(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2017-187724
(P2017-187724A)
(43)公開日　平成29年10月12日(2017.10.12)

(51)Int.Cl.		F I			テーマコード（参考）
G09C 1/00 (2006.01)		G09C	1/00	610A	5J104

審査請求 未請求　請求項の数 15　ＯＬ　（全 25 頁）

(21)出願番号　特願2016-78386(P2016-78386)
(22)出願日　平成28年4月8日(2016.4.8)

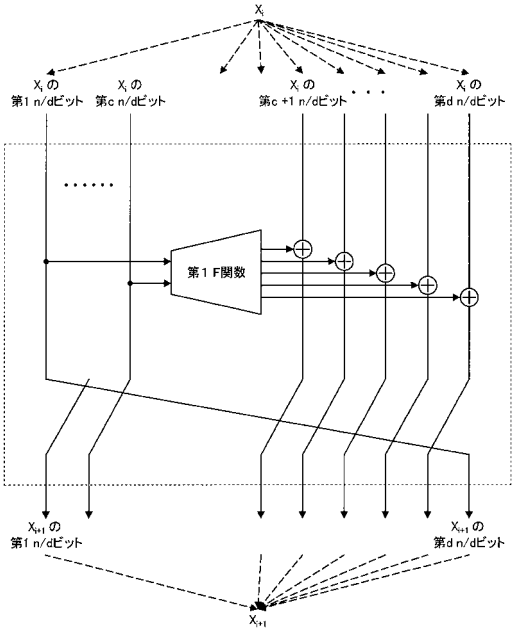
申請有り

(71)出願人　000002185
ソニー株式会社
東京都港区港南１丁目７番１号
(71)出願人　504213249
テクニカル　ユニバーシティ　オブ　デンマーク
デンマーク王国　２８００　コーゲーエス　リュンビュー　アンケル　エンエルンズバイ　１　ビルディング　１０１エー
(74)代理人　100095957
弁理士　亀谷　美明
(74)代理人　100096389
弁理士　金本　哲男
(74)代理人　100101557
弁理士　萩原　康司

最終頁に続く

(54)【発明の名称】暗号化装置、暗号化方法、復号化装置、及び復号化方法

(57)【要約】
【課題】ホワイトボックスモデルの暗号化において、データの秘匿性を高める。
【解決手段】本開示に係る暗号化装置は、入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化するデータ暗号化部を備え、複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいて入力値を暗号化するテーブル化された暗号化関数を有する。
【選択図】図１０



【特許請求の範囲】
【請求項１】
入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化するデータ暗号化部を備え、
複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいて入力値を暗号化するテーブル化された暗号化関数を有する、暗号化装置。
【請求項２】
前記暗号化関数には、前記ラウンド関数へ入力されるビットのうちの一部が入力され、
前記暗号化関数は、前記暗号化関数へ入力可能なビットの一部を固定値とし、前記暗号化関数の出力値の一部を破棄することで、前記暗号化関数へ入力可能なビット数から前記暗号化関数へ入力されたビット数の差分に相当するビット数の出力値を出力する、請求項１に記載の暗号化装置。
【請求項３】
前記ラウンド関数は、前記ラウンド関数へ入力されるビットのうち前記暗号化関数に入力されなかったビットと、前記暗号化関数からの前記出力値のビットの排他論理和を演算する、請求項２に記載の暗号化装置。
【請求項４】
前記ラウンド関数は、前記暗号化関数へ入力されたビットの値と前記排他論理和により得られたビットの値を出力する、請求項３に記載の暗号化装置。
【請求項５】
前記ラウンド関数は、前記暗号化関数へ入力されたビットの値を前記排他論理和により得られたビットの値よりも下位ビットとして出力する、請求項４に記載の暗号化装置。
【請求項６】
前記ラウンド関数の出力と予め定められた所定値との排他論理和を演算し、得られた値を次のラウンド関数への入力又は前記データ暗号化部の出力とする、請求項１に記載の暗号化装置。
【請求項７】
１の前記ラウンド関数が複数の前記暗号化関数を有する、請求項１に記載の暗号化装置。
【請求項８】
複数の前記ラウンド関数において、後段のラウンド関数ほど前記暗号化関数に大きなビットの入力値が入力される、請求項２に記載の暗号化装置。
【請求項９】
１の前記ラウンド関数が複数の前記暗号化関数を有し、
前記ラウンド関数へ入力されるビットが分割されて複数の前記暗号化関数へ入力され、
複数の前記暗号化関数は非線形演算を行い、
前記ラウンド関数は、複数の前記暗号化関数による前記非線形演算の結果を線形変換演算して出力する、請求項１に記載の暗号化装置。
【請求項１０】
複数の前記暗号化関数のそれぞれにおいて、入力されるビット数と出力されるビット数が同一である、請求項９に記載の暗号化装置。
【請求項１１】
複数の前記暗号化関数のそれぞれに入力されるビット数が異なる、請求項９に記載の暗号化装置。
【請求項１２】
前記暗号化関数は、前記データ暗号化部に対応する秘密鍵から生成される拡張鍵によって暗号化を行う、請求項１に記載の暗号化装置。
【請求項１３】

入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化することを備え、
複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、暗号化方法。
【請求項１４】
入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化する暗号化処理の逆演算により復号を行うデータ復号化部を備え、
複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、復号化装置。
【請求項１５】
入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化する暗号化処理の逆演算により復号を行うことを備え、
複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数によりにより入力値を暗号化する、復号化方法。

【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、暗号化装置、暗号化方法、復号化装置、及び復号化方法に関する。
【背景技術】
【０００２】
従来、例えば下記の非特許文献１，２には、既存のブロック暗号をホワイトボックスモデルにおいても安全であるように変換する方法が記載されている。非特許文献１，２に記載された方法は、既存のアルゴリズム(ＤＥＳ,ＡＥＳ)のホワイトボックス実装に関し、演算を大きなテーブル参照に変換し、秘密鍵をテーブルに埋め込むことで、内部演算が外部から見られても安全性を保障する技術に関する。
【０００３】
非特許文献１，２に記載された方法では、秘密鍵の値はテーブルの中に含まれるため、鍵付のテーブルが生成される。そして、各テーブルの安全性を上げるために、テーブルの前後には、秘密の非線形関数が加えられる。また、暗号化アルゴリズムＥの前後には、エクスターナルエンコーディング（External Encoding）として、関数ＩＮと関数ＯＵＴが加えられる。
【０００４】
また、下記の非特許文献３には、Decompositionが難しいと期待されている問題でtableを構成し、ホワイトボックスモデルにおいても安全なブロック暗号を構成する方法が記載されている。具体的に、非特許文献に記載された方法は、ホワイトボックス用のテーブルを秘密の非線形関数(S layer)と秘密の線形関数(A layer)を重ねて構成させる方法であり、具体的には、３層の秘密の線形関数（Ａｌａｙｅｒ）と２層の秘密の非線形関数（Ｓ ｌａｙｅｒ）から構成される。
【先行技術文献】
【非特許文献】
【０００５】
【非特許文献１】S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot “A white-box DESimplementation for DRM applications. ” DRM 2002

【非特許文献２】S. Chow, P. Eisen, H. Johnson, P.C. van Oorschot “White-BoxCryptography and an AES Implementation?” SAC 2002
【非特許文献３】A. Biryukov, C. Bouillaguet, D. Khovratovich:“CryptographicSchemes Based on the ASASA Structure: Black-Box, White-Box, and Public-Key”,ASIACRYPT 2014
【発明の概要】
【発明が解決しようとする課題】
【０００６】
しかし、非特許文献１，２に記載された方法では、安全性を上げるために、暗号化関数Ｅの入出力にエクスターナルエンコーディングとして新たな関数を付加する必要があり、エクスターナルエンコーディングを無くすと安全性が大幅に低下する問題がある。また、オリジナルの暗号化関数Ｅとは異なる関数になる。さらに、例えエクスターナルエンコーディングを付加したとしても、Practicalな攻撃方法が提案されている。
【０００７】
より具体的には、非特許文献１，２に記載された技術はＡＥＳ，ＤＥＳに対するホワイトボックス技術であるが、エクスターナルエンコーディングを用いた場合，ホワイトボックスモデルでの暗号化は、異なる暗号アルゴリズムになってしまう。このため、純粋な意味でＡＥＳのホワイトボックス実装とは言えず、異なる暗号アルゴリズムを本質的には実装していることになる。更に、エンコーディングされた平文を元に戻す際に、同一デバイス内の他のセキュアドメインでエンコーディングされた平文から正規の平文に戻す作業が必要である。つまり、ホワイトボックス実装が必要な環境において、セキュアに演算できる領域が必要ということであり、ホワイトボックスモデルとは矛盾しており、アプリケーションが限定される問題がある。また、エクスターナルエンコーディングを用いない場合は、一番初めと最後のラウンドにエクスターナルエンコーディングを一部用いることができず、安全性が大きく低下する問題がある。例えエクスターナルエンコーディングを付加したとしても、Practicalな攻撃方法が提案されている。
【０００８】
また、非特許文献３に記載された方法では、定量的に安全性を評価できない問題があり、Practicalな攻撃方法が既に提案されている。繰り返し段数を増やすことで安全性を上げるアプローチも考えられるが、秘密の非線形関数(S layer)と秘密の線形関数(A layer)を重ねて関数を構成する方法に関しては研究の歴史が浅く、定量的に安全性を評価することは困難である。
【０００９】
そこで、ホワイトボックスモデルにおいて、秘密鍵の秘匿性を保障し、安全に暗号化演算をすることが望まれていた。
【課題を解決するための手段】
【００１０】
本開示によれば、入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化するデータ暗号化部を備え、複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいて入力値を暗号化するテーブル化された暗号化関数を有する、暗号化装置が提供される。
【００１１】
また、本開示によれば、入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化することを備え、複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、暗号化方法が提供される。

【００１２】
また、本開示によれば、入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化する暗号化処理の逆演算により復号を行うデータ復号化部を備え、複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、復号化装置が提供される。
【００１３】
また、本開示によれば、入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化する暗号化処理の逆演算により復号を行うことを備え、複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数によりにより入力値を暗号化する、復号化方法が提供される。
【発明の効果】
【００１４】
以上説明したように本開示によれば、ホワイトボックスモデルにおいて、鍵の秘匿性を実現でき，安全な暗号化演算により、データの秘匿性を高めることが可能となる。
なお、上記の効果は必ずしも限定的なものではなく、上記の効果とともに、または上記の効果に代えて、本明細書に示されたいずれかの効果、または本明細書から把握され得る他の効果が奏されてもよい。
【図面の簡単な説明】
【００１５】
【図１】共通鍵ブロック暗号を示す模式図である。
【図２】暗号化を行うブロック（暗号関数Ｅ）の内部構成を示す模式図である。
【図３】Ｆｅｉｓｔｅｌ構造を示す模式図である。
【図４】ＳＰＮ構造を示す模式図である。
【図５】共通鍵ブロック暗号によって構成されるブラックボックスモデルを示す模式図である。
【図６】共通鍵ブロック暗号によって構成されるホワイトボックスモデルを示す模式図である。
【図７】本実施形態に係る暗号化技術の概要を示す模式図である。
【図８】具体例（Ｂ）、具体例（Ｃ）、具体例（Ｄ）、具体例（Ｅ）のそれぞれについいて、全体構成、Ｆ関数／Ｓ関数の種類、テーブルサイズ可変の可否、を示す模式図である。
【図９】暗号タイプに応じた処理を示すフローチャートである。
【図１０】具体例（Ｂ）を示す模式図である。
【図１１】Ｆ関数の構成を示す模式図である。
【図１２】図１０において、ｎ＝ｎ’＝１２８、ｃ＝１、ｄ＝１６の場合の全体構成を示す模式図である。
【図１３】図１２の例のＦ関数の構成を示している。
【図１４】図１０において、ｎ＝１２８、ｃ＝１、ｄ＝８の場合の全体構成を示す模式図である。
【図１５】図１０において、ｎ＝１２８、ｃ＝１、ｄ＝４の場合の全体構成を示す模式図である。
【図１６】図１０において、ｎ＝１２８、ｃ＝３、ｄ＝１６の場合の全体構成を示す模式図である。
【図１７】１つのラウンド内に２つＦ関数がある例であって、ｎ＝１２８、ｄ＝４の例を示す模式図である。
【図１８】具体例（Ｃ）を示す模式図である。

【図１９】図１８に示したＳ関数のそれぞれの構成を示す模式図である。
【図２０】図１８において、ｎ＝１２８、ｄ＝８の場合を示す模式図である。
【図２１】具体例（Ｄ）を示す模式図である。
【図２２】具体例（Ｅ）を示す模式図である。
【図２３】本実施形態に係る暗号化による安全性を説明するための模式図である。
【図２４】本実施形態に係る暗号化による安全性を説明するための模式図である。
【図２５】著作権保護技術（ＤＲＭ：Digital Rights Management）への適用例を示す模式図である。
【図２６】図２５をより詳細に示す模式図である。
【図２７】ＮＦＣのエミュレーションを利用した支払システムへの適用例を示す模式図である。
【図２８】図２７をより詳細に示す模式図である。
【図２９】メモリリークに対しても安全な方式を示す模式図である。
【図３０】サイドチャンネル攻撃に対して安全な暗号化の例を示す模式図である。
【発明を実施するための形態】
【００１６】
以下に添付図面を参照しながら、本開示の好適な実施の形態について詳細に説明する。なお、本明細書及び図面において、実質的に同一の機能構成を有する構成要素については、同一の符号を付することにより重複説明を省略する。
【００１７】
なお、説明は以下の順序で行うものとする。
１．前提となる技術
２．本実施形態の概要
３．具体的構成例
３．１．具体例（Ｂ）
３．２．具体例（Ｃ）
３．３．具体例（Ｄ）
３．４．具体例（Ｅ）
４．復号化のための構成例
５．既存技術との相違
５．１．既存技術１との違い
５．２．既存技術２との違い
６．本実施形態に係る暗号化による安全性について
７．本実施形態が適用されるアプリケーションの例
【００１８】
１．前提となる技術
暗号化と復号化に同じ鍵を用いる共通鍵ブロック暗号の技術が知られている。図１は、共通鍵ブロック暗号を示す模式図であって、ｋビットの鍵長に対応したｎビット共通鍵ブロック暗号アルゴリズムＥを示している。暗号化の際には、平文Ｐ（ｎビット）からｋビットの秘密鍵Ｋを用いて暗号関数Ｅにより暗号文Ｃ（ｎビット）が生成される。復号化の際には、暗号文Ｃ（ｎビット）からｋビットの秘密鍵Ｋを用いて復号関数Ｄ（＝E^{-1}）により平文Ｐ（ｎビット）が生成される。このような共通鍵ブロック暗号では、例えば図１中に示すような通信路にデータが伝送される場合に、盗聴者（以下、攻撃者とも称する）に対して平文の秘匿性を実現できる。
【００１９】
平文Ｐと暗号文Ｃのビット長をブロックサイズと称し、ここではｎで表す。ｎは任意の整数値を取りうるが、通常、ブロック暗号アルゴリズムごとに予め１つに決められている。ブロック長がｎのブロック暗号のことをｎビットブロック暗号と称する場合がある。秘密鍵Ｋのビット長をｋで表し、鍵のビット長ｋは任意の整数値を取りうる。共通鍵ブロック暗号アルゴリズムは、１つまたは複数の鍵サイズに対応することになる。例えば、ある

ブロック暗号アルゴリズムＡはブロックサイズｎ＝１２８であり、ｋ＝１２８、またはｋ＝１９２、またはｋ＝２５６の鍵サイズに対応するという構成もあり得る。

【００２０】

暗号化アルゴリズムＥに対応する復号アルゴリズムＤは、暗号化アルゴリズムＥの逆関数E^{-1}と定義でき、入力として暗号文Ｃと鍵Ｋを受け取り、平文Ｐを出力する。

【００２１】

図２は、暗号化を行うブロック（暗号関数Ｅ）の内部構成を示す模式図である。暗号関数Ｅは、鍵スケジュール部１００とデータ暗号化部２００とから構成される。鍵スケジュール部１００は、秘密鍵Ｋを入力とし、ある定められたステップによりビット長を拡大してできた拡大鍵Ｋ’(ビット長ｋ’)を出力する。データ暗号化部２００は、平文Ｐを受け取り、鍵スケジュール部から拡大された拡大鍵Ｋ’を受け取ってデータの変換を行い、暗号文Ｃを出力する。データ暗号化部２００は、拡大鍵Ｋ’から得られるラウンド関数を繰り返し処理することで、暗号化を行う。

【００２２】

データ暗号化部２００は、処理単位であるラウンド関数に分割できるものとする。ラウンド関数は入力として２つのデータを受け取り、内部で処理を施したのち、１つのデータを出力する。入力データの一方は暗号化途中のｎビットデータであり、あるラウンドにおけるラウンド関数の出力が次のラウンド関数の入力として供給される。入力データの他方は鍵スケジュール部１００から出力された拡大鍵Ｋ’の一部のデータであり、この鍵データのことをラウンド鍵と称する。また、ラウンド関数の総数を総ラウンド数と称する。総ラウンド数は、暗号アルゴリズムごとに予め定められている値である。ここでは、総ラウンド数をＲで表す。データ暗号化部２００の入力側から１ラウンド目の入力データをX_1とし、ｉ番目のラウンド関数に入力されるデータをＸｉ、ラウンド鍵をＲＫｉとすると、データ暗号化部２００の構成は図２のように示される。

【００２３】

ブロック暗号アルゴリズムに応じてラウンド関数はさまざまな形態を取り得る。ラウンド関数はその暗号アルゴリズムが採用する構造によって分類できる。代表的な構造として、ここではＳＰＮ構造、Ｆｅｉｓｔｅｌ構造、拡張Ｆｅｉｓｔｅｌ構造を例示する。

【００２４】

図３は、Ｆｅｉｓｔｅｌ構造を示す模式図である。また、図４は、ＳＰＮ構造を示す模式図である。図３に示すＦｅｉｓｔｅｌ構造の基本的な構成例では、各ラウンド関数において、ｎビットの入力データX_iが上位ｎ／２ビットと下位ｎ／２ビットに分割され、各ラインのデータのサイズはｎ／２ビットとなる。ここで、Ｆ関数には上位ｎ／２ビットが入力されて、ｎ／２ビットが出力される。この出力は下位ｎ／２ビットにそれぞれ排他的論理和される。その後、データの左右を入れ替えたものを出力データX_{i+1}とする。Ｆ関数は非線形関数をもとに構成される。ＳＰＮ構造とは異なり、Ｆ関数は置換である必要はない。一般的に、Ｆ関数は、ブロック暗号から生成されることはなく、計算の軽い非線形演算で生成されるが、本実施形態では、Ｆ関数をブロック暗号から生成する。

【００２５】

拡張Ｆｅｉｓｔｅｌ構造（一般化Ｆｅｉｓｔｅｌ構造）は、Ｆｅｉｓｔｅｌ構造ではデータ分割数が２であったものを３以上に分割する形に拡張したものである。分割数をｄとすると、分割数ｄによって様々な拡張Ｆｅｉｓｔｅｌ構造を定義することができる。Ｆ関数の入出力のサイズが相対的に小さくなるため、小型実装に向いている。また、各ラウンド関数において、複数のＦ関数を持つことができる。

【００２６】

後述する図１７では、ｄ＝４であり、且つ１つのラウンド内に２つのＦ関数が並列に適用される場合の拡張Ｆｅｉｓｔｅｌ構造の一例を示している。この例では、第１Ｆ関数と第２Ｆ関数の各々において、$RK1_i$と$RK2_i$を鍵入力とする。また、後述する図１４では、ｄ＝８であり、且つ１つのラウンド内に１つのＦ関数が適用される場合の拡張Feistel構造の一例を示している。この例では、Ｆ関数への入力サイズがｎ/８ビットであり、

Ｆ関数からの出力サイズが７ｎ／８ビットであり、出力は７つのｎ／８ビットのデータに分割され、残りの７つの１６ビットデータに排他的論理和される。なお、ｎ＝１２８　ｂｉｔとする。

【００２７】

また、図４に示すＳＰＮ構造の基本的な構成例では、ｎビットの入力データの全てに対して、ラウンド鍵との排他的論理和演算、非線形変換、線形変換処理などが適用される。非線形変換部をＳ層(Substitution-layer)、線形変換部をＰ層(Permutation-layer)と称し、それぞれは置換(全単射関数)である。各ラウンド関数において、ｎビットの入力データＸｉがｄ通りのデータに分割され、各ラインのデータのサイズはｎ／ｄ［ｂｉｔ］となる。ここで、非線形変換演算をＳ関数と定義し、各データ毎にｎ／ｄ［ｂｉｔ］の入出力の非線形変換演算Ｓ層(Substitution-layer)が実行される。その後、線形変換Ｐ層(Permutation-layer)としてｎビットの入出力線形変換Ｌが実行される。なお、線形変換演算をＬ関数と定義する。

【００２８】

ブロック暗号の安全性モデルとして、ブラックボックスモデルとホワイトボックスモデルがある。図５は、共通鍵ブロック暗号によって構成されるブラックボックスモデルを示す模式図である。ブラックボックスモデルでは、秘密鍵を求めようとする攻撃者の能力が、ブロック暗号の入出力を認識して自由にコントロール可能であるが、攻撃者はブロック暗号の中間値は認識できない。つまり、ブラックボックスモデルは、攻撃者がブロック暗号アルゴリズムの入力、出力である平文Ｐ、暗号文Ｃにしかアクセスすることができない安全性モデルである。攻撃者による攻撃は、攻撃者が平文Ｐと暗号文Ｃのペアの値を知っているのみである既知平文暗号文攻撃と、これに加えて攻撃者が値自体を自由にコントロールできる選択平文暗号文攻撃に分けることができる。ブラックボックスモデルでは，暗号演算自体は安全に実行され、攻撃者が暗号の中間値を見たり、改ざんすることができないことを想定している。ハードウェアサポートなどを利用し、暗号演算の耐タンパ性が保障できている場合等が対応する。ブラックボックス用の暗号アルゴリズムの実装方法をブラックボックス実装と称する。このようなブラックボックスモデルでは、攻撃者に秘密鍵を知られないように安全に設計することが可能である。ブラックボックスモデルにおいては、ブロック暗号は、秘密鍵Ｋを求めることが計算量的に困難であり（鍵回復攻撃耐性）、ブロック暗号と擬似ランダム鍵付置換を識別するのが計算量的に困難である（識別攻撃耐性）ように設計される。ブラックボックスモデルで安全なブロック暗号は、例えば、ＡＥＳ,ＣＬＥＦＩＡ,ＰＲＥＳＥＮＴ,Ｐｉｃｃｏｌｏなどの暗号化技術により実現可能である。

【００２９】

図６は、共通鍵ブロック暗号によって構成されるホワイトボックスモデルを示す模式図である。ホワイトボックスモデルは、ブラックボックスモデルよりも強い攻撃者を想定した安全性モデルであり、攻撃者がブロック暗号アルゴリズムの入力、出力である平文Ｐ、暗号文Ｃのみならず、演算の中間値にも自由にアクセスできる。ホワイトボックスモデルでは、攻撃者はブロック暗号の入力である平文Ｐ、暗号文Ｃを自由にコントロールでき、更に攻撃者が演算中の任意の中間値を見ることや、改ざんできることを想定している。ハードウェアのサポートがないオールソフトウェアなどの、実装上の制約により、耐タンパが保障できないケースに対応している。また、バッファオーバーフロー等の実装上の脆弱性やマルウェア等により中間値が漏れてしまう場合にも対応する。ホワイトボックス用の暗号アルゴリズムの実装方法をホワイトボックス実装と称する。ホワイトボックス実装によれば、ソフトウエアのみでブロック暗号を構成することも可能である。

【００３０】

このように、ホワイトボックスモデルでは、攻撃者の能力が、ブロック暗号の入出力を認識して自由にコントロール可能であり、ブロック暗号の中間値を認識して自由にコントロール可能である。ホワイトボックスモデルにおいては、攻撃者が鍵Ｋを求めることは計算量的に困難であることが求められる。また、鍵Ｋを求める変わりにコード自体を直接用

いて大きな鍵として用いる攻撃（ｃｏｄｅ　ｌｉｆｔｉｎｇと呼ばれる）に対しての耐性も求められる。ブロック暗号の中間値を攻撃者が認識可能なホワイトボックスモデルでは、このような攻撃に対して定量的な安全性をもつ必要がある。

【００３１】

２．本実施形態の概要

本実施形態では、上述したホワイトボックスモデルにおいて、信頼できない実行環境において、安全に暗号復号を行う技術、秘密鍵を守る技術を提案する。信頼できない環境として、秘密鍵を安全に保管できない場合、暗号演算の中間値を攻撃者が認識できる場合が挙げられる。

【００３２】

図７は、本実施形態に係る暗号化技術の概要を示す模式図であって、基本となる構成例（Ａ）に係る暗号化装置を示している。ブロック暗号Ｅを複数のテーブル３００から構成し、各テーブルをブラックボックスモデルで安全なブロック暗号Ｅ’（内部ブロック暗号）として構成する。これにより、安全なブロック暗号Ｅを構成することができる。ホワイトボックス実装では、ブロック暗号Ｅ’からなる部品の一部、または全てをテーブル化して実装する。ブロック暗号Ｅ’のアルゴリズムはユーザが自由に選択可能とする。なお、暗号化装置は、ＣＰＵなどの中央演算処理装置と、これを機能させるためのプログラムから構成することができる。この場合に、そのプログラムは、暗号化装置が備えるメモリなどの記録媒体に格納されることができる。また、ブロック暗号を構成するテーブルは、暗号化装置が備える記録媒体に格納されることができる。

【００３３】

このように、本実施形態の基本となる構成例（Ａ）では、ブラックボックスモデルで安全なブロック暗号Ｅ’を構成要素（部品）として、ホワイトボックスモデルで安全なブロック暗号Ｅを構成する。内部ブロック暗号Ｅ’のアルゴリズムは、ユーザが自由に選択でき、入力として受け取る。ホワイトボックス実装では、内部ブロック暗号Ｅ’ベースの関数を鍵に依存させ、一部もしくは全てがテーブルで実装される。つまり、鍵スケジュール部１００から出力された拡大鍵Ｋ’により内部ブロック暗号Ｅ’を生成してテーブル化する。テーブル化することにより、その都度暗号化演算が行われる場合と比較して、鍵の秘匿性を大幅に高めることできる。

【００３４】

また、構成例（Ａ）の具体例（Ｂ）として、ブロック暗号ＥがＦｅｉｓｔｅｌ構造からなり、一種類の入出力サイズのＦ関数から構成され、Ｆ関数は内部ブロック暗号Ｅ’をもとに生成される。ここでＦ関数は、内部ブロック暗号Ｅ’の入力の一部を固定、出力の一部を破棄することによってＥ’から変換される。ホワイトボックス実装では、Ｆ関数すべてがテーブルで実装される。

【００３５】

また、構成例（Ａ）の具体例（Ｃ）として、ブロック暗号ＥがＳＰＮ構造からなり、一種類の入出力サイズのＳ関数から構成され、Ｓ関数は内部ブロック暗号Ｅ’をもとに生成される。ここでＳ関数は、同じサイズの内部ブロック暗号から構成される。ホワイトボックス実装では、Ｓ関数すべてがテーブルで実装される。

【００３６】

また、構成例（Ａ）の具体例（Ｄ）として、ブロック暗号Ｅが拡張Ｆｅｉｓｔｅｌ構造からなり、複数種類の入出力サイズのＦ関数から構成され、Ｆ関数は内部ブロック暗号Ｅ’をもとに生成される。ここで、Ｆ関数は、内部ブロック暗号の入力の一部を固定、出力の一部を破棄することによって生成される。ホワイトボックス実装では、一部若しくは全てがテーブル実装される。

【００３７】

また、構成例（Ａ）の具体例（Ｅ）として、ブロック暗号ＥがＳＰＮ構造からなり、複数種類の入出力サイズのＳ関数から構成され、Ｓ関数は内部ブロック暗号Ｅ’をもとに生成する。ここで、Ｓ関数は、同じサイズの内部ブロック暗号から構成される。ホワイトボ

ックス実装では、一部若しくは全てがテーブル実装される。

【００３８】

図８は、具体例（Ｂ）、具体例（Ｃ）、具体例（Ｄ）、具体例（Ｅ）のそれぞれについいて、全体構成、Ｆ関数／Ｓ関数の種類、テーブルサイズ可変の可否、を示す模式図である。

【００３９】

また、図９は、暗号タイプに応じた処理を示すフローチャートである。図９において、先ずステップＳ１０では、内部ブロック暗号Ｅ’に鍵Ｋを依存させ、鍵付関数E'_Kを生成する。次のステップＳ１２では、暗号タイプを判定し、Ｆｅｉｓｔｅｌ構造の場合はステップＳ１４へ進む。ステップＳ１４では、Ｆ関数をE'_Kから生成する。次のステップＳ１６では、Ｆ関数をテーブル化する。次のステップＳ１８では、Ｆｅｉｓｔｅｌ構成でテーブルを接続し、暗号関数Ｅを生成する。

【００４０】

また、ステップＳ１２でＳＰＮ構造の場合はステップＳ２０へ進み、Ｓ関数をE'_Kから生成する。次のステップＳ２２では、Ｓ関数をテーブル化する。次のステップＳ２４では、ＳＰＮ構成でテーブルを接続し、暗号関数Ｅを生成する。ステップＳ１８，Ｓ２４の後はステップＳ２６へ進み、テーブルベースの関数からコード生成を行う。これにより、ホワイトボックス暗号化コードが生成される。

【００４１】

３．具体的構成例

以下では、具体例（Ｂ）、具体例（Ｃ）、具体例（Ｄ）、具体例（Ｅ）の構成例とその効果について詳細に説明する。ここで、内部ブロック暗号Ｅ’は、ｎ’ビットブロック暗号であり、ブラックボックスモデルにおいて安全であるものとする。

【００４２】

３．１．具体例（Ｂ）

図１０は、具体例（Ｂ）を示す模式図であって、一般化Ｆｅｉｓｔｅｌ構造による構成例を示している。図１０に示す例では、ｎビットの入力データＸｉがｄ通りのデータに分割され、各ラインのデータのサイズはｎ／ｄビットとなる。ここで、Ｆ関数はｃ×ｎ／ｄｂｉｔ入力、（ｄ－ｃ）×（ｎ／ｄ）（＝ｎ－（ｃ×ｎ／ｄ））［ｂｉｔ］出力で、ｃ通りのラインのデータが入力され、出力はｄ－ｃ通りのｎ／ｄ［ｂｉｔ］のデータに分割され、残りのｄ－ｃ通りのラインにそれぞれ排他的論理和される。Ｆ関数は内部ブロック暗号Ｅ’をもとに構成される。ここでＥ’のブロックサイズｎ’が、ｎ’＞（ｄ－ｃ）×（ｎ／ｄ）　かつ　ｎ’＞ｃ×（ｎ／ｄ）を満たすものとする（条件１）。図１０に示すように、ブロック暗号Ｅ’へ入力されたビットの値は、排他論理和により得られたビットの値よりも下位ビットとして出力される。

【００４３】

図１１は、Ｆ関数の構成を示す模式図である。ｎ’ビットの内部ブロック暗号Ｅ’から、ｃ×ｎ／ｄ［ｂｉｔ］入力、（ｄ－ｃ）×（ｎ／ｄ）［ｂｉｔ］出力のＦ関数の構成方法は以下の通りである。先ず、図１１に示すように、内部ブロック暗号Ｅ’の入力ｎ’［ｂｉｔ］のうち、任意のｎ’－（ｃ×ｎ／ｄ）［ｂｉｔ］を定数値（例えばall 0）に固定し、入力サイズをｃ×ｎ／ｄにする。次に、出力の任意の（ｃ×ｎ／ｄ）［ｂｉｔ］を破棄（ｄｉｓｒｅｇａｒｄ）することにより、出力サイズをｎ’－（ｃ×ｎ／ｄ）にする。このような、内部ブロック暗号Ｅ’に対する一部の入力ビット固定、出力の一部破棄により、条件１を満たす任意の内部ブロック暗号Ｅ’からｆ関数を構成する。テーブル化により、Ｆ関数はｎ’ビットの入出力に対応するテーブルから構成される。例えば、８ビットの入出力の場合、入力値（０～２５５）に対して出力値を対応付けしたテーブルが生成される。このテーブルに対し、一部の入力ビット固定、出力の一部破棄を行うことで、８ビット入力、１２０ビット出力などの入出力ビット数の調整を行うことができる。ここで、各ラウンドにおいてＦ関数を変更するため、ｎ’－（ｃ×ｎ／ｄ）ビットの出力にラウンド固有の定数を排他論理和（ＸＯＲ）する。例えば、ラウンド固有の定数は、例えばラ

ウンド数とし、ラウンド数をＸＯＲする。ラウンド数４の場合は４をＸＯＲすることになる。但し、この排他論理和はテーブル参照後に行われるため、この演算自体はテーブルには含めない。これにより、一通りのＦ関数テーブルで、ラウンド毎に異なるＦ関数を表現することができる。従って、各ラウンド関数のＦ関数自体は共通に構成することも可能であり、テーブルを格納するメモリ領域を大幅に削減することができる。

【００４４】

図１２～図１５は、具体的な構成例を示す模式図である。図１２はｎ＝ｎ’＝１２８、ｃ＝１、ｄ＝１６の場合の全体構成を示しており、図１３は、図１２の例のＦ関数の構成を示している。また、図１４は、ｎ＝１２８、ｃ＝１、ｄ＝８の場合、図１５はｎ＝１２８、ｃ＝１、ｄ＝４の場合、図１６はｎ＝１２８、ｃ＝３、ｄ＝１６の場合、をそれぞれ示している。

【００４５】

図１７は、１つのラウンド内に２つＦ関数がある例であって、ｎ＝１２８、ｄ＝４の例を示す模式図である。以上の全ての例において、Ｆ関数は、ホワイトボックス実装ではテーブル実装される。図１２、図１４、図１５、図１６の例において、テーブルサイズ（Ｆ関数のサイズ）は、それぞれ、約３．８４［ｂｙｔｅ］、９１８［Ｋｂｙｔｅ］、５１．５［Ｇｂｙｔｅ］、２１８［Ｍｂｙｔｅ］程度である。

【００４６】

３．２．具体例（Ｃ）

図１８は、具体例（Ｃ）を示す模式図であって、ＳＰＮ構造による構成例を示している。図１８に示す例では、ｎビットの入力データＸｉがｄ通りのデータに分割され、各ラインのデータのサイズはｎ／ｄ［ｂｉｔ］である。ここで、各データ毎にｎ／ｄ［ｂｉｔ］の入出力のＳ関数による演算（非線形変換演算Ｓ層(Substitution-layer)）が実行される。その後、Ｌ関数による演算（線形変換Ｐ層(Permutation-layer)）としてｎ－ｂｉｔ入出力線形変換が実行される。ここで、Ｓ関数とＬ関数（入出力線形変換Ｌ）は全単射関数であり、Ｌ関数はラウンド定数演算を含む。Ｓ関数は、内部ブロック暗号Ｅ’をもとに構成されるが、Ｓ関数は全単射関数である必要があり、図１１のように内部ブロック暗号Ｅ’の入力ビット固定、出力の一部の破棄による変換では構成することができない。このため、ｎ／ｄ［ｂｉｔ］のブロック暗号を用いる必要がある。よって、内部ブロック暗号Ｅ’のブロックサイズｎ’の条件は、ｎ’＝ｎ／ｄとなる（条件２）。

【００４７】

図１９は、図１８に示したＳ関数のそれぞれの構成を示す模式図である。図１９に示すように、Ｓ関数を構成する内部ブロック暗号Ｅ’の入出力のサイズは、共にｎ／ｄ［ｂｉｔ］である。従って、例えば、８ビットの入出力の場合、入力値（０～２５５）に対して出力値を対応付けしたテーブルが生成され、このテーブルによりＳ関数の演算が行われる。線形変換演算を行うＬ関数は、例えば正方行列から構成される。Ｓ関数の入出力が８ビットの場合、Ｓ関数からの８ビットの出力がＬ関数に入力され、８ビットの値に対して８×８のマトリクスの正方行列を乗算することで、８ビットの値がＬ関数から出力される。このように、Ｌ関数は、Ｓ関数からの出力値を拡散する機能を有する。

【００４８】

図２０は、具体的な構成例を示す模式図であって、ｎ＝１２８、ｄ＝８の場合を示している。Ｓ関数は、ホワイトボックス実装ではテーブル実装される。図２０のテーブルサイズは約２５６［ｂｙｔｅ］程度である。Ｓ関数の場合も、図１１に示したＦ関数の場合と同様に、各Ｓ関数を変更するため、Ｓ関数の出力にラウンド固有の定数をＸＯＲすることができる。これにより、Ｓ関数自体を共通にすることができるため、テーブルを格納するメモリ領域を大幅に削減することができる。

【００４９】

３．３．具体例（Ｄ）

図２１は、具体例（Ｄ）を示す模式図であって、変形Ｆｅｉｓｔｅｌ構造による構成例を示している。図２１に示す例では、ｎビットの入力データがｄ通りのデータに分割され

、各ラインのデータのサイズはｎ／ｄであり、サイズの異なる４種類のＦ関数から構成されている。初めのラウンドではｎ／ｄ［ｂｉｔ］入力、(ｎ－ｎ／ｄ)［ｂｉｔ］出力のＦ関数、２番目のラウンドでは２ｎ／ｄ［ｂｉｔ］入力、（ｎ－２ｎ／ｄ)［ｂｉｔ］出力のＦ関数、３番目のラウンドでは３ｎ／ｄ［ｂｉｔ］入力、（ｎ－３ｎ／ｄ）［ｂｉｔ］出力のＦ関数、４番目のラウンドでは４ｎ／ｄ［ｂｉｔ］入力、（ｎ－４ｎ／ｄ）［ｂｉｔ］出力のＦ関数が用いられる。この４ラウンドを基本として、任意ラウンド繰り返す。図１１で示した方法と同様に、任意のサイズのＦ関数は、内部ブロック暗号Ｅ’から生成され、出力にラウンド定数がＸＯＲされる。

【００５０】

ホワイトボックス実装では、ユーザの求めるコード(テーブルサイズ)に応じて、これらのうちの一部若しくは全てがテーブルで実装される。ｎ＝１２８、ｄ＝１６の場合は、それぞれのラウンドのＦ関数のテーブルサイズは、初めのラウンドでは約３．８４［ｂｙｔｅ］、２番目のラウンドでは９１８［Ｋｂｙｔｅ］、３番目のラウンドでは２１８［Ｍｂｙｔｅ］、４番目のラウンドでは５１．５［Ｇｂｙｔｅ］となる。ユーザの要求に応じて、どのＦ関数をテーブル実装するか選択することにより、全体のコードサイズを調整可能である。例えば、４番目のラウンド関数はテーブル化せずに関数演算をその都度行うようにすれば、全体のコードサイズを抑えることができる。

【００５１】

３．４．具体例（Ｅ）

図２２は、具体例（Ｅ）を示す模式図であって、変形ＳＰＮ構造による構成例を示している。図２２に示す例では、ｎビットの入力データがｄ通りのデータに分割され、各ラインのデータのサイズはｎ／ｄであり、サイズの異なる３種類のＳ関数から構成されている。各ラウンドのＳ層は、ｎ／ｄ［ｂｉｔ］の入出力、２ｎ／ｄ［ｂｉｔ］の入出力、４ｎ／ｄ［ｂｉｔ］の入出力、のＳ関数が用いられる。ホワイトボックス実装では、ユーザの求めるコード(テーブルサイズ)に応じて、これらの一部若しくは全てがテーブルで実装される。例えば。ｎ＝１２８、ｄ＝８で、８［ｂｉｔ］、１６［ｂｉｔ］、３２［ｂｉｔ］のデータが実装されている場合を考える。それぞれのテーブルサイズは、２５６［ｂｙｔｅ］、１３２［Ｋｂｙｔｅ］、１７．２［Ｇｂｙｔｅ］となる。ユーザの要求に応じて、どのＳ関数をテーブル実装するか選択することにより、全体のコードサイズを調整可能である。

【００５２】

本実施形態によれば、ホワイトボックスモデルにおいて、key extractionの安全性は内部ブロック暗号Ｅ’のブラックボックスモデルでの鍵回復問題の安全性に帰着する。これは、ホワイトボックス実装において、内部ブロック暗号Ｅ’がテーブル実装されていることによるもので、攻撃者はホワイトモデルにおいても、テーブルの入出力にしかアクセスできない。これは、内部ブロック暗号Ｅ’のブラックボックスモデルとマッチする。内部状態（内部ブロック暗号Ｅ’）に信頼性の高い暗号(例えばＡＥＳ)を用ることで、ホワイトボックスモデルについても、内部ブロック暗号Ｅ’のブラックボックスモデルの鍵回復と同等の安全性を有することができる。

【００５３】

また、攻撃者は、鍵を知らない限りテーブルサイズを小さくすることはできない(Space-hardness)。内部ブロック暗号Ｅ’の鍵の情報を知らない限り、攻撃者はＥ’をテーブル演算以外で計算することはできない。このため、与えられたテーブルより小さいものに変換することはできない。これは、攻撃者がｃｏｄｅ　ｌｉｆｔｉｎｇ攻撃する際に、大容量のデータが必要なことを意味している。データサイズに比例してコード抜き取りに必要な時間は増加するため、ｃｏｄｅ　ｌｉｆｔｉｎｇ作業を非常に時間のかかる作業にしている。更に、もしコード全体自体をとったとしても、そのサイズを圧縮することができず、コード配布の際に大容量のデータを送付する必要があり、配布のリスクを低減することが可能である。

【００５４】

また、external encodingについても、External Encodingなしで安全性を保障可能である。

【００５５】

更に、実装要求に応じた様々なサイズのテーブルを構成することが可能である。具体例（Ｂ）、具体例（Ｃ）によれば、分割数ｄの値を変更することにより、任意のテーブルサイズのアルゴリズムを構成することができる。また、具体例（Ｄ）、具体例（Ｅ）によれば、分割数ｄの値、又は使用するＦ関数、Ｓ関数のサイズを適切に複数選択することで、同じアルゴリズムで複数のテーブルサイズの実装が可能となる。

【００５６】

また、ユーザは、内部ブロック暗号Ｅ’を自由に選択できる。内部ブロック暗号Ｅ’は、入出力サイズの条件(条件１、条件２)を満たす限り、自由に選択することが可能である。ブラックボックスで用いる場合、テーブル実装は不要で内部演算を直接演算できる。この場合、内部ブロック暗号Ｅ’を適切に選択することで、さまざまな実装ニーズに応えることが可能である。例えば、ＡＥＳを内部ブロック暗号Ｅ’として使用し、ＡＥＳ－ＮＩを用いることにより、ソフトウェアで非常に高速に実装でき、且つキャッシュタイミング攻撃に対しても安全に実装できる。また、ソフトウェアで軽量な暗号Ｐｉｃｃｏｌｏ,Ｐｒｉｄｅを用いることにより、ＲＡＭサイズ等の実装制約が大きい環境でも実装可能である。

【００５７】

４．復号化のための構成例

上述したように、暗号化アルゴリズムＥに対応する復号アルゴリズムＤは、暗号化アルゴリズムＥの逆関数$Ｅ^{-1}$と定義でき、入力として暗号文Ｃと鍵Ｋを受け取り、平文Ｐを出力する。復号アルゴリズムＤにおいても、ブラックボックス実装によりテーブルを構成することで、ブラックボックスモデルと同等の安全性を確保することが可能である。

【００５８】

５．既存技術との相違

以下では、本実施形態に係る技術と、前述した非特許文献１，２に記載された方法（既存技術１とする）、非特許文献３に記載された方法（既存技術３とする）との相違について説明する。

【００５９】

５．１．既存技術１との違い

既存技術1は、既存のＡＥＳ，ＤＥＳ等のアルゴリズムの実装方法であり、ホワイトボックス用暗号技術ではなく、ホワイトボックスモデルに対して安全ではないことが既に示されている。従って、ホワイトボックスモデルにおける安全性を大幅に向上した本実施形態に係る技術とは相違する。

【００６０】

５．２．既存技術２との違い

既存技術２では、内部ブロック暗号Ｅ’を自由に選ぶことはできず、内部ブロック暗号Ｅ’は予め決められているＡＳＡＳＡ構造に限定される。本実施形態では、ブラックボックスモデルで安全なブロック暗号でも安全性を満たすことを示し、内部ブロック暗号Ｅ’は、入出力サイズの条件（条件１、条件２）を満たす限り、自由に選択することが可能である。

【００６１】

これにより、ブラックボックス実装において、実装環境や求められる安全性に応じて、内部ブロック暗号Ｅ’を自由に選択可能となる。例えば、ＡＥＳを内部ブロック暗号Ｅ’として使い、ＡＥＳ－ＮＩを用いることにより、ソフトウェアで非常に高速な実装を行うことができ、且つキャッシュタイミング攻撃に対しても安全に実装できる。また、ソフトウェアで軽量な暗号Ｐｉｃｃｏｌｏ，Ｐｒｉｄｅを用いることにより、ＲＡＭサイズ等の実装制約が大きい環境でも実装可能である。また、ＡＳＡＳＡ構造はホワイトボックスモデルにおいては安全ではないが、本実施形態に係る方法では、ホワイトボックスモデルに

対しても安全性を保障することが可能である。

【００６２】

６．本実施形態に係る暗号化による効果について

図２３は、本実施形態に係る暗号化による安全性を説明するための模式図であって、図１１に示したＦｅｉｓｔｅｌ構造によるＦ関数をＡＥＳにより構成した例を示している。上述したように、ホワイトボックスモデルでは、攻撃者はテーブルの入出力にアクセス可能である。内部ブロック暗号Ｅ’のブラックボックスモデルと同じテーブルから鍵を求める問題（ホワイトボックスモデル）は、ＡＥＳの鍵回復攻撃(ブラックボックスモデル)と等しい。従って、本実施形態に係る暗号化によれば、ＡＥＳの鍵回復攻撃(ブラックボックスモデル)と同等の安全性を確保することが可能である。ホワイトボックスモデルにおいて、安全性は内部ブロック暗号Ｅ’のブラックボックスモデルでの鍵回復問題の安全性に帰着する。攻撃者は、鍵を知らない限りテーブルサイズを小さくすることはできない(Space-hardness)。

【００６３】

図２４は、本実施形態に係る暗号化による安全性を説明するための模式図であって、攻撃者が攻撃する際に必要となるデータ量を示している。攻撃のためには、非常に多くのデータを入手しなければ秘密鍵Ｋを取得することはできない。具体的には、１２８　ｂｉｔ鍵と比較した場合、$10^{4.4}$～$10^{10.5}$倍のデータ量が必要である。また、攻撃者がデータを入手できたとしても、圧縮することができないので、不正に配布する際に大容量のデータであるため抑止力になる。

【００６４】

また、本実施形態によれば、実装要求に応じた様々なサイズのテーブルを構成することが可能である。具体例(Ｂ),（Ｃ）の構成では、分割数ｄの数を変更することにより、任意のテーブルサイズのアルゴリズムを構成することができる。また、具体例（Ｄ），（Ｅ）の構成では、分割数ｄの数や使われるＦ関数、Ｓ関数のサイズを適切に複数選択することで、同じアルゴリズムで複数のテーブルサイズの実装が可能である。更に、テーブルの内部演算をユーザが自由に選択可能であり、ブラックボックス実装において最適なものを選択することが可能である。

【００６５】

７．本実施形態が適用されるアプリケーションの例

本実施形態に係る技術は、図１に示したような通信路におけるデータの秘匿性を実現する他、様々なアプリケーションに適用することができる。以下では、幾つかのアプリケーションの例を説明する。

【００６６】

図２５は、著作権保護技術（ＤＲＭ：Digital Rights Management）への適用例を示す模式図である。図２５に示すように、クラウド上のコンテンツサーバ４００で暗号化を行い、コンテンツ（暗号文Ｃ）がコンテンツサーバ４００からユーザデバイス４１０へ配信される。ユーザデバイス４１０は、パーソナルコンピュータ（ＰＣ）、スマートフォンなどの電子機器である。コンテンツ（暗号文Ｃ）はユーザデバイス４１０において復号される。

【００６７】

図２６は、図２５をより詳細に示す模式図である。コンテンツサーバ４００は、ホワイトボックス暗号化関数により映画、音楽などのコンテンツを暗号化する。また、コンテンツサーバ４００では、ライセンス生成器４０２によりライセンスを生成し、暗号化されたコンテンツと共にユーザデバイス４１０へ送る。ユーザデバイス４１０は、送られたライセンスをライセンス検証器４１２により検証し、ライセンスの検証に成功した場合は、ホワイトボックス復号関数により、暗号化されたコンテンツを復号する。

【００６８】

図２５及び図２６に示したような著作権保護技術では、ユーザデバイス４１０でコンテンツを復号する必要がある。この場合、仮に鍵Ｋが露呈した場合、コンテンツの不正配布

に繋がる。つまり、暗号化が安全でない環境では、ユーザデバイス４１０が信頼できない環境となる。本実施形態によれば、悪意のあるユーザがコンテンツの秘密鍵Ｋを取得することを、ホワイトボックス暗号化技術により確実に防止することが可能である。

【００６９】

図２７は、ＮＦＣのエミュレーションを利用した支払システムへの適用例を示す模式図である。図２７に示すように、このシステムでは、ＮＦＣの読取装置４２０にユーザデバイス４３０を近づけてエミュレーションを行う。ユーザデバイス４３０は、ホストＣＰＵ４３２、ＮＦＣコントローラ４３４、セキュアエレメント４３６を有する。

【００７０】

図２８は、図２７をより詳細に示す模式図である。クラウド上のサーバ４４０は、ユーザの証明用の情報（Ｃｒｅｄｅｎｔｉａｌ　ｉｎｆｏｒｍａｔｉｏｎ）と、支払情報（Ｐａｙｍｅｎｔ　ｉｎｆｏｒｍａｔｉｏｎ）を有する。ユーザデバイス４３０は、モバイル機器などの電子機器であり、サーバ４４０と暗号化通信を行い、証明用の情報をやり取りする。また、ユーザデバイス４３０は、読取装置４２０と暗号化通信を行い、証明用の情報をやり取りする。暗号化通信では、本実施形態に係るホワイトボックス暗号化により暗号化が行われる。このため、ユーザデバイス４３０は、ホワイトボックス暗号関数、復号関数を備えている。ホワイトボックス暗号化により暗号化を行うことで、支払いに関する証明のデータを守ることができ、ユーザデバイス４３０がセキュアエレメント４３６を備えていなくてもＮＦＣのエミュレーションが可能となる。

【００７１】

図２９は、メモリリークに対しても安全な方式を示す模式図である。このシステムでは、ソフトウェアの脆弱性(buffer overflow, heart bleed)やマルウェアにより、メモリが攻撃者にリークした場合も、安全性を保障する。マルウェアやメモリリークの脆弱性のあるデバイス４４５において、ホワイトボックス暗号化方式が有するＳｐａｃｅ　ｈａｒｄｎｅｓｓの性質により、数キロバイト、数ギガバイト以上のデータが漏れない限り安全性は低下しない。図２９の例において、コードサイズをＴとすると、Ｔ／４以上のデータが漏れない限り安全性は低下しない。なお、Ｓｐａｃｅ　ｈａｒｄｎｅｓｓとは、一定以上のサイズのメモリが漏れない限り暗号の安全性を保障できる技術である。この方法は、内部ネットワークから外部ネットワークからの通信量を制限している場合に特に有効である。

【００７２】

図３０は、サイドチャンネル攻撃に対して安全な暗号化の例を示す模式図である。ホワイトボックス暗号化方式は、通常はソフトウェア用であるが、Reconfigurable Hardware (FPGA)で実装することにより、ハードウェアでサイドチャンネルに安全な暗号方式として使うことができる。例えば図３０に示すＩＣカード４５０など、ハードウェアでサイドチャンネル攻撃を受ける可能性のあるデバイスにおいて、特に有効である。

【００７３】

以上、添付図面を参照しながら本開示の好適な実施形態について詳細に説明したが、本開示の技術的範囲はかかる例に限定されない。本開示の技術分野における通常の知識を有する者であれば、特許請求の範囲に記載された技術的思想の範疇内において、各種の変更例または修正例に想到し得ることは明らかであり、これらについても、当然に本開示の技術的範囲に属するものと了解される。

【００７４】

また、本明細書に記載された効果は、あくまで説明的または例示的なものであって限定的ではない。つまり、本開示に係る技術は、上記の効果とともに、または上記の効果に代えて、本明細書の記載から当業者には明らかな他の効果を奏しうる。

【００７５】

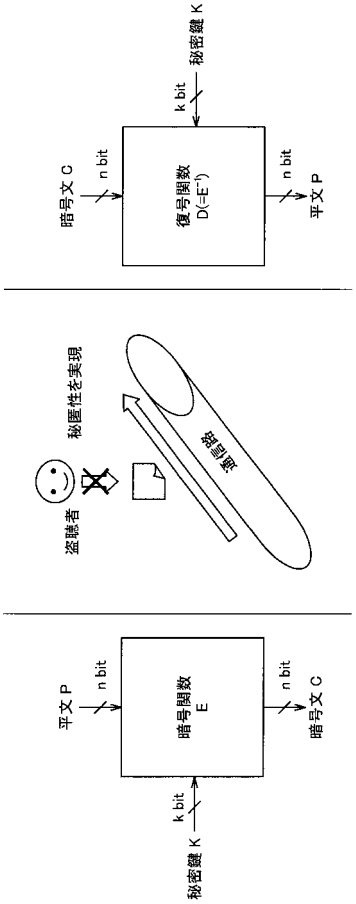
なお、以下のような構成も本開示の技術的範囲に属する。

（１）　入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルによ

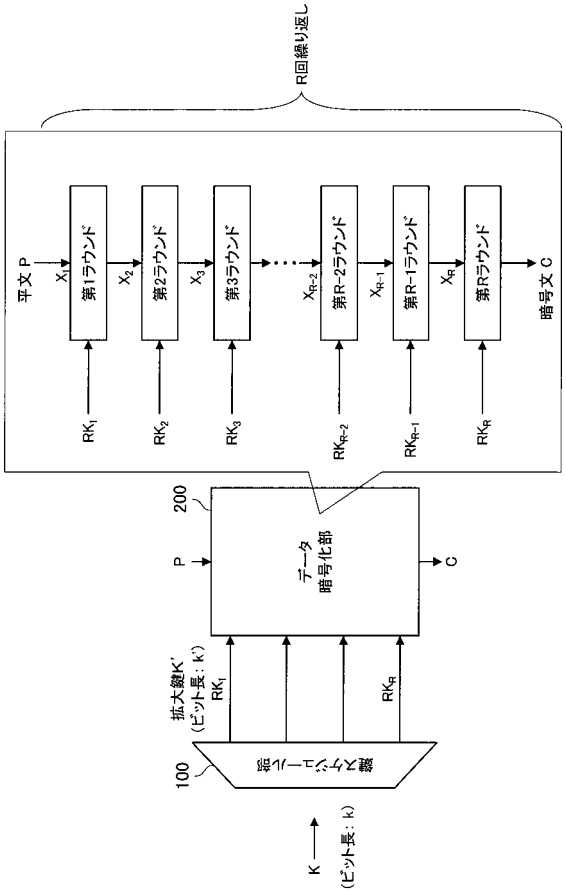
り暗号化するデータ暗号化部を備え、
　複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいて入力値を暗号化するテーブル化された暗号化関数を有する、暗号化装置。
（２）　前記暗号化関数には、前記ラウンド関数へ入力されるビットのうちの一部が入力され、
　前記暗号化関数は、前記暗号化関数へ入力可能なビットの一部を固定値とし、前記暗号化関数の出力値の一部を破棄することで、前記暗号化関数へ入力可能なビット数から前記暗号化関数へ入力されたビット数の差分に相当するビット数の出力値を出力する、前記（１）に記載の暗号化装置。
（３）　前記ラウンド関数は、前記ラウンド関数へ入力されるビットのうち前記暗号化関数に入力されなかったビットと、前記暗号化関数からの前記出力値のビットの排他論理和を演算する、前記（２）に記載の暗号化装置。
（４）　前記ラウンド関数は、前記暗号化関数へ入力されたビットの値と前記排他論理和により得られたビットの値を出力する、前記（３）に記載の暗号化装置。
（５）　前記ラウンド関数は、前記暗号化関数へ入力されたビットの値を前記排他論理和により得られたビットの値よりも下位ビットとして出力する、前記（４）に記載の暗号化装置。
（６）　前記ラウンド関数の出力と予め定められた所定値との排他論理和を演算し、得られた値を次のラウンド関数への入力又は前記データ暗号化部の出力とする、前記（２）～（５）のいずれかに記載の暗号化装置。
（７）　１の前記ラウンド関数が複数の前記暗号化関数を有する、前記（１）に記載の暗号化装置。
（８）　複数の前記ラウンド関数において、後段のラウンド関数ほど前記暗号化関数に大きなビットの入力値が入力される、前記（２）～（６）のいずれかに記載の暗号化装置。
（９）　１の前記ラウンド関数が複数の前記暗号化関数を有し、
　前記ラウンド関数へ入力されるビットが分割されて複数の前記暗号化関数へ入力され、
　複数の前記暗号化関数は非線形演算を行い、
　前記ラウンド関数は、複数の前記暗号化関数による前記非線形演算の結果を線形変換演算して出力する、前記（１）に記載の暗号化装置。
（１０）　複数の前記暗号化関数のそれぞれにおいて、入力されるビット数と出力されるビット数が同一である、前記（９）に記載の暗号化装置。
（１１）　複数の前記暗号化関数のそれぞれに入力されるビット数が異なる、前記（９）又は（１０）に記載の暗号化装置。
（１２）　前記暗号化関数は、前記データ暗号化部に対応する秘密鍵から生成される拡張鍵によって暗号化を行う、前記（１）～（１１）のいずれかに記載の暗号化装置。
（１３）　入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化することを備え、
　複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、暗号化方法。
（１４）　入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルにより暗号化する暗号化処理の逆演算により復号を行うデータ復号化部を備え、
　複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数により入力値を暗号化する、復号化装置。
（１５）　入力値を順次に暗号化処理する複数のラウンド関数の少なくとも一部がテーブル化され、前記ラウンド関数の入出力値を外部から認識可能なホワイトボックスモデルに

より暗号化する暗号化処理の逆演算により復号を行うことを備え、
　複数の前記ラウンド関数のそれぞれは、入出力値を外部から認識可能であり中間値は外部から認識できないブラックボックスモデルにおいてテーブル化された暗号化関数によりにより入力値を暗号化する、復号化方法。
【符号の説明】
【００７６】
　１００　　データ暗号化部
　３００　　テーブル

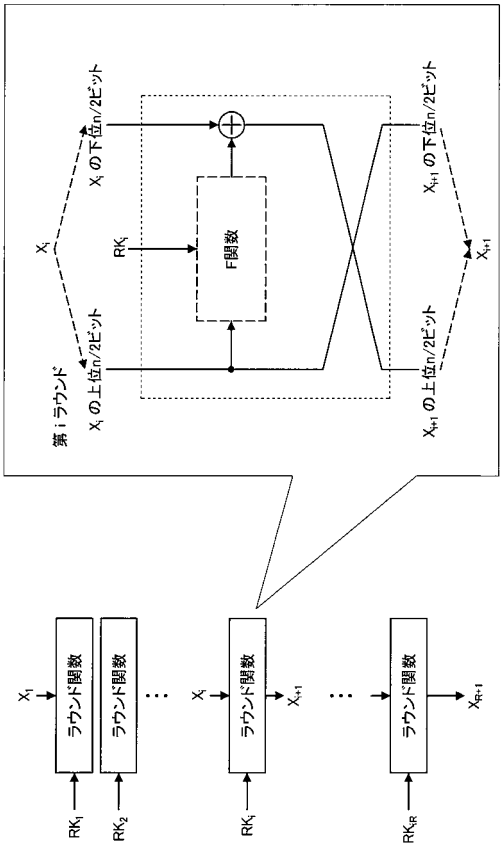
【図１】



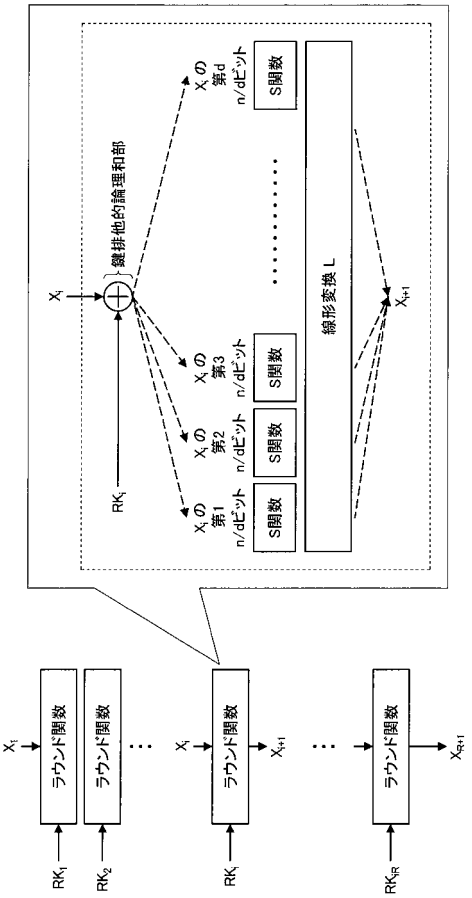
【図２】



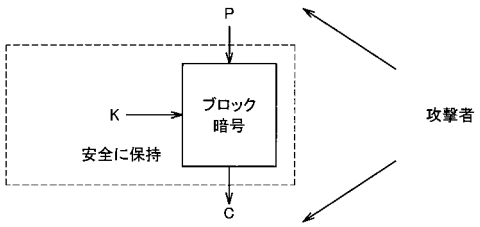
【図３】


第 i ラウンド
F関数
ラウンド関数
ラウンド関数
ラウンド関数
ラウンド関数

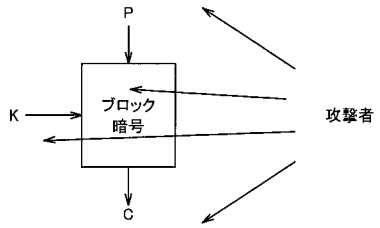
【図４】


鍵排他的論理和部
S関数
S関数
S関数
S関数
線形変換 L
ラウンド関数
ラウンド関数
ラウンド関数
ラウンド関数

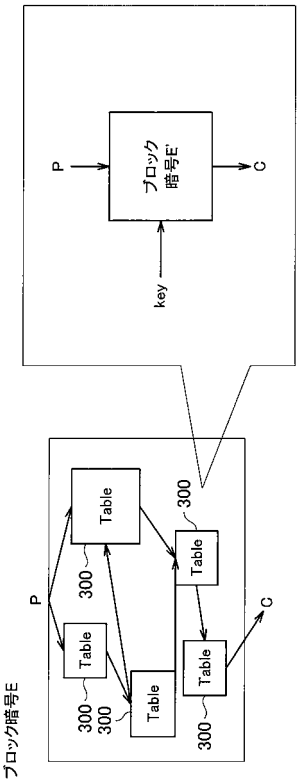
【図５】


P
K
ブロック
暗号
安全に保持
C
攻撃者

【図６】


P
K
ブロック
暗号
C
攻撃者

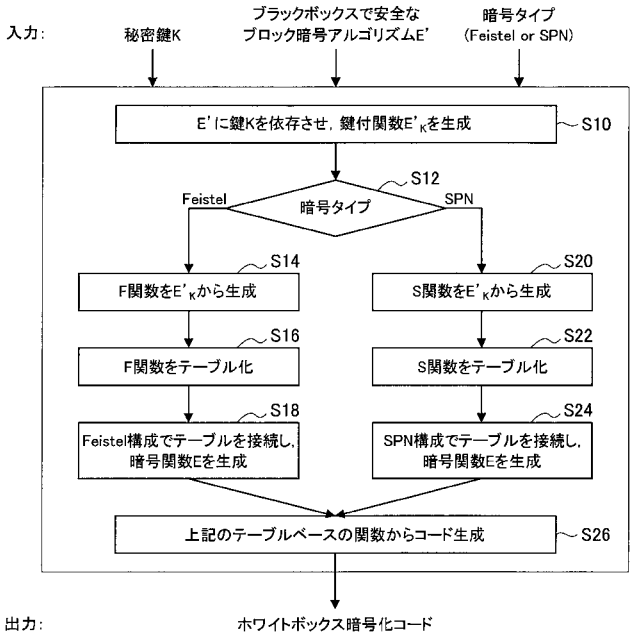
【図７】


P
ブロック
暗号E'
C
key
ブロック暗号E
Table
300

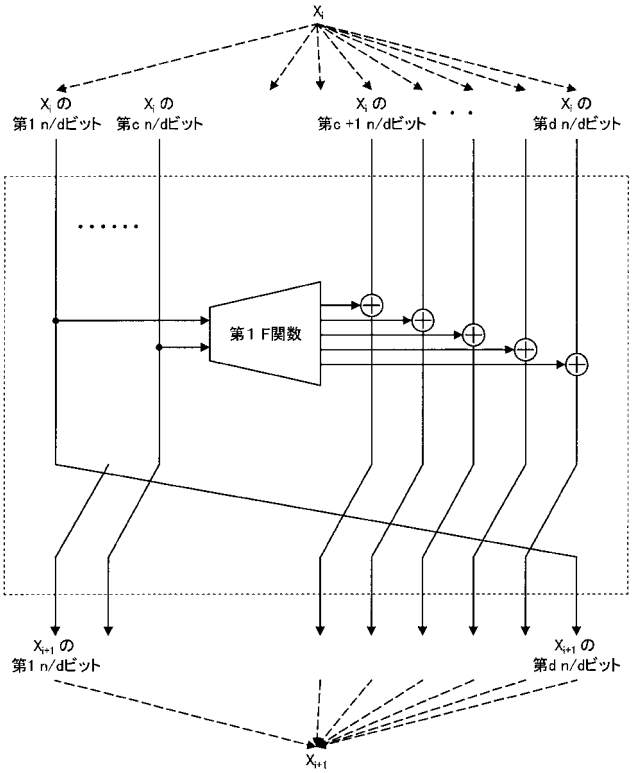
【図８】

	全体構成	F関数/S関数の種類	Table size可変
構成B	Feistel	1	No
構成C	SPN	1	No
構成D	Feistel	複数	Yes
構成E	SPN	複数	Yes

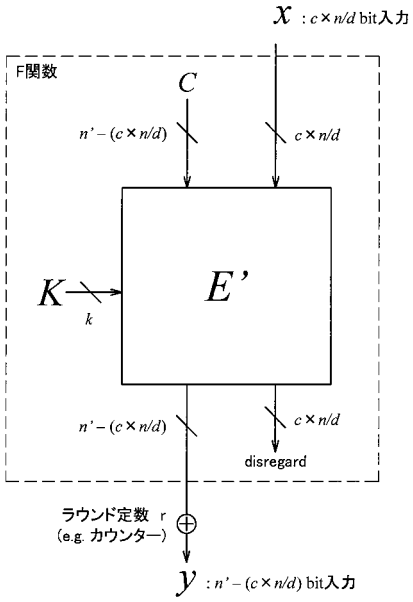
【図９】



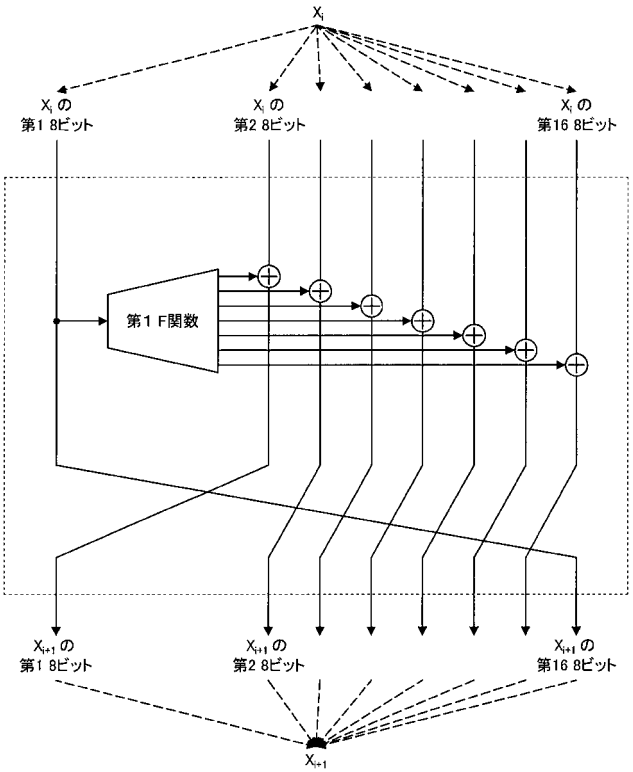
【図１０】



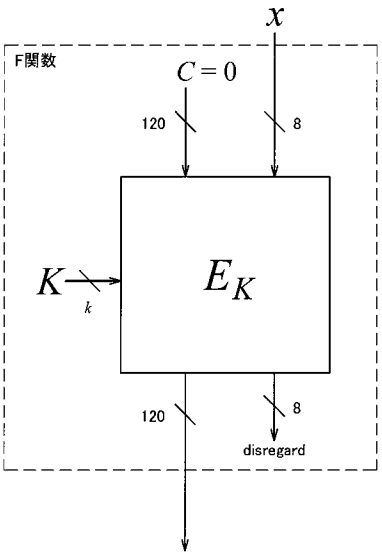
【図１１】



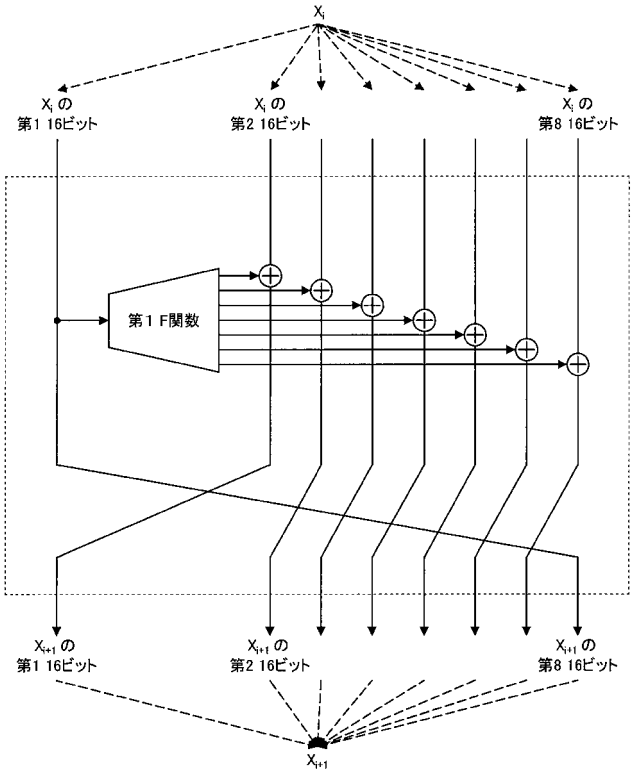
【図１２】



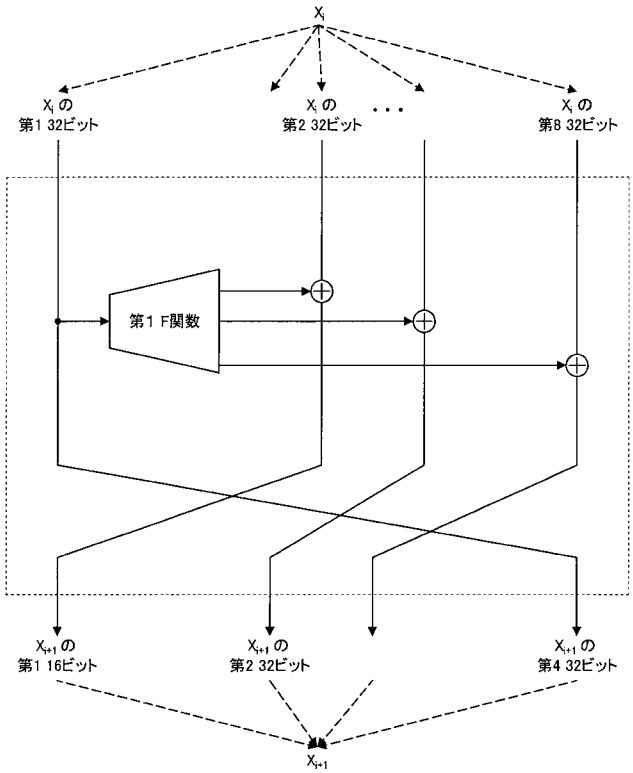
【図１３】



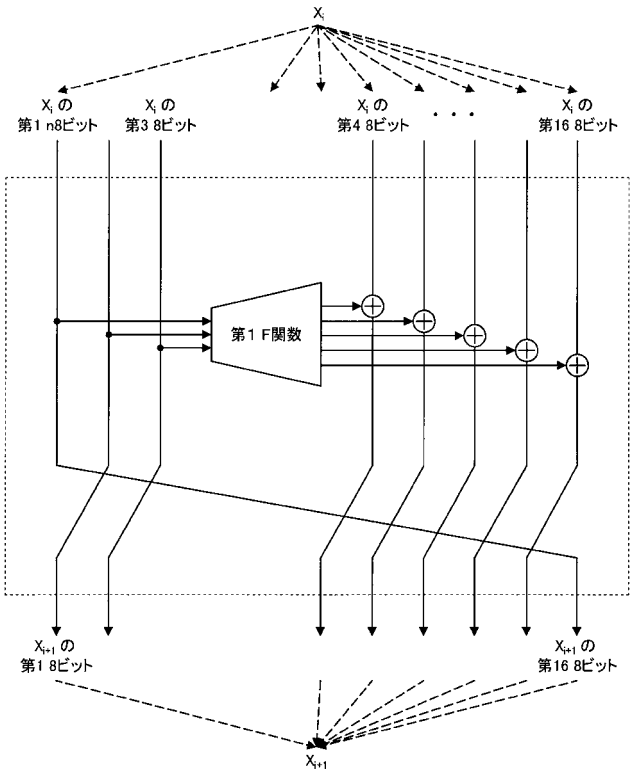
【図１４】



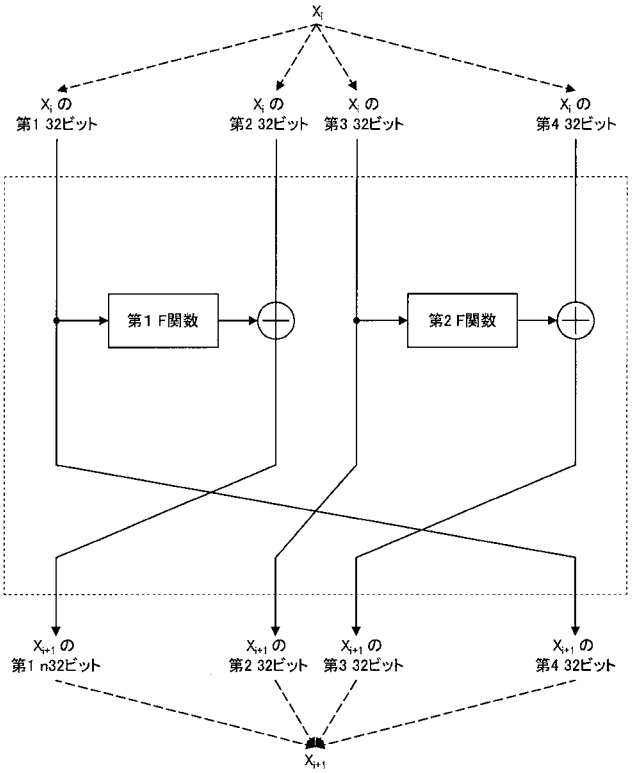
【図１５】



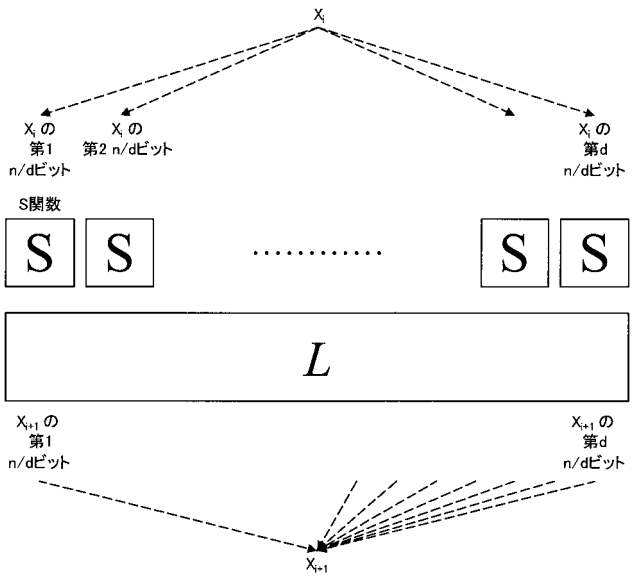
【図１６】



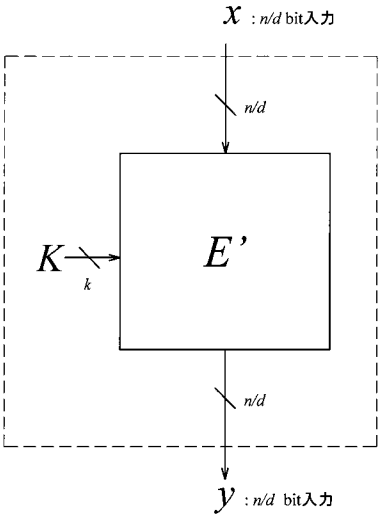
【図１７】



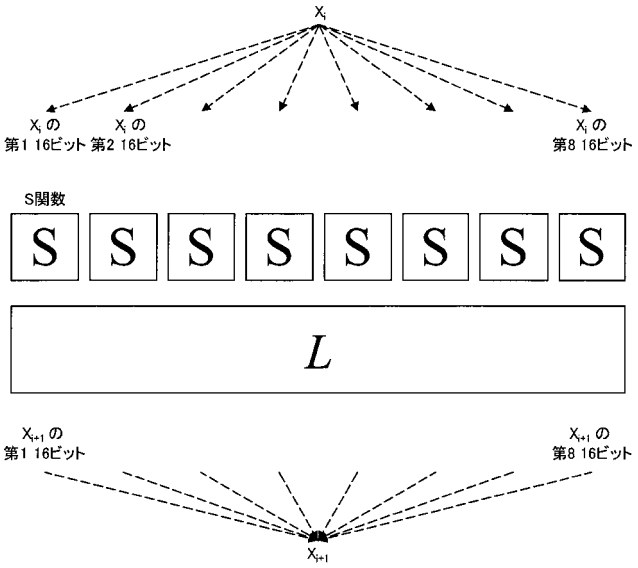
【図１８】



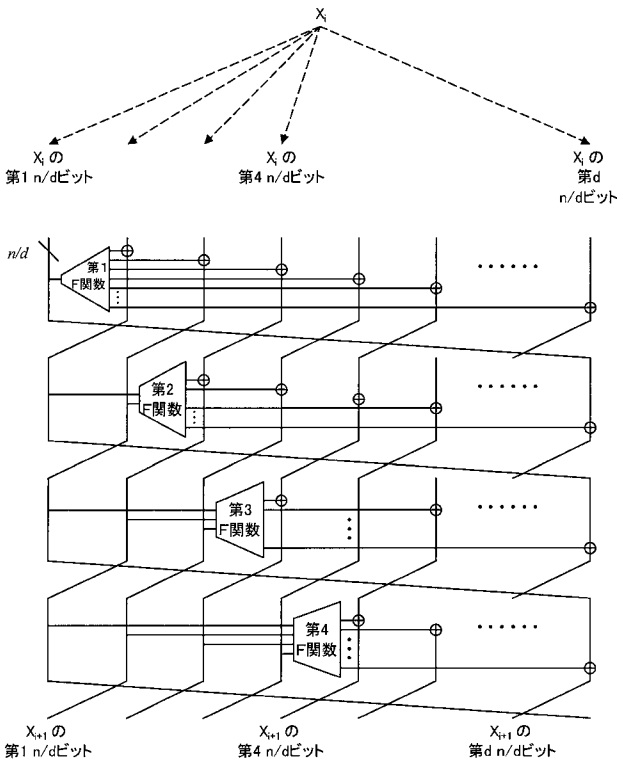
【図１９】



【図２０】

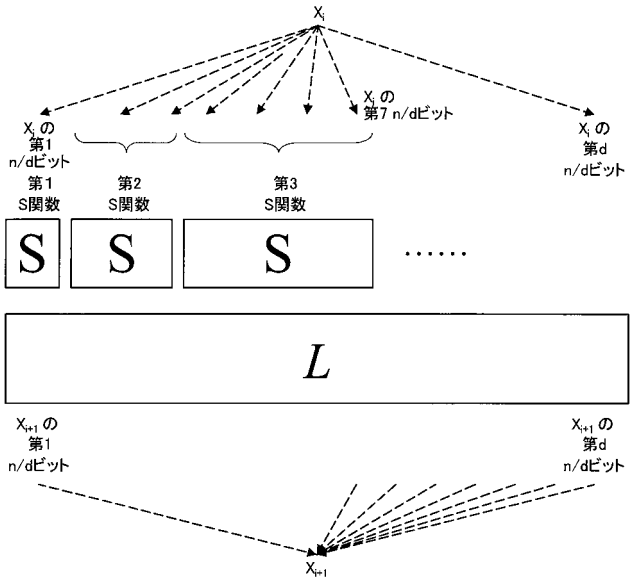


【図２１】



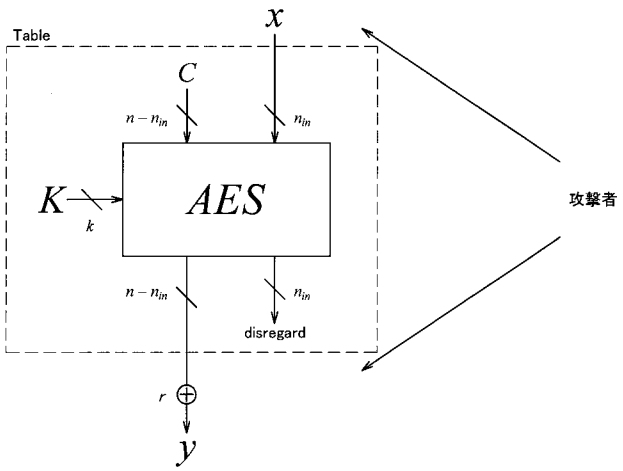
第1 F関数 (n/d bit 入力, n − n/d bit 出力)
第2 F関数 (2n/d bit 入力, n − 2n/d bit 出力)
第3 F関数 (3n/d bit 入力, n − 3n/d bit 出力)
第4 F関数 (4n/d bit入力, n − 4n/d bit 出力

【図２２】

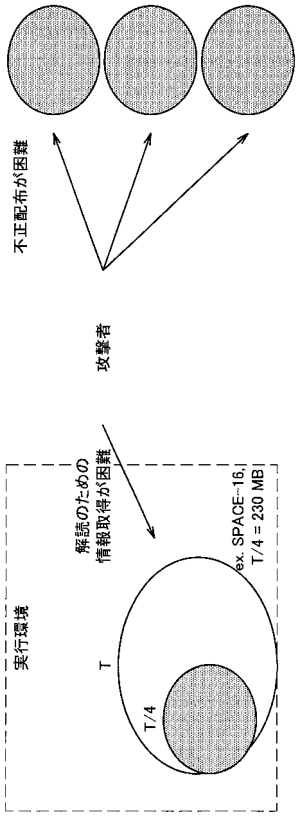


第1 S関数 (n/d bit to n − n/d bit)
第2 S関数 (2n/d bit to n − 2n/d bit)
第3 S関数 (4n/d bit to n − 4n/d bit)

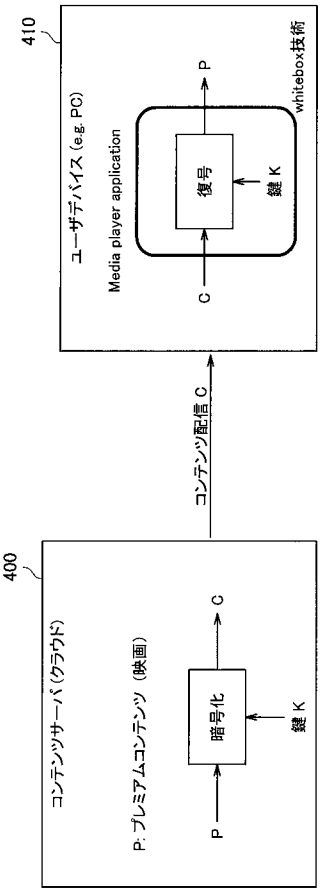
【図２３】



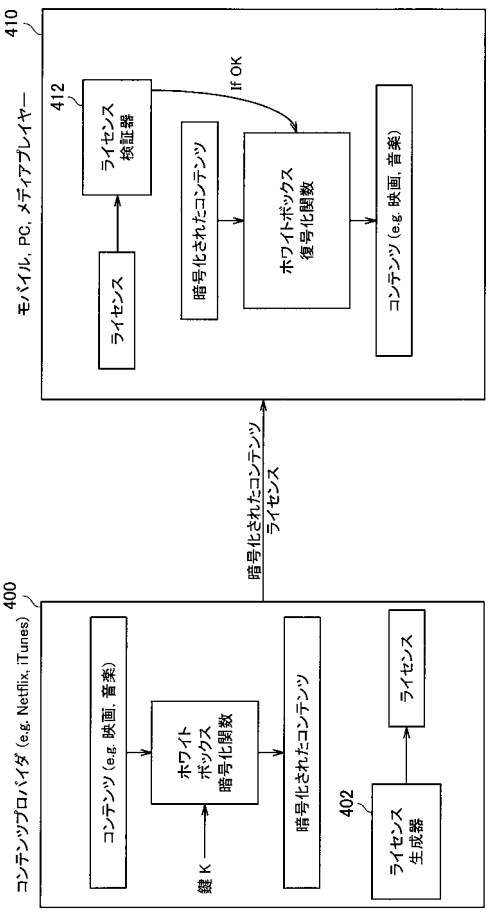
【図２４】



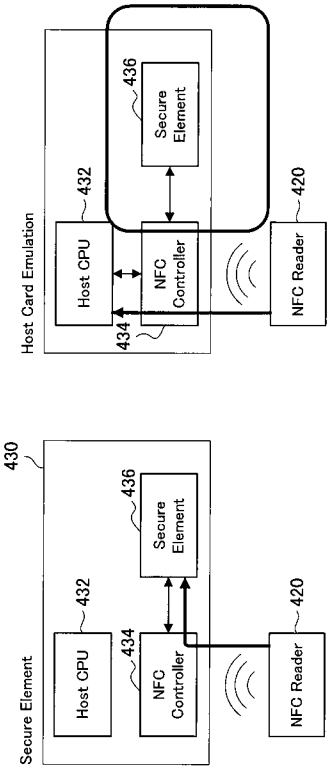
【図２５】



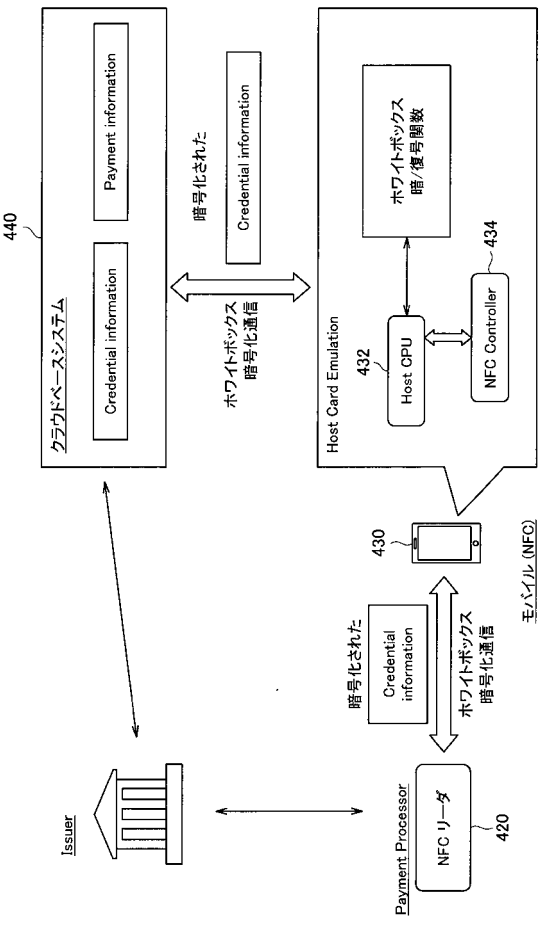
【図２６】



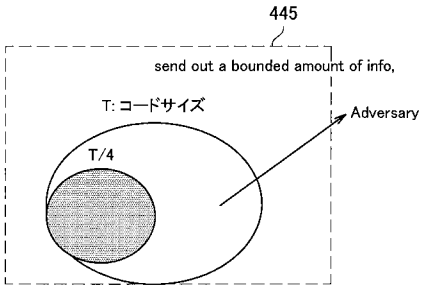
【図２７】



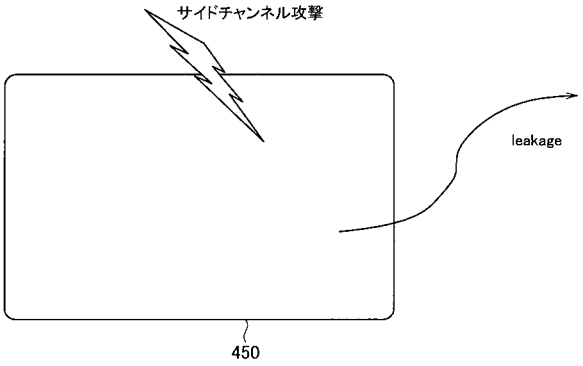
【図２８】



【図２９】


445
send out a bounded amount of info,
T: コードサイズ
Adversary
T/4

【図３０】


サイドチャンネル攻撃
leakage
450

フロントページの続き

(74)代理人　100128587
弁理士　松本　一騎
(72)発明者　五十部　孝典
東京都港区港南１丁目７番１号　ソニー株式会社内
(72)発明者　アンドレイ　ボグダノフ
デンマーク、ビルディング３２４，ルーム２２１、２８００コーゲンス、リュンビュー

Ｆターム(参考) 5J104 AA41 JA07 NA02 NA28