(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4626148号
(P4626148)
(45)発行日　平成23年2月2日(2011.2.2)　(24)登録日　平成22年11月19日(2010.11.19)

(51)Int.Cl.　F I
G09C 1/00 (2006.01)　G09C 1/00 620A

請求項の数 7 (全 19 頁)

(21)出願番号　特願2004-1602(P2004-1602)
(22)出願日　平成16年1月7日(2004.1.7)
(65)公開番号　特開2005-195829(P2005-195829A)
(43)公開日　平成17年7月21日(2005.7.21)
審査請求日　平成19年1月9日(2007.1.9)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　100100310
弁理士　井上　学
(72)発明者　桶屋　勝幸
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所システム開発研究所内
(72)発明者　高木　剛
ドイツ連邦共和国、６５７１９、ホッフハイムヴァーラウ市、リューデスハイム通り７番

審査官　青木　重徳

最終頁に続く

(54)【発明の名称】復号または署名作成におけるべき乗剰余算の計算方法

(57)【特許請求の範囲】
【請求項１】
公開鍵暗号において，秘密指数dと，公開鍵nと，暗号文cとから，平文mを復号する，公開鍵暗号における復号装置であって，
前記暗号文cをランダム化された暗号文tに変換する処理部と，
ランダム化された暗号文tをランダム化された平文uに変換する処理部と，
ランダム化された平文uを平文mに変換する処理部と，
べき剰余計算部と，を有し，
前記ランダム化された暗号文tに変換する処理部は，
乱数rを生成する処理部と，
前記乱数rと前記乱数rより導出された整数sを用いてランダム化された暗号文tに変換する処理部とを含み，
前記ランダム化された平文uに変換する処理部は，
前記秘密指数dより導出された値を用いてランダム化された平文uを計算する処理部を含み，
前記平文mに変換する処理部は，
ランダム化された平文uに前記整数sを乗ずる処理部を含む
ことを特徴とする公開鍵暗号における復号装置。
【請求項２】
請求項１記載の公開鍵暗号における復号装置であって，さらに，

モジュラー乗算部を備え，
前記乱数rより導出された前記整数ｓとして，前記べき剰余計算部により計算されるre-1を用い，
前記ランダム化された暗号文tに変換する処理部は，前記モジュラー乗算部を用いて，srを前記暗号文cに乗ずる処理部を含む
ことを特徴とする公開鍵暗号における復号装置。
【請求項３】
請求項１または２に記載の公開鍵暗号における復号装置であって，さらに，
前記ランダム化された平文uを計算する処理部は，
前記べき剰余計算部を用いて，前記秘密指数dより導出された値としてd-1を用い，前記ランダム化された暗号文tのd-1乗を計算する処理部を含む
ことを特徴とする公開鍵暗号における復号装置。
【請求項４】
請求項１記載の公開鍵暗号における復号装置であって，さらに，
モジュラー乗算部を備え，
前記暗号文tに変換する処理部は，
前記モジュラー乗算部を用いて，前記乱数rより導出された値としてr2を用い，r2を前記暗号文cに乗ずる処理部を含む
ことを特徴とする公開鍵暗号における復号装置。
【請求項５】
請求項１記載の公開鍵暗号における復号装置であって，さらに，
前記公開鍵nは，p mod 4 = q mod 4 = 3となる素因数p,qを含み，
前記ランダム化された平文uを計算する処理部は，
前記べき剰余計算部を用いて前記秘密指数dから(p-3)/4および(q-3)/4を導出する処理部と，
ランダム化された暗号文tの(p-3)/4乗を計算する処理部と，
ランダム化された暗号文tの(q-3)/4乗を計算する処理部と，を含む
ことを特徴とする公開鍵暗号における復号装置。
【請求項６】
請求項４に記載の公開鍵暗号における復号装置であって，さらに，
前記平文mに変換する処理部における前記整数sとして前記r2を用いる
ことを特徴とする公開鍵暗号における復号装置。
【請求項７】
公開鍵暗号を用いた電子署名において，秘密指数dと，公開鍵nと，データcとから，署名データmを生成する，公開鍵暗号を用いた署名作成装置であって，
前記データcをランダム化されたデータtに変換する処理部と，
ランダム化されたデータtをランダム化された署名uに変換する処理部と，
ランダム化された署名uを署名mに変換する処理部と，を有し，
前記ランダム化されたデータtに変換する処理部は，
乱数rを生成する処理部と，
前記乱数rと前記乱数rより導出された整数sを用いてランダム化されたデータtに変換する処理部とを含み，
前記ランダム化された署名uに変換する処理部は，
前記秘密指数dより導出された値を用いてランダム化された署名uを計算する処理部を含み，
前記署名mに変換する処理部は，
ランダム化された署名uに前記整数sを乗ずる処理部を含む
ことを特徴とする公開鍵暗号を用いた署名作成装置。
【発明の詳細な説明】
【技術分野】

【０００１】
本発明はセキュリティ技術に係り、特にべき乗剰余算を用いたデータ処理方法に関する。

【背景技術】
【０００２】
ＲＳＡ暗号はRivest, Shamir, Adleman により提案された公開鍵暗号である。公開鍵暗号には、公開鍵と呼ばれる一般に公開してよい情報と、秘密鍵と呼ばれる秘匿しなければならない秘密情報がある。与えられたデータの暗号化や署名の検証には公開鍵を用い、与えられたデータの復号化や署名の作成には秘密鍵を用いる。
【０００３】
ＲＳＡ暗号における秘密鍵は、大きな素数p,qおよび整数d、公開鍵は、整数n,eである。これらの数値の間には、

$$n=pq \quad (式1)$$

$$ed=1 \bmod \mathrm{Phi}(n) \quad (式2)$$

という関係が成り立っている。ただし、Phi(n)はオイラー関数であり、整数nと互いに素な正整数の個数を表す。n=pqの場合は、

$$\mathrm{Phi}(n) = (p-1)(q-1) \quad (式3)$$

となる。式1、式2より、任意の整数zに対して、

$$z^{ed} = z \bmod n \quad (式4)$$

という関係が成り立つ。この性質を利用することにより、暗号化・復号化等を達成することができる。すなわち、暗号化や署名の検証においては、

$$x^{e} \bmod n \quad (式5)$$

が計算され、復号化や署名の作成においては、

$$y^{d} \bmod n \quad (式6)$$

が計算される。ここでx,yは、入力データを表す整数である。この計算はべき乗剰余算と呼ばれる。
【０００４】
一般に、処理速度の向上のために、eの値は小さくとる。通常用いられる値は

$$e=65537 \quad (=2^{16}+1) \quad (式7)$$

である。
【０００５】
また、ＲＳＡ暗号の高速化手法として中国人剰余定理（以下、ＣＲＴという）が知られている。一方、ＲＳＡ暗号を性能向上させた様々な公開鍵暗号が提案されている。非特許文献４～７に記載されている、Multi-prime RSA, Multi-exponent RSA, Rabin cryptosystem, HIME(R)等がその例である。これらの暗号系へもＣＲＴは適用可能である。
【０００６】
ＲＳＡ暗号等を暗号装置として実装した場合、暗号処理に要する計算時間や電力消費量、電磁波等が観測可能である。それらの情報をもとに暗号装置内部に格納されている秘密鍵等の秘密情報を暴く方法が提案されている。この方法はサイドチャネル攻撃と呼ばれている。サイドチャネル攻撃は非特許文献１に記載されている。
【０００７】
ＣＲＴを用いたＲＳＡ暗号へのサイドチャネル攻撃が非特許文献２に記載されている。この攻撃は、Novakの攻撃と呼ばれている。この攻撃は、上記Multi-prime RSA, Multi-exponent RSA, Rabin cryptosystem, HIME(R)等の公開鍵暗号へも拡張可能である。
【０００８】
一方、サイドチャネル攻撃を防ぐ手法として、逆元演算を用いる方法が非特許文献３に記載されている。しかしながら、逆元演算は多くの計算量を必要とする計算である。そのため、この方法はサイドチャネル攻撃を防ぐことはできるが、多くの計算時間を必要とする。また、逆元演算をあらかじめ計算し、その値をメモリに格納しておくこともできるが、その場合は多くのメモリを使用する。

【０００９】
【非特許文献１】P.C. Kocher, J. Jaffe, and B. Jun, “Differential Power Analysis,” in the proceedings of CRYPTO 1999, Lecture Note in Computer Science 1666, Springer-Verlag, pp.388-397, (1999).
【００１０】
【非特許文献２】R. Novak, “SPA-Based Adaptive Chosen-Ciphertext Attack on RSA Implementation,” in the proceedings of 2002 International Workshop on Practice and Theory in Public Key Cryptography (PKC 2002), Lecture Note in Computer Science 2274, Springer-Verlag, pp.252-262, (2002).
【非特許文献３】P.C. Kocher, “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems,” in the proceedings of CRYPTO 1996, Lecture Note in Computer Science 1109, Springer-Verlag, pp.104-113, (1996).
【非特許文献４】Public-Key Cryptography Standards, PKCS # 1,Amendment 1: Multi-Prime RSA, RSA Laboratories, (2000)
【非特許文献５】T. Takagi, “Fast RSA-type cryptosystem modulo pkq”, in the proceedings of CRYPTO 1998, Lecture Note in Computer Science 1462, Springer-Verlag, pp.318-326, (1998).
【非特許文献６】D. Boneh, “Simplified OAEP for the RSA and RabinFunctions,” in the proceedings of CRYPTO 2001, Lecture Note in Computer Science 2139, pp.275-291, (2001).
【非特許文献７】M. Nishioka, H. Satoh, and K. Sakurai, ”Design and Analysis of Fast Provably Secure Public-Key Cryptosystems Based on a Modular Squaring,” in the proceedings of The 4th International Conference on Information Security and Cryptology (ICISC 2001), Lecture Note in Computer Science 2288, pp.81-102, (2001).
【発明の開示】
【発明が解決しようとする課題】
【００１１】
情報通信ネットワークの進展と共に電子情報に対する秘匿や認証のために暗号技術は不可欠な要素となってきている。暗号技術に課せられる要件としては、安全性以外にも高速性やメモリ使用量などがある。特に、スマートカード（ICカードともいう）等においては利用可能なリソースは少なく、限られたリソース内で暗号処理を最適化する必要がある。
【００１２】
上記技術は、サイドチャネル攻撃を防ぐ方法としては有効であるが、高速性やメモリ使用量に関しては考慮されていない。
【課題を解決するための手段】
【００１３】
本発明は、サイドチャネル攻撃を防ぐことのできる、かつ高速性やメモリ使用量に優れるべき乗剰余計算方法を提供する。
【００１４】
本発明は、公開鍵暗号において、秘密指数d，公開鍵n，暗号文cから、平文mを復号する復号方法であって、上記暗号文cをランダム化された暗号文tに変換するステップと、ランダム化された暗号文tをランダム化された平文uに変換するステップと、ランダム化された平文uを平文mに変換するステップと、を有するものであって、上記ランダム化された暗号文tに変換するステップは、乱数rを生成するステップと、上記乱数ｒと上記乱数rより導出された整数sを用いてランダム化された暗号文tに変換するステップとを含み、上記ランダム化された平文uに変換するステップは、上記秘密指数dより導出された値を用いてランダム化された平文uを計算するステップを含み、上記平文mに変換するステップは、ランダム化された平文uに上記整数sを乗ずるステップとを含むことを特徴とする。
【００１５】

本発明は、また、上記乱数rより導出された前記整数ｓとして、r^{e-1}を用い、上記暗号文tに変換するステップは、srを暗号文cに乗ずるステップを含む用に構成しても良い。

【００１６】

本発明は、また、上記ランダム化された平文uに変換するステップにおける上記秘密指数dより導出された値としてd-1を用い、上記ランダム化された平文uに変換するステップは、ランダム化された暗号文tのd-1乗を計算するステップを含むように構成してもよい。

【００１７】

本発明は、また、上記暗号文tに変換するステップにおける上記乱数rより導出された値として、r^4を用い、上記暗号文tに変換するステップは、r^4を暗号文cに乗ずるステップを含むように構成してもよい。

【００１８】

本発明は、また、上記公開鍵nは、
p mod 4 = q mod 4 = 3となる素因数p,qを含み、上記ランダム化された平文uに変換するステップにおける上記秘密指数dより導出された値として
(p-3)/4および(q-3)/4を用い、上記ランダム化された平文uに変換するステップは、ランダム化された暗号文tの
(p-3)/4乗を計算するステップと、ランダム化された暗号文tの
(q-3)/4乗を計算するステップと、を含むように構成してもよい。

【００１９】

本発明は、また、上記平文mに変換するステップにおける整数sとしてr^2を用いるように構成してもよい。

【００２０】

本発明は、また、公開鍵暗号を用いた電子署名において、秘密指数d，公開鍵n，データcから、署名データmを生成する署名作成方法であって、上記データcをランダム化されたデータtに変換するステップと、ランダム化されたデータtをランダム化された署名uに変換するステップと、ランダム化された署名uを署名mに変換するステップと、を有するように構成してもよい。

【００２１】

本発明は、また、上記ランダム化されたデータtに変換するステップは、乱数rを生成するステップと、上記乱数rと上記乱数rより導出された整数sを用いてランダム化されたデータtに変換するステップとを含み、上記ランダム化された署名uに変換するステップは、上記秘密指数dより導出された値を用いてランダム化された署名uを計算するステップを含み、上記署名mに変換するステップは、ランダム化された署名uに上記整数sを乗ずるステップとを含むように構成してもよい。

【００２２】

以上のように、本発明のべき乗剰余計算方法によれば、入力された暗号文はランダム化された暗号文に変換され、ランダム化された平文から平文への変換の際に逆元計算を行なう必要はないため、サイドチャネル攻撃を防ぐことのできる、かつ高速性やメモリ使用量に優れるべき乗剰余計算方法が利用可能になる。

【発明の効果】

【００２３】

本発明によれば、逆元演算を用いることなく、サイドチャネル攻撃に対して安全で、かつ高速性やメモリ使用量に優れる、公開鍵暗号技術における復号または署名作成処理が可能となる。

【発明を実施するための最良の形態】

【００２４】

以下、本発明の実施例を図面により説明する。

【００２５】

図１はネットワークによって接続された本発明によるべき乗剰余算の計算法を適用したコンピュータＡ101、コンピュータＢ121がネットワーク142により接続されたシステム構

成を示すものである。

【００２６】

図1の暗号通信システムにおけるコンピュータＡ101でデータの暗号化を行なうには、$c=m^e \bmod n$を計算して出力し、コンピュータＢ121で暗号文の復号化を行なうには、$c^d \bmod n$を計算して出力すればよい。ここでmは暗号化するデータを示す整数、e，nはそれぞれ公開鍵を示す整数、cはデータmに対する暗号文を示す整数、dは秘密鍵を示す整数である。

【００２７】

ネットワーク142には、暗号文cのみ送信され、データmを復元するためには、$c^d \bmod n$を計算する必要がある。ところが、秘密鍵dはネットワーク142には送信されないため、秘密鍵を保持しているものだけが、データmを復元できることになる。

【００２８】

図１において、コンピュータＡ101は、ＣＰＵ113やコプロセッサ114などの演算装置、ＲＡＭ103、ＲＯＭ106や外部記憶装置107などの記憶装置、コンピュータ外部とのデータ入出力を行なう入出力インタフェース110を装備しており、外部にはコンピュータＡをユーザが操作するためのディスプレイ108、キーボード109、着脱可能な可搬型記憶媒体の読み書き装置などが接続されている。

【００２９】

更にコンピュータＡ101は、ＲＡＭ103、ＲＯＭ106や外部記憶装置107などの記憶装置によって、記憶部102を実現し、ＣＰＵ113やコプロセッサ114などの演算装置が、記憶部102に格納されたプログラムを実行することにより、データ処理部112を実現する。

【００３０】

データ処理部112は、本実施形態においては、暗号化処理部112として入力されたデータの暗号化を行なう。

【００３１】

コンピュータＢ121はコンピュータＡ101と同様のハードウェア構成を備える。

【００３２】

更にコンピュータＢ121は、ＲＡＭ123、ＲＯＭ126や外部記憶装置127などの記憶装置によって、記憶部122を実現し、ＣＰＵ133やコプロセッサ134などの演算装置が、記憶部122に格納されたプログラムを実行することにより、データ処理部132を実現する。

【００３３】

データ処理部132は、本実施形態においては、復号化処理部132として機能し、暗号化されたデータである暗号文の復号化を行なう。

【００３４】

なお、上記各プログラムは、予め、上記コンピュータ内の記憶部に格納されていてもよいし、必要なときに、入出力インタフェースと上記コンピュータが利用可能な媒体を介して、他の装置から上記記憶部に導入されてもよい。媒体とは、例えば、入出力インタフェースに着脱可能な記憶媒体、または通信媒体（すなわちネットワークまたはネットワークを伝搬する搬送波）を指す。

【００３５】

次に、図1のコンピュータＡ101が、入力されたデータを暗号化する場合の動作について説明する。データはディジタルデータであればよく、テキスト、画像、音などの種類は問わない。

【００３６】

暗号化処理部112は、入出力インタフェース110を介して、平文データを受け取ると、入力された平文データのビット長が予め定めたビット長か否かを判断する。予め定めたビット長より長い場合には、予め定めたビット長となるように平文データを区切る。以下、所定のビット長に区切られている部分データ（単にデータともいう）について説明する。

【００３７】

次に、暗号化処理部112は、データのビット列によって表される数値mに対して、べき乗剰余$c=m^e \bmod n$を計算し、暗号化されたデータcを得る。コンピュータＡ101は暗号

化処理部112で暗号化された1つ以上の部分データから暗号化された出力データを組み立てる。コンピュータＡ101は暗号化されたデータ141を入出力インタフェース110より出力し、ネットワーク142を介してコンピュータＢ121へ転送する。

【００３８】

次に、コンピュータＢ121が、暗号化されたデータ141を復号化する場合の動作について説明する。

【００３９】

復号化処理部132は、入出力インタフェース130を介して、暗号化されたデータが入力されると、入力された暗号化されたデータのビット長があらかじめ定めたビット長か否かを判断する。あらかじめ定めたビット長よりも長い場合は、あらかじめ定めたビット長となるように暗号化されたデータを区切る。以下、所定のビット長に区切られている部分データ（単にデータともいう）について説明する。

【００４０】

次に、復号化処理部132は、データのビット列により表される数値cに対して、
べき乗剰余$m'=c^d \bmod n$を計算し、暗号化される前の部分データmに相当するm'を得る。コンピュータＢ121は、復号化処理部132で復号化された部分データから平文データを組み立て、入出力インタフェース130を介して、ディスプレイ128などから出力する。

【００４１】

次に、コンピュータＢ121が、復号化処理を行う場合の、データ処理部132の処理を詳細に説明する。

【実施例１】

【００４２】

第1実施例では、復号化処理部132として、図2で示されるデータ処理部132の機能ブロックを用いる。データ処理部132は、モジュラー乗算部201、乱数生成部202、汎用べき乗剰余計算部203、定数204からなる。

【００４３】

復号化処理部132が所定のビット長に区切られている暗号文cから、秘密鍵dを用いて、暗号化される前の部分データである平文mを計算する第1の計算方法を、図3を用いて説明する。

【００４４】

復号化処理部132は暗号文cを受け取る。乱数生成部202は、乱数rを生成する（301）。汎用べき乗剰余計算部203は、
$r^{e-1} \bmod n$を計算し、sに格納する（302）。モジュラー乗算部201は、
$src \bmod n$を計算し、tに格納する（303）。この計算は、モジュラー乗算部201がs，rから
$sr \bmod n$を計算し、その後
$(sr)\ c \bmod n$を計算することにより達成できる。汎用べき乗剰余計算部203は、
t^{d-1} modを計算し、uに格納する（304）。モジュラー乗算部201は、
$us \bmod n$を計算し、vに格納する（305）。モジュラー乗算部201は、
$vc \bmod n$を計算し、m'に格納する。復号化処理部132は、m'を暗号化される前の部分データである平文mとして出力する。平分mから平分データが組み立てられる。

【００４５】

次に中国人剰余定理を用いて汎用べき乗剰余算を高速化する場合の処理について、図4、図5を用いて説明する。この場合、汎用べき乗剰余計算部203はＣＲＴ計算部203として働く。

【００４６】

ＣＲＴ計算部203は、モジュラー乗算部401、モジュラー加減算部402、モジュラー剰余乗算部403、べき乗剰余計算部404、加減乗算部405、定数406からなる。ＣＲＴ計算部203は、整数tと整数dから、整数p,qを用いて、
$t^d \bmod n$を計算する。ここでp,qはnを割りきる素数である。

【００４７】

ＣＲＴ計算部203は、整数tと整数dを受け取る。モジュラー剰余計算部403は、
$d \bmod p-1$, $d \bmod q-1$ を計算し、d_p, d_qにそれぞれ格納する（501）。モジュラー剰余計算部403は
$t \bmod p$, $t \bmod q$ を計算し、t_p, t_qにそれぞれ格納する（502）。べき乗剰余計算部404は、$t_p^{\ dp} \bmod p$, $t_q^{\ dq} \bmod q$を計算し、それぞれm_p, m_qに格納する（503）。モジュラー乗算部401およびモジュラー加減算部402は、
$(m_q-m_p)\mathrm{pInv} \bmod q$を計算し、hに格納する（504）。この計算は、モジュラー加減算部402が
$m_q-m_p \bmod q$を計算し、モジュラー乗算部401が、その結果にpInvを乗ずる（すなわち $(m_q-m_p)\mathrm{pInv} \bmod q$）ことにより達成できる。加減乗算部405は、$m_p+ph$を計算し、$m_{pq}$に格納する（505）。この計算は、加減乗算部405がphを計算し、その結果にm_pを加える（すなわち $m_p+(ph)$）ことにより達成できる。ただし、pInvは、

$$\mathrm{pInv} = p^{-1} \bmod q \qquad \text{（式8）}$$

をみたす整数である。pInvは定数406としてＣＲＴ計算部203に格納されている。ＣＲＴ計算部203は、m_{pq}を
$t^d \bmod n$として出力する（506）。

【００４８】

復号化処理部132の行なう処理により出力されるm'はm'=mとなる。この理由は次の通りである。

$$m' = vc = usc = t^{d-1}sc$$
$$= (src)^{d-1}sc = (r^e c)^{d-1} r^{e-1} c$$
$$= c^d r^{ed-e+e-1} = c^d \bmod n \qquad \text{（式9）}$$

ここで、e,d の定義より、$r^{ed-1} = 1$が成り立つ。cはmを暗号化したものであるので、

$$c^d = m \bmod n \qquad \text{（式10）}$$

が成り立つ。したがって、m'=mとなる。

【００４９】

上記の計算方法では、剰余乗算等の演算は用いるが、逆元演算は用いない。したがって、非特許文献３の手法と比べて、高速な計算が可能である。また、eを式7により定めた場合、直接$c^d \bmod n$を計算する方法と比べると、計算量増加は剰余乗算20回分である。一般に、1024ビットの汎用べき乗剰余算を行なった場合、1500回程度の剰余乗算を必要とするため、上記計算方法の計算量増加は、この値と比べて比較的小さいと言える。

【００５０】

また、上記の計算方法では逆元演算を用いないため、逆元演算をあらかじめ計算し、その値をメモリに格納する、ということを行なう必要がない。そのため、メモリ使用量が少なくてすむ。

【００５１】

また、上記の計算方法は、サイドチャネル攻撃に対する防御に関しても有効である。この理由は次の通りである。上記の計算方法で秘密鍵を用いるのはステップ503のみである。ステップ503の入力である整数tは、入力されたデータcと、ステップ501で生成された乱数rのe乗との乗算結果である。したがって、攻撃者は整数tの値を予測することはできない。Novakの攻撃は、汎用べき乗剰余算に入力される値が攻撃者にとって既知であることを利用しているので、攻撃者はNovakの攻撃を行なうことができない。

【００５２】

以上の通り、上記の計算方法は、サイドチャネル攻撃に有用な情報を与えないため、サイドチャネル攻撃に対して耐性がある。また、逆元演算を用いないため、高速な計算が可能であり、さらに、逆元演算の結果をあらかじめメモリに格納する必要もないため、メモリ使用量も少なくてすむ。

【００５３】

なお、上記の計算方法は、multi-prime RSA や multi-exponent RSA に対しても適用できる。この場合においても、サイドチャネル攻撃に耐性を有し、さらに、逆元演算を用い

ないため、高速計算可能かつ小メモリとなる。

【００５４】

まず、multi-prime RSA の場合について説明する。

【００５５】

非特許文献４に開示されているように、multi-prime RSAにおける秘密鍵は、大きな素数p,q,rおよび整数d、公開鍵は、整数n,eである。これらの数値の間には、

$n = pqr$ （式11）

$ed = \mathrm{mod}\ Phi(n)$ （式12）

という関係が成り立っている。ここでは、大きな素数を３つとしたが、４つやそれ以上でもよい。式12のnの場合、

$Phi(n) = (p-1)(q-1)(r-1)$ （式13）

となる。multi-prime RSAにおいても、式4が成立し、そのため、暗号化や署名の検証においては式5が計算され、復号化や署名の作成においては式6が計算される。従って復号化処理を行なう場合、第1の計算方法を用いることにより、サイドチャネル攻撃に耐性を有し、さらに、逆元演算を用いないため、高速計算可能かつ小メモリとなる。

【００５６】

次に中国人剰余定理を用いて汎用べき乗剰余算を高速化する場合の処理について、図4、図6を用いて説明する。この場合、汎用べき乗剰余計算部203はＣＲＴ計算部203として働く。

【００５７】

ＣＲＴ計算部203は、整数tと整数dから、整数p,q,rを用いて、$t^d \bmod n$ を計算する。ここで、p,q,rはnを割りきる素数である。

【００５８】

ＣＲＴ計算部203は、整数tと整数dを受け取る。モジュラー剰余計算部403は、
$d \bmod p-1$, $d \bmod q-1$, $d \bmod r-1$ を計算し、d_p, d_q, d_r にそれぞれ格納する（601）。
モジュラー剰余計算部403は
$t \bmod p$, $t \bmod q$, $t \bmod r$ を計算し、t_p, t_q, t_rにそれぞれ格納する（602）。べき乗剰余計算部404は、
$t_p^{\ dp} \bmod p$, $t_q^{\ dq} \bmod q$, $t_r^{\ dr} \bmod r$ を計算し、それぞれ m_p, m_q, m_r に格納する（603）。モジュラー乗算部401およびモジュラー加減算部402は、
$(m_q-m_p)pInv \bmod q$ を計算し、hに格納する（604）。この計算は、モジュラー加減算部402が
$m_q-m_p \bmod q$ を計算し、モジュラー乗算部401が、その結果にpInvを乗ずる
（すなわち $(m_q-m_p)pInv \bmod q$）ことにより達成できる。ただし、pInvは、式*をみたす整数である。加減乗算部405は、m_p+phを計算し、m_{pq}に格納する（605）。この計算は、加減乗算部405がphを計算し、その結果にm_pを加える（すなわち $m_p+(ph)$）ことにより達成できる。モジュラー乗算部401およびモジュラー加減算部402は、
$(m_r-m_{pq})pqInv \bmod r$ を計算し、hに格納する（606）。この計算は、モジュラー加減算部402が
$m_r-m_{pq} \bmod r$ を計算し、モジュラー乗算部401が、その結果にpqInvを乗ずる
（すなわち $(m_r-m_{pq})pqInv \bmod r$）ことにより達成できる。ただし、pqInvは、

$pqInv = (pq)^{-1} \bmod r$ （式14）

をみたす整数である。pqInvは定数406として、ＣＲＴ計算部203に格納されている。加減乗算部405は、$m_{pq}+pqh$を計算し、m_{pqr} に格納する（607）。この計算は、加減乗算部405がpqを計算し、その結果にhを乗じ、さらにその結果にm_{pq}を加える（すなわち $m_{pq}+((pq)h)$）ことにより達成できる。ＣＲＴ計算部203は、m_{pqr} を
$t^d \bmod n$ として出力する（608）。

【００５９】

次に、multi-exponent RSA の場合について説明する。

【００６０】

非特許文献５に開示されているように、multi-exponent RSAにおける秘密鍵は、大きな素数p,qおよび整数d、公開鍵は、整数n,eである。これらの数値の間には、

$n = p^2q$ （式15）

$ed = \mathrm{mod}\ Phi(n)$ （式16）

という関係が成り立っている。ここでは、べき指数を2としたが、3やそれ以上でもよい。式16のnの場合、

$Phi(n) = p(p-1)(q-1)$ （式17）

となる。multi-exponent RSAにおいても、式4が成立し、そのため、暗号化や署名の検証においては式5が計算され、復号化や署名の作成においては式6が計算される。従って復号化処理を行なう場合、第1の計算方法を用いることにより、サイドチャネル攻撃に耐性を有し、さらに、逆元演算を用いないため、高速計算可能かつ小メモリとなる。

【００６１】

次に中国人剰余定理を用いて汎用べき乗剰余算を高速化する場合の処理について、図4、図7を用いて説明する。この場合、汎用べき乗剰余計算部203はＣＲＴ計算部203として働く。

【００６２】

ＣＲＴ計算部203は、整数tと整数dから、整数p,qを用いて、$t^d \bmod n$ を計算する。ここで、p,qはnを割りきる素数であり、さらにnはp^2でも割りきれる。

【００６３】

ＣＲＴ計算部203は、整数tと整数dを受け取る。モジュラー剰余計算部403は、$d \bmod p-1$, $d \bmod q-1$ を計算し、d_p, d_q にそれぞれ格納する（701）。モジュラー剰余計算部403は

$t \bmod p$, $t \bmod q$ を計算し、t_p, t_q にそれぞれ格納する（702）。べき乗剰余計算部404は、

$t_p^{\,dp-1} \bmod p$を計算し、kに格納する（703）。モジュラー剰余計算部403は、

$t_p\ k \bmod p$ を計算し、m_p に格納する。べき乗剰余計算部404は、

$t_q^{\,dq} \bmod q$ を計算し、m_q に格納する（704）。べき乗剰余計算部404、モジュラー加減算部402、およびモジュラー剰余計算部403は、

$c - m_p^{\,e} \bmod p^2$ を計算し、gに格納する。モジュラー乗算部401は、

$g\ k\ eInv \bmod p^2$ を計算し、bに格納する。加減乗算部405は、$m_q + b$ を計算し、m_{p2} に格納する（705）。これらの計算は、べき乗剰余計算部404が

$m_p^{\,e} \bmod p^2$ を計算し、モジュラー剰余計算部403が

$c \bmod p^2$ を計算し、モジュラー加減算部402がそれらの差分をとる（すなわち

$(c \bmod p^2)\ (m_p^{\,e}) \bmod p^2$）ことによりgが計算できる。モジュラー乗算部401が$g\ k \bmod p^2$ を計算し、その結果にeInv を乗ずる

（すなわち $(gk)eInv \bmod p^2$）ことによりbが計算できる。その後、加減乗算部405が $m_q + b$ を計算することにより、達成できる。ただし、eInvは、

$eInv = (e)^{-1} \bmod p$ （式18）

をみたす整数である。eInvは定数406としてＣＲＴ計算部203に格納されている。モジュラー乗算部401およびモジュラー加減算部402は、

$(m_q - m_{p2})p^2Inv \bmod q$ を計算し、hに格納する（706）。この計算は、モジュラー加減算部402が

$m_q - m_{p2} \bmod q$ を計算し、モジュラー乗算部401が、その結果にp^2Invを乗ずる

（すなわち $(m_q - m_{p2})p^2Inv \bmod q$）ことにより達成できる。ただし、$p^2$Invは、

$p^2Inv = (p^2)^{-1} \bmod r$ （式19）

をみたす整数である。p^2Invは定数406としてＣＲＴ計算部203に格納されている。加減乗算部405は、$m_{p2} + p^2h$を計算し、m_{p2q}に格納する（707）。この計算は、加減乗算部405が$(p^2)h$を計算し、その結果にm_{p2}を加える（すなわち $m_p + ((p^2)h)$）ことにより達成できる。ＣＲＴ計算部203は、m_{p2q} を

$t^d \bmod n$ として出力する（708）。

【実施例２】
【００６４】
第2実施例では、Rabin暗号の場合について説明する。
【００６５】
非特許文献６に開示されているように、Rabin暗号における秘密鍵は、大きな素数p,q、公開鍵は、整数n,eである。Rabin暗号の場合は特に

$$e=2 \qquad (式20)$$

である。また素数p,qは、

$$p \bmod 4 = q \bmod 4 = 3 \qquad (式21)$$

をみたすように選ばれる。これらの数値の間には、

$$n = pq \qquad (式22)$$

という関係が成り立っている。Rabin暗号の場合、暗号化や署名の検証においては、xの平方

$$y = x^2 \bmod n \qquad (式23)$$

を計算し、復号化や署名の作成においては、式23をみたすxを計算する。そのため、Rabin暗号では、式23をみたす整数が4つ存在し、そのうち適切なものを選択することとなる。
【００６６】
第2実施例では、復号化処理部132として、図8で示されるデータ処理部132の機能ブロックを用いる。データ処理部132は、モジュラー乗算部801、乱数生成部802、汎用べき乗剰余計算部803、定数804、モジュラー加減算部805、加減乗算部806からなる。
【００６７】
復号化処理部132が所定のビット長に区切られている暗号文cから、秘密鍵を用いて、平文mを計算する第2の計算方法を、図9を用いて説明する。
【００６８】
復号化処理部132は暗号文cを受け取る。乱数生成部802は、乱数rを生成する（901）。モジュラー乗算部801は、
$r^2 \bmod n$を計算し、sに格納する（902）。モジュラー乗算部801は、
$s^2c \bmod n$を計算し、tに格納する（903）。この計算は、モジュラー乗算部801がsから
$s^2 \bmod n$を計算し、その後
$(s^2)\ c \bmod n$を計算することにより達成できる。汎用べき乗剰余計算部803は、
$t^{(p-3)/4} \bmod p$, $t^{(q-3)/4} \bmod q$ を計算し、それぞれu_p, u_q格納する（904）。モジュラー乗算部801およびモジュラー加減算部805は、
$(u_q-u_p)pInv \bmod q$ を計算し、h_1に格納する。加減乗算部806は、
u_p+ph_1を計算し、w_1に格納する（905）。これらの計算は、モジュラー加減算部805が
$u_q-u_p \bmod q$ を計算し、モジュラー乗算部801が、その結果にpInvを乗ずる
（すなわち $(u_q-u_p)pInv \bmod q$）ことによりh_1が計算できる。その後、加減乗算部806がph_1を計算し、その結果にu_pを加える
（すなわち $u_p+(ph_1)$）ことにより、達成できる。ただし、pInvは、式8をみたす整数である。モジュラー乗算部801およびモジュラー加減算部805は、
$(u_q+u_p)pInv \bmod q$ を計算し、h_2に格納する。加減乗算部806は、
$-u_p+ph_2$を計算し、w_2に格納する（906）。これらの計算は、モジュラー加減算部805が
$u_q+u_p \bmod q$ を計算し、モジュラー乗算部801が、その結果にpInvを乗ずる
（すなわち $(u_q+u_p)pInv \bmod q$）ことによりh_2が計算できる。その後、加減乗算部806がph_2を計算し、その結果からu_pを減ずる（すなわち $-u_p+(ph_2)$）ことにより、達成できる。加減乗算部806は、
$n-w_1$, $m-w_2$ を計算し、それぞれw_3, w_4 に格納する（907）。モジュラー乗算部801は、
$w_1\ cs \bmod n$, $w_2\ cs \bmod n$, $w_3\ cs \bmod n$, $w_4\ cs \bmod n$を計算し、それぞれm_1, m_2, m_3, m_4 に格納する（908）。これらの計算は、モジュラー乗算部801が $cs \bmod n$ を計算し、その結果にそれぞれw_1, w_2, w_3, w_4 を乗ずることにより達成できる。復号化処理部132は、m_1, m_2, m_3, m_4 を平文mの候補として出力する（909）。

【００６９】
復号化処理部132の行なう処理により出力されるm_1，m_2，m_3，m_4は
$$m_1^2 = m_2^2 = m_3^2 = m_4^2 = c \bmod n \quad (式24)$$
をみたす。この理由は次の通りである。まず、u_pに関して次の式が成立する。
【００７０】
$$u_p^2 = (t^{(p-3)/4})^2$$
$$= (s^2 c)^{(p-3)/2} = s^{p-3} m^{p-3} \bmod p \quad (式25)$$
ここで、pは素数であるので、任意の整数zに対して、
$$z^p = z \bmod p \quad (式26)$$
が成り立つことに注意すると、
$$u_p^2 = s^{-2} m^{-2} \bmod p \quad (式27)$$
が成り立つ。同様に、
$$u_q^2 = s^{-2} m^{-2} \bmod q \quad (式28)$$
も成り立つ。他方、w_1に関して次の式が成立する。
【００７１】
$$w_1^2 = u_p^2 \bmod p \quad (式29)$$
$$w_1^2 = (u_p + p(u_q - u_p)pInv)^2 \bmod q \quad (式30)$$
pInvは式*をみたすので、
$$w_1^2 = u_q^2 \bmod q \quad (式31)$$
が成り立つ。式22、式27、式28、式29、式31を考慮すると、
$$m_1^2 = w_1^2 c^2 s^2 \bmod n$$
$$= c \bmod n \quad (式32)$$
が成り立つ。m_2，m_3，m_4に対しても、同様にして示すことができる。
【００７２】
上記の計算方法では、剰余乗算等の演算は用いるが、逆元演算は用いない。したがって、非特許文献３の手法と比べて、高速な計算が可能である。また、e=2であるため、通常のRabin暗号の計算手法と比べると、計算量増加は剰余乗算5回分である。一般に、1024ビットの汎用べき乗剰余算を行なった場合、1500回程度の剰余乗算を必要とするため、上記計算方法の計算量増加は、この値と比べて比較的小さいと言える。
【００７３】
また、上記の計算方法では逆元演算を用いないため、逆元演算をあらかじめ計算し、その値をメモリに格納する、ということを行なう必要がない。そのため、メモリ使用量が少なくてすむ。
【００７４】
また、上記の計算方法は、サイドチャネル攻撃に対する防御に関しても有効である。この理由は次の通りである。上記の計算方法で秘密鍵が初めて用いられるのはステップ904であり、その後の演算は、ステップ904の演算結果を用いる。ステップ904の入力である整数tは、入力されたデータcと、ステップ901で生成された乱数rの4乗との乗算結果である。したがって、攻撃者は整数tの値を予測することはできない。Novakの攻撃は、汎用べき乗剰余算に入力される値が攻撃者にとって既知であることを利用しているので、攻撃者はNovakの攻撃を行なうことができない。
【００７５】
以上の通り、上記の計算方法は、サイドチャネル攻撃に有用な情報を与えないため、サイドチャネル攻撃に対して耐性がある。また、逆元演算を用いないため、高速な計算が可能であり、さらに、逆元演算の結果をあらかじめメモリに格納する必要もないため、メモリ使用量も少なくてすむ。
【００７６】
なお、ステップ906で $-u_p$ を $p-u_p$ におきかえて実行してもよい。そうすると、負の整数を扱うことなく実行できるため、プログラムを簡略化できる。
【実施例３】

【００７７】
第3実施例では、HIME(R)の場合について説明する。 非特許文献７に開示されているように、HIME(R)における秘密鍵は、大きな素数p,q、公開鍵は、整数n,eである。HIME(R)の場合は、Rabin暗号と同様に、eは式20をみたすように選ばれる。また素数p,qについても、式21をみたすように選ばれる。これらの数値の間には、

$$n = p^2 q \qquad (式33)$$

という関係が成り立っている。HIME(R)の場合、暗号化や署名の検証においては、xの平方

$$y = x^2 \bmod n \qquad (式34)$$

を計算し、復号化や署名の作成においては、式34をみたすxを計算する。そのため、HIME(R)では、式34をみたす整数が4つ存在し、そのうち適切なものを選択することとなる。

【００７８】
第3実施例では、復号化処理部132として、図8で示されるデータ処理部132の機能ブロックを用いる。

【００７９】
復号化処理部132が所定のビット長に区切られている暗号文cから、秘密鍵を用いて、平文mを計算する第3の計算方法を、図10を用いて説明する。

【００８０】
復号化処理部132は暗号文cを受け取る。乱数生成部802は、乱数rを生成する（1001）。モジュラー乗算部801は、
$r^2 \bmod n$を計算し、sに格納する（1002）。モジュラー乗算部801は、
$s^2 c \bmod n$を計算し、tに格納する（1003）。この計算は、モジュラー乗算部801がsから
$s^2 \bmod n$を計算し、その後
$(s^2)\ c \bmod n$を計算することにより達成できる。汎用べき乗剰余計算部803は、
$t^{(p-3)/4} \bmod p$, $t^{(q-3)/4} \bmod q$ を計算し、それぞれu_p, u_q格納する（1004）。モジュラー乗算部801は、
$u_p\ t \bmod p$ を計算し、kに格納する（1005）。モジュラー乗算部801およびモジュラー加減算部805は、
$(u_q - u_p)\mathrm{pInv} \bmod q$ を計算し、h_1に格納する。加減乗算部806は、
$u_p + ph_1$を計算し、$u_{pq,1}$に格納する（1006）。これらの計算は、モジュラー加減算部805が
$u_q - u_p \bmod q$ を計算し、モジュラー乗算部801が、その結果にpInvを乗ずる
（すなわち $(u_q - u_p)\mathrm{pInv} \bmod q$）ことにより$h_1$が計算できる。その後、加減乗算部806が$ph_1$を計算し、その結果に$u_p$を加える
（すなわち $u_p + (ph_1)$）ことにより、達成できる。ただし、pInvは、式8をみたす整数である。モジュラー乗算部801およびモジュラー加減算部805は、
$t\ u_{pq,1}^{\ 2} \bmod n$ を計算し、g_1に格納する。モジュラー乗算部801は、
$g_1\ k\ (2\mathrm{Inv}) \bmod n$ を計算し、b_1に格納する。モジュラー加減乗算部805は、
$u_{pq,1} + b_1$を計算し、u_1に格納する（1007）。これらの計算は、モジュラー乗算部801が
$u_{pq,1}^{\ 2} \bmod n$ を計算し、モジュラー加減算部805が、tからその結果を減ずる
（すなわち $t\ (u_{pq,1}^{\ 2}) \bmod n$）ことにより、$g_1$が計算できる。その後、モジュラー乗算部801が、
$k\ (2\mathrm{Inv}) \bmod n$ を計算し、その結果にg_1を乗ずる
（すなわち $g_1\ (k\ (2\mathrm{Inv})) \bmod n$）ことにより、$b_1$が計算できる。その後、加減乗算部806が、
$u_{pq,1} + b_1$ を計算することにより、達成できる。ただし、2Invは、

$$2\mathrm{Inv} = 2^{-1} \bmod p \qquad (式35)$$

をみたす整数である。モジュラー乗算部801およびモジュラー加減算部805は、
$(u_q + u_p)\mathrm{pInv} \bmod q$ を計算し、h_2に格納する。加減乗算部806は、
$p - u_p + ph_2$を計算し、$u_{pq,2}$に格納する（1008）。これらの計算は、モジュラー加減算部805が
$u_q + u_p \bmod q$ を計算し、モジュラー乗算部801が、その結果にpInvを乗ずる

（すなわち $(u_q+u_p)\mathrm{pInv} \bmod q$）ことにより$h_2$が計算できる。その後、加減乗算部806が$p$ h_2を計算し、その結果にpを加え、u_pを減ずる
（すなわち $p-u_p+(ph_2)$）ことにより、達成できる。

【００８１】

モジュラー乗算部801およびモジュラー加減算部805は、
$t\ u_{pq,2}^{\ 2} \bmod n$ を計算し、g_2に格納する。モジュラー乗算部801は、
$g_2\ k\ (2\mathrm{Inv}) \bmod n$ を計算し、b_2に格納する。モジュラー加減乗算部805は、
$u_{pq,2}+b_2$を計算し、u_2に格納する（1009）。これらの計算は、モジュラー乗算部801が
$u_{pq,2}^{\ 2} \bmod n$ を計算し、モジュラー加減算部805が、tからその結果を減ずる
（すなわち $t\ (u_{pq,2}^{\ 2}) \bmod n$）ことにより、$g_2$が計算できる。その後、モジュラー乗算部801が、
$k\ (2\mathrm{Inv}) \bmod n$ を計算し、その結果にg_2を乗ずる
（すなわち $g_2\ (k\ (2\mathrm{Inv})) \bmod n$）ことにより、$b_2$が計算できる。その後、加減乗算部806が、
$u_{pq,2}+b_2$ を計算することにより、達成できる。ただし、2Invは、式35をみたす整数である。加減乗算部806は、
$n-u_1$, $m-u_2$ を計算し、それぞれu_3, u_4 に格納する（1010）。モジュラー乗算部801は、
$u_1\ cs \bmod n$, $u_2\ cs \bmod n$, $u_3\ cs \bmod n$, $u_4\ cs \bmod n$を計算し、それぞれm_1, m_2, m_3, m_4 に格納する（1011）。これらの計算は、モジュラー乗算部801が $cs \bmod n$ を計算し、その結果にそれぞれu_1, u_2, u_3, u_4 を乗ずることにより達成できる。復号化処理部132は、m_1, m_2, m_3, m_4 を平文mの候補として出力する（1012）。

【００８２】

復号化処理部132の行なう処理により出力されるm_1, m_2, m_3, m_4は

$$m_1^{\ 2} = m_2^{\ 2} = m_3^{\ 2} = m_4^{\ 2} = c \bmod n \quad （式36）$$

をみたす。この理由は次の通りである。まず、u_pに関して次の式が成立する。

【００８３】

$$u_p^{\ 2} = (t^{(p-3)/4})^2$$
$$= (s^2 c)^{(p-3)/2} = s^{p-3} m^{p-3} \bmod p \quad （式37）$$

ここで、pは素数であるので、任意の整数zに対して、

$$z^p = z \bmod p \quad （式38）$$

が成り立つことに注意すると、

$$u_p^{\ 2} = s^{-2} m^{-2} \bmod p \quad （式39）$$

が成り立つ。同様に、

$$u_q^{\ 2} = s^{-2} m^{-2} \bmod q \quad （式40）$$

も成り立つ。式39、式40を用いると、

$$u_{pq,1}^{\ 2} = s^{-2} m^{-2} \bmod p \quad （式41）$$
$$u_{pq,1}^{\ 2} = s^{-2} m^{-2} \bmod q \quad （式42）$$

を示すことができ、したがって、

$$u_{pq,1}^{\ 2} = s^{-2} m^{-2} \bmod pq \quad （式43）$$

が成り立つ。$u = (sm)^{-1} \bmod n$, $u = u_{pq,1} + b$ と表すと、

$$t = u^2 = u_{pq,1}^{\ 2} + 2u_{pq,1}\ b \bmod n \quad （式44）$$

となる。そのため、

$$b = (t\ \ u_{pq,1}^{\ 2})((2\ u_{pq,1})^{-1} \bmod p) \bmod n \quad （式45）$$

となる。

【００８４】

$$k = u_p^{\ 1} \bmod p \quad （式46）$$

であることに注意すれば、$b = b_1$ が成り立つ。したがって、

$$m_1 = (sm)^{-1}\ cs = m \bmod n \quad （式47）$$

となる。m_2, m_3, m_4 についても同様に示すことができる。

【００８５】

上記の計算方法では、剰余乗算等の演算は用いるが、逆元演算は用いない。したがって、非特許文献３の手法と比べて、高速な計算が可能である。また、e=2であるため、通常のHIME(R)の計算手法と比べると、計算量増加は剰余乗算5回分である。一般に、1024ビットの汎用べき乗剰余算を行なった場合、1500回程度の剰余乗算を必要とするため、上記計算方法の計算量増加は、この値と比べて比較的小さいと言える。

【００８６】

また、上記の計算方法では逆元演算を用いないため、逆元演算をあらかじめ計算し、その値をメモリに格納する、ということを行なう必要がない。そのため、メモリ使用量が少なくてすむ。

【００８７】

また、上記の計算方法は、サイドチャネル攻撃に対する防御に関しても有効である。この理由は次の通りである。上記の計算方法で秘密鍵が初めて用いられるのはステップ1004であり、その後の演算は、ステップ1004の演算結果を用いる。ステップ1004の入力である整数tは、入力されたデータcと、ステップ1001で生成された乱数rの4乗との乗算結果である。したがって、攻撃者は整数tの値を予測することはできない。Novakの攻撃は、汎用べき乗剰余算に入力される値が攻撃者にとって既知であることを利用しているので、攻撃者はNovakの攻撃を行なうことができない。

【００８８】

以上の通り、上記の計算方法は、サイドチャネル攻撃に有用な情報を与えないため、サイドチャネル攻撃に対して耐性がある。また、逆元演算を用いないため、高速な計算が可能であり、さらに、逆元演算の結果をあらかじめメモリに格納する必要もないため、メモリ使用量も少なくてすむ。

【００８９】

なお、ステップ906で $-u_p$ を $p-u_p$ におきかえて実行してもよい。そうすると、負の整数を扱うことなく実行できるため、プログラムを簡略化できる。

【００９０】

以上、コンピュータＢ121が、暗号化されたデータ141を復号化する場合のデータ処理部132の動作を説明したが、コンピュータＡ101が入力データを暗号化する場合も同様である。

【００９１】

また、本発明は、署名作成方法をしても用いることができる。その場合には、暗号文cは、署名対象データcとして扱い、署名生成の際には、

$$m = c^d \bmod n \qquad \text{（式48）}$$

を計算し、mをデータcに対する署名として出力すればよい。

【００９２】

また、上記実施形態におけるコンピュータは、スマートカードや携帯電話、情報家電であってもよい。またその際、必要なデータはオンラインで上記コンピュータ内に入力され、べき乗剰余計算を行なってもよい

また、上記各実施形態におけるデータ処理部、ＣＲＴ計算部は、専用のハードウェアを用いて行なってもよい。モジュラー乗算部、モジュラー加減算部、乱数生成部、汎用べき乗剰余計算部、モジュラー剰余計算部、加減乗算部をコプロセッサまたはそれ以外の専用ハードウェアで実現してもよい。

【図面の簡単な説明】

【００９３】

【図１】実施形態におけるシステム構成図である。

【図２】第1実施例の実施形態におけるデータ処理部の構成図である。

【図３】第1実施例における復号化処理を示すフローチャート図である。

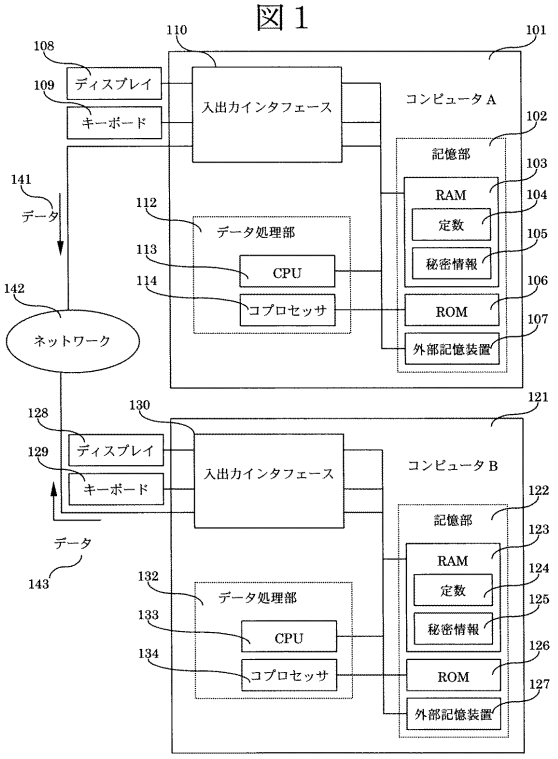
【図４】第１実施例の実施形態におけるＣＲＴ計算部の構成図である。

【図５】第１実施例におけるＣＲＴ計算部の行なう計算方法を示すフローチャート図である。

【図６】第1実施例におけるＣＲＴ計算部の行なう計算方法を示すフローチャート図である。
【図７】第1実施例におけるＣＲＴ計算部の行なう計算方法を示すフローチャート図である。
【図８】第２、第３実施例の実施形態におけるデータ処理部の構成図である。
【図９】第２実施例における復号化処理を示すフローチャートである。
【図１０】第３実施例における復号化処理を示すフローチャートである。
【符号の説明】
【００９４】
101、121：コンピュータ、102、122：記憶部、112、132：データ処理部、104、124、204、406、804：定数、105、125：秘密情報、110、130：入出力インターフェース、108、128：ディスプレイ、109、129：キーボード、142：ネットワーク、141、143：データ、201、401、801：モジュラー乗算部、202、802：乱数生成部、203、803：汎用べき乗剰余計算部、402、805：モジュラー加減算部、403：モジュラー剰余計算部、404：べき乗剰余計算部、405、806：加減乗算部。

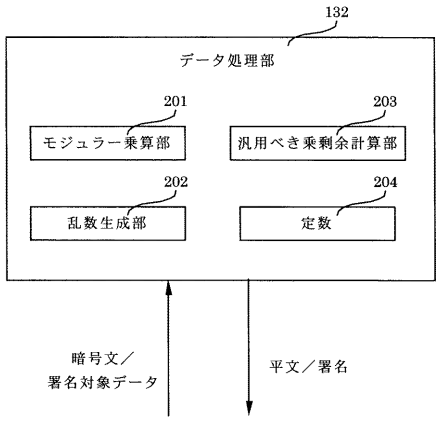
【図１】

図１



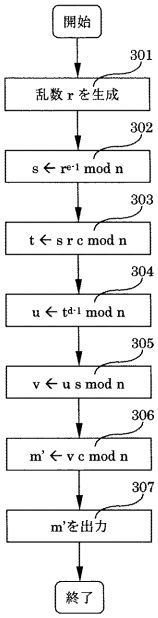
【図２】

図２



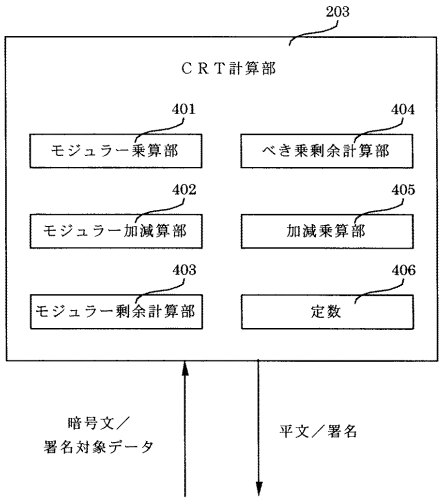
【図３】

図３



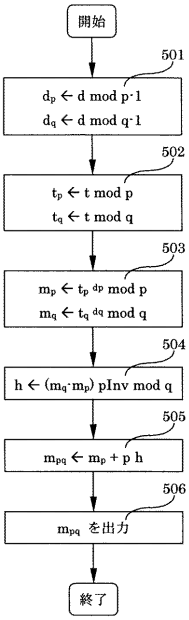
【図４】

図４



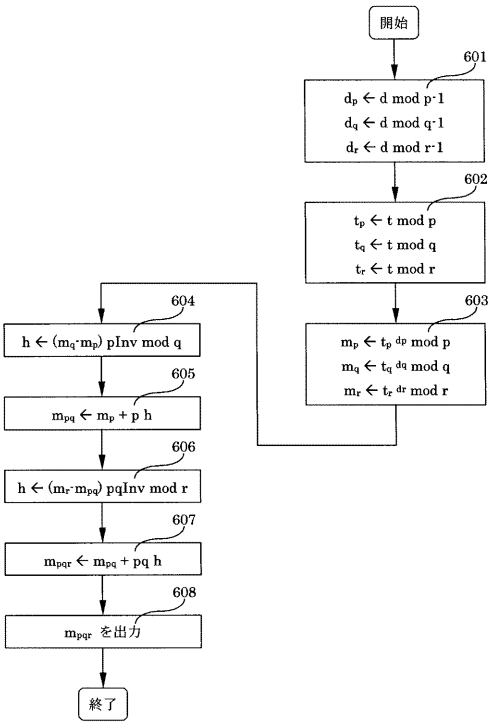
【図５】

図５



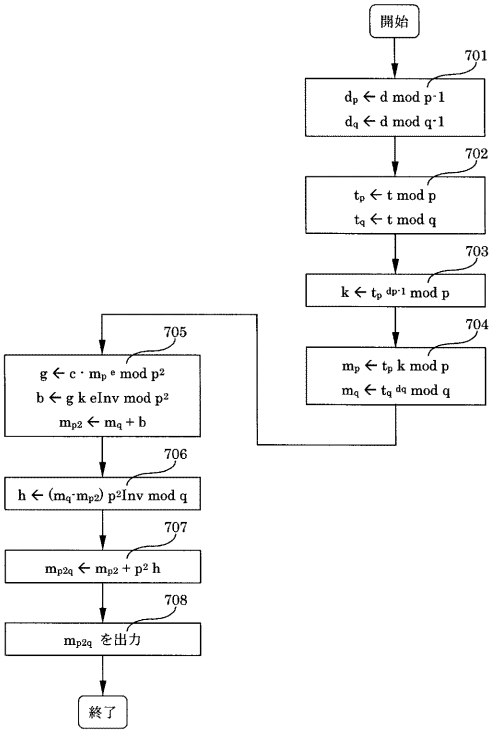
【図６】

図６



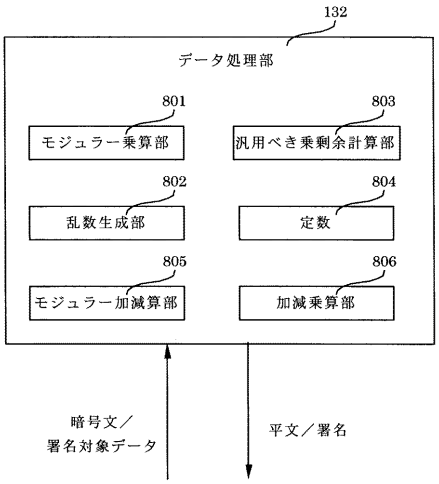
【図７】

図７



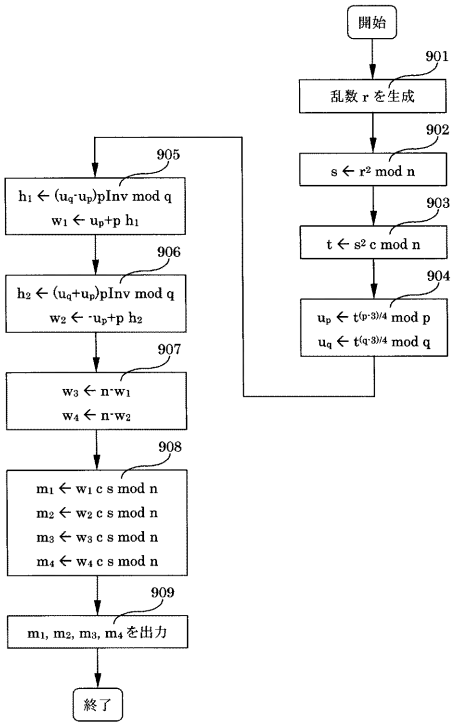
【図８】

図８



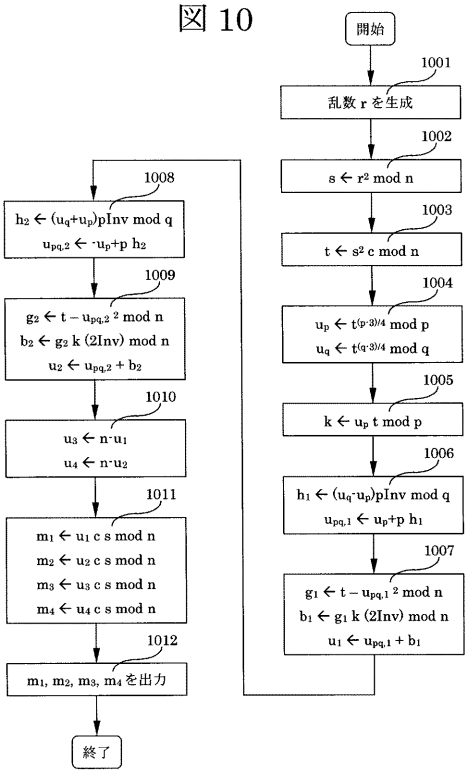
【図９】

図９



【図１０】

図１０



フロントページの続き

(56)参考文献 特開２００３－２０８０９７（ＪＰ，Ａ）
特開２００２－２４７０２５（ＪＰ，Ａ）
特開２００１－００５７３１（ＪＰ，Ａ）
特開２０００－１８２０１２（ＪＰ，Ａ）
国際公開第９９／０３５７８２（WO，Ａ１）
Katsuyuki Okeya and Tsuyoshi Takagi, “A More Flexible Countermeasure against Side Channel Attacks Using Window Method”, LMCS, ２００３年 ９月, Vol.2779, p.397-410

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０９Ｃ １／００