量（実験値）

実験で得られた確率は以下のとおり。

$p1 \cdot p2 \cdot p3 = 2^{-14.404}$

となる。条件を満たさない場合にdistinguisherが成り立つ確率は1/2とすると

任意にdistinguisherが成り立つ確率は、

$$\Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2)\}$$
$$= \Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2) \mid \text{条件OK}\} + \Pr\{\mathrm{lsb}(O_1) = \mathrm{lsb}(O_2) \mid \text{条件NG}\}$$
$$= p1 \cdot p2 \cdot p3 \cdot 1 + \{1 - p1 \cdot p2 \cdot p3\} \cdot 2^{-1}$$
$$\fallingdotseq 2^{-1} \cdot (1 + 2^{-14.404})$$

解読に必要なデータ量は実験的に以下のようになる。

$O(q^{-2}) = O(2^{28.81})$

[図11]

実験結果 (distinguisher)

データ量N	棄却率
2^{24}	42 %
2^{25}	53 %
2^{26}	45 %
2^{27}	67 %
2^{28}	85 %
2^{29}	96 %

[図12]

元アルゴリズム

PRGA(S)
i = 0
j = 0
while (1)
    i = (i + 1) mod N
    j = (j + S[i]) mod N
    k = (k + S[j]) mod M
    out = (S[(S[i] + S[j]) mod N] + k) mod M
    S[(S[i] + S[j]) mod N] = k + S[i] mod M

($N = 2^n$, $M = 2^m$)

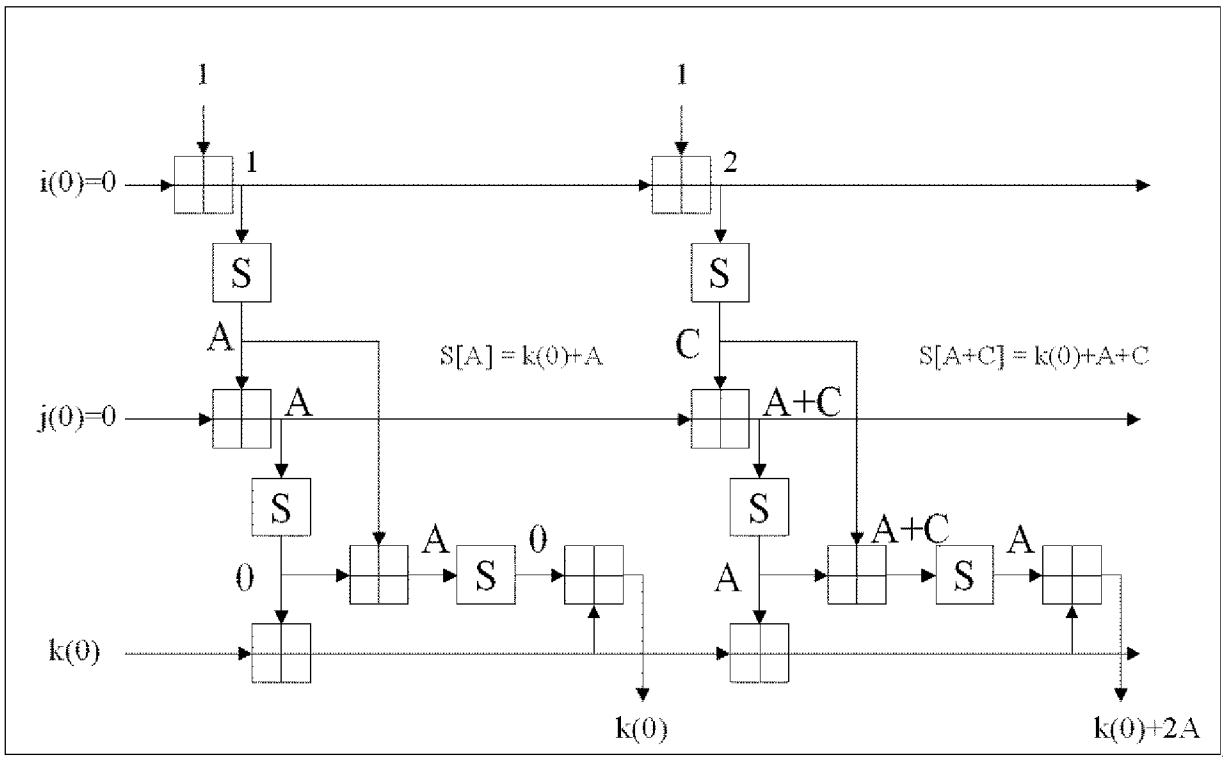
[図13]

変更アルゴリズム

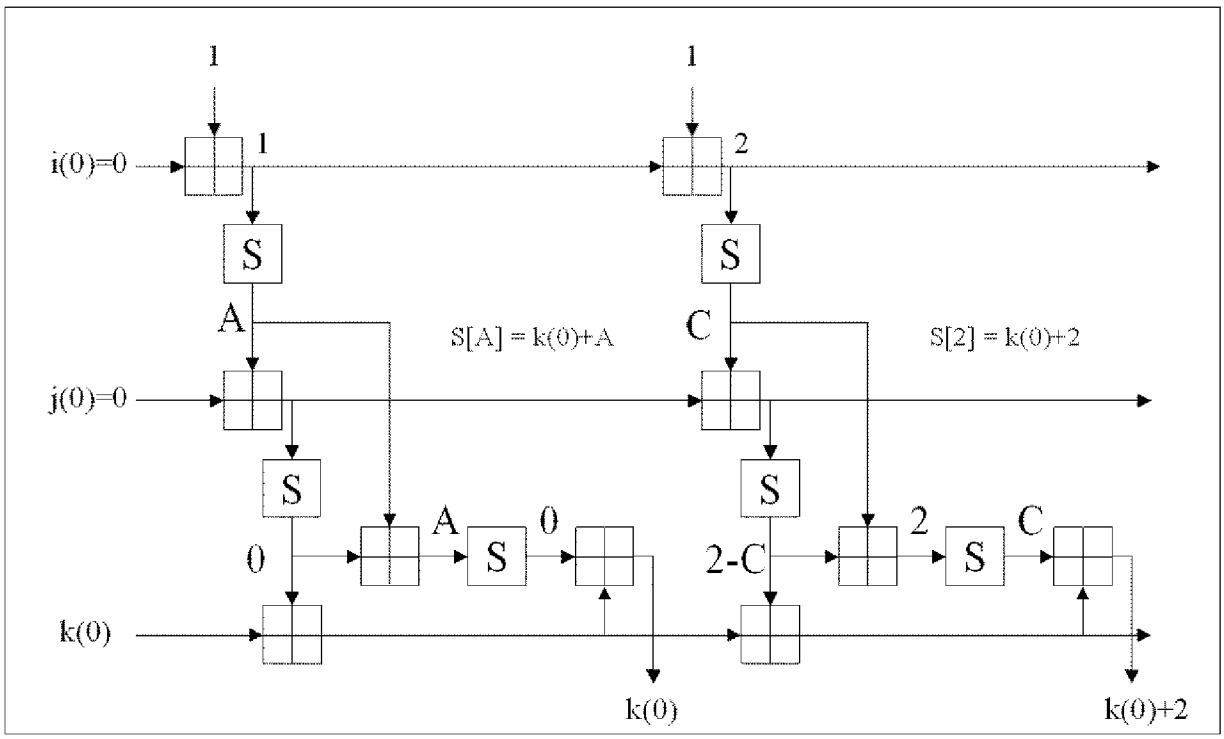
PRGA(S)
i = 0
j = 0
while (1)
    i = (i + 1) mod N
    j = (j + S[i]) mod N
    k = ((k + S[j]) <<< n) mod M
    out = (S[(S[i] + S[j]) mod N] + k) mod M
    S[(S[i] + S[j]) mod N] = k + S[i] mod M

(N = 2^n, M = 2^m)

[図14]



[図15]



[図16]

KSA(K, S)
for i=0 to N-1
    a[i] = i
j = k = 0
for r = 0 to 3
    for i = 0 to N-1
        j = (j + a[i] + K[i mod l]) mod N
        Swap(a[i], a[j])
        k = k + (a[i] <<< 8×r)
    for i = 0 to N-1
        S[i] = S[i] ⊕ (a[i] <<< 8×(3-r))

[図17]

```
PRGA(S)
i = 0
j = 0
while (1)
    i = (i + 1) mod N
    j = (j + (S[i] >>> 24)) mod N
    k = (k + (S[j] >>> 16)) mod M
    out = ((S[(S[i] + S[j]) mod N] >>> 8) + k) mod M
    S[(S[i] + S[j]) mod N] = k + S[i] mod M
```

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2007/063797

A. CLASSIFICATION OF SUBJECT MATTER
H04L9/22(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L9/22

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2007
Kokai Jitsuyo Shinan Koho 1971-2007 Toroku Jitsuyo Shinan Koho 1994-2007

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Gong et al."Towards a General RC4-like Keystream Generator", [online], Guang Gong's Home Page, [retrieved on 2007-08-28], Retrieved from the Internet: <URL:http://calliope.uwaterloo.ca/~ggong/publication/CISC141.pdf>	1-22
A	Wu,"Cryptanalysis of a 32-bit RC4-like Stream Cipher", [online], Cryptology ePrint Archive, 2005/219, [retrieved on 2007-08-28], Retrieved from the Internet: <URL:http://eprint.iacr.org/2005/219.pdf >	1-22

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
28 August, 2007 (28.08.07)	04 September, 2007 (04.09.07)
Name and mailing address of the ISA/ Japanese Patent Office	Authorized officer
Facsimile No.	Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2007/063797

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Tsunoo et al."The Most Efficient Distinguishing Attack on VMPC and RC4", [online], eSTREAM, the ECRYPT Stream Cipher Project, [retrieved on 2007-08-28], Retrieved from the Internet: <URL: http://www.ecrypt.eu.org/stream/papersdir/037.pdf >	1-22

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００７／０６３７９７

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　H04L9/22(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　H04L9/22

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２００７年
日本国実用新案登録公報	１９９６－２００７年
日本国登録実用新案公報	１９９４－２００７年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	Gong et al.” Towards a General RC4-like Keystream Generator”, [online], Guang Gong's Home Page, [retrieved on 2007-08-28], Retrieved from the Internet : <URL:http://calliope.uwaterloo.ca/~ggong/publication/CISC141.pdf>	１－２２

☑　Ｃ欄の続きにも文献が列挙されている。　　☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
２８．０８．２００７	０４．０９．２００７

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｓ　３８５７
石田　信行
電話番号　０３－３５８１－１１０１　内線　３５４６

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	Wu,” Cryptanalysis of a 32-bit RC4-like Stream Cipher”, [online],Cryptology ePrint Archive, 2005/219, [retrieved on 2007-08-28], Retrieved from the Internet : <URL : http://eprint.iacr.org/2005/219.pdf >	１－２２
A	Tsunoo et al.” The Most Efficient Distinguishing Attack on VMPC and RC4”, [online], eSTREAM, the ECRYPT Stream Cipher Project, [retrieved on 2007-08-28], Retrieved from the Internet : <URL: http://www.ecrypt.eu.org/stream/papersdir/037.pdf >	１－２２