(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5224481号
(P5224481)

(45)発行日　平成25年7月3日(2013.7.3)　(24)登録日　平成25年3月22日(2013.3.22)

(51)Int.Cl.			F I		
H04L	***9/32***	***(2006.01)***	H04L	9/00	675B
H04L	***9/08***	***(2006.01)***	H04L	9/00	601C
			H04L	9/00	673A

請求項の数 27　(全 51 頁)

(21)出願番号	特願2010-519823 (P2010-519823)
(86)(22)出願日	平成21年7月10日 (2009.7.10)
(86)国際出願番号	PCT/JP2009/062578
(87)国際公開番号	WO2010/005071
(87)国際公開日	平成22年1月14日 (2010.1.14)
審査請求日	平成23年1月4日 (2011.1.4)
(31)優先権主張番号	特願2008-179670 (P2008-179670)
(32)優先日	平成20年7月10日 (2008.7.10)
(33)優先権主張国	日本国(JP)

(73)特許権者　301021533
独立行政法人産業技術総合研究所
東京都千代田区霞が関１－３－１
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100130409
弁理士　下山　治
(74)代理人　100148345
弁理士　駒木　寛隆

最終頁に続く

(54)【発明の名称】パスワード認証方法

(57)【特許請求の範囲】

【請求項１】

クライアント・サーバ間の相互認証方法であって、

上記サーバが実行する処理が、

（ａ）上記サーバで生成した乱数ｙ（ｙ∈（Ｚ／ｑＺ）＊）に基づいて、サーバ側マスター秘密Ｋｓを、式：

$$Ks = g^{y} \quad \cdots\cdots(式１)$$

によって計算するステップと、

（ｂ）上記クライアントで計算された第１クライアント情報Ｕを、第１識別情報（Ｃ，ＷＩＤ，ｉｄ）と共に該クライアントから受信するステップと、

（ｃ）上記受信した第１クライアント情報Ｕ及び第１識別情報を利用して、第１サーバ情報Ｙを、式：

$$Y = U^{y} \cdot W^{y \cdot r} \quad \cdots\cdots(式２)\ 又は$$

$$Y = U^{y \cdot r} \cdot W^{y} \quad \cdots\cdots(式２')$$

によって計算するステップと、

（ｄ）上記計算した第１サーバ情報Ｙを上記クライアントへ送信するステップと、

（ｅ）上記サーバ側マスター秘密Ｋｓを用いて、上記クライアントから受信したクライアント認証情報Ｖｃを認証するステップと、

（ｆ）上記サーバ側マスター秘密Ｋｓを用いて、サーバ認証情報Ｖｓを生成し上記Ｖｓをクライアントに送信するステップと、

を含み、
ただし上の記載において、ｑは群（Ｇ，・）の位数を表し、ｇはその集合Ｇの生成元を表し、”・”は上記Ｇ上の二項演算子を表し、
式（２）及び式（２’）における記号Ｗはパスワードｐｗに関する情報を含む部分を表し、上記クライアントが生成したパスワード情報ｖに基づいて
$W = g^{v}$ ……（式３）、
又は上記クライアントが生成したパスワード情報ｖと上記クライアントが生成した乱数ｔのコミット値Ｔに基づいて
$W = T^{v} = g^{t \times v}$ ……（式３’）
により得ることができる部分であり、該パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする関数H_1（）の出力から計算される値であり、×は整数上の掛け算であり、式（２）及び式（２’）における記号ｒは、上記クライアントが上記第１クライアント情報Ｕを計算するより前の時刻に知ることのできない値であると共に、上記サーバ及び上記クライアントのいずれにおいても計算可能な値を表し、Ｕ，Ｗ，Ｙ，Ｔ，ｇ∈Ｇであり、
上記クライアントが実行する処理が、
（ａ’）上記クライアントで生成した乱数ｕ（ｕ∈（Ｚ／ｑＺ）＊）に基づき、上記第１クライアント情報Ｕを、
$U = g^{u}$
によって計算するステップと、
（ｂ’）上記計算した第１クライアント情報Ｕを、上記第１識別情報と共に上記サーバに送信するステップと、
（ｃ’）上記第１クライアント情報Ｕの送信に応じて、上記サーバから上記第１サーバ情報Ｙを受信するステップと、
（ｄ’）上記$W = g^{v}$ …（式３）の場合に、
ブラインド値ｂを、上記パスワード情報ｖ及び上記値ｒに基づいて、次の計算式：
$b = u + v \times r \mod q$（$Y = U^{y} \cdot W^{y \cdot r}$である場合） ……（式４）又は
$b = u \times r + v \mod q$（$Y = U^{y \cdot r} \cdot W^{y}$である場合） ……（式４’）
によって計算するステップ、又は
上記 $W = T^{v} = g^{t \times v}$ …（式３’）の場合に、
上記クライアントが生成したパスワード情報ｖと上記クライアントが生成した乱数ｔに基づいて、前記ブラインド値ｂを、次の計算式：
$b = u + t \times v \times r \mod q$（$Y = U^{y} \cdot W^{y \cdot r}$である場合） ……（式５）又は
$b = u \times r + t \times v \mod q$（$Y = U^{y \cdot r} \cdot W^{y}$である場合） ……（式５’）
によって計算するステップと、
（ｅ’）受信した上記第１サーバ情報Ｙに基づいて、クライアント側マスター秘密Ｋｃを、
$Kc = Y^{(1/b \mod q)}$
によって計算するステップと、
（ｆ’）上記クライアント側マスター秘密Ｋｃを利用してクライアント認証情報Ｖｃを生成し、上記サーバに送信するステップと
（ｇ’）上記サーバから受信した上記サーバ認証情報Ｖｓを、上記クライアント側マスター秘密Ｋｃを利用して認証するステップと、
を含む、方法。ただし、
「１／ｂ ｍｏｄ ｑ」は「ａ×ｂ≡１ ｍｏｄ ｑ」を満たす１以上ｑ未満の整数ａを示す。
【請求項２】
上記値ｒは、上記第１クライアント情報Ｕがクライアントから送信された後にサーバからクライアントに送信される乱数、又は、少なくとも上記第１クライアント情報Ｕを一方向性関数F_2（）の入力として得られた出力より計算できる値である、

請求項１に記載の方法。
【請求項３】
上記パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする一方向性関数Ｆ$_1$（）の出力より計算できる値である、
請求項１、２のいずれかに記載の方法。
【請求項４】
上記パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする関数Ｈ$_1$（）又は一方向性関数Ｆ$_1$（）の出力と乱数ｓとを少なくとも結合した値、又は、上記パスワードｐｗと上記乱数ｓとを少なくとも結合した値である、
請求項１、２のいずれかに記載の方法。
【請求項５】
上記パスワード情報ｖは
ｖ＝ｓ＋ｈｐｗ　ｍｏｄ　ｑ　又は
ｖ＝ｓ×ｈｐｗ　ｍｏｄ　ｑ　又は
ｖ＝ｓ（＋）ｈｐｗ　又は
上記パスワードｐｗと乱数ｓを少なくとも入力とする関数Ｈ$_1$（）又は一方向性関数Ｆ$_1$（）の出力より計算できる値
と表記可能である、
請求項１、２、４のいずれかに記載の方法。
ここで、ｈｐｗは、上記パスワードｐｗを少なくとも入力とする関数Ｈ$_1$（）又は一方向性関数Ｆ$_1$（）の出力より計算できる値であり、（＋）は排他的論理和である。
【請求項６】
上記部分Ｗ又は上記パスワード情報ｖが、上記第１識別情報（Ｃ，ＷＩＤ，ｉｄ）に関連づけられて上記サーバの記憶装置に予め格納されており、
上記サーバが、上記受信した第１識別情報に基づいて該記憶装置から上記部分Ｗ（上記部分Ｗが格納されている場合）又は、上記パスワード情報ｖ（上記パスワード情報ｖが格納されている場合）を検索するステップを有する、
請求項１から５のいずれかに記載の方法。
【請求項７】
上記部分Ｗ又は上記パスワード情報ｖが、上記第１識別情報（Ｃ，ＷＩＤ，ｉｄ）を少なくとも入力に含む一方向性関数Ｈ$_3$（）の出力から計算される第２識別情報に関連づけられて上記サーバの記憶装置に予め格納されており、
上記サーバが、上記受信した第１識別情報（Ｃ，ＷＩＤ，ｉｄ）を少なくとも入力に含む一方向性関数Ｈ$_3$（）の出力から計算した上記第２識別情報により該記憶装置から上記部分Ｗ（上記部分Ｗが格納されている場合）又は、上記パスワード情報ｖ（上記パスワード情報ｖが格納されている場合）を検索するステップを有する、
請求項１から５のいずれかに記載の方法。
【請求項８】
上記クライアントがクライアント側改ざん検出子生成鍵、クライアント側改ざん検出子検証鍵を持ち、上記サーバが、上記第１識別情報又は上記第１識別情報（Ｃ，ＷＩＤ，ｉｄ）を少なくとも入力に含む一方向性関数Ｈ$_3$（）の出力から計算される第２識別情報と共に、上記クライアント側改ざん検出子生成鍵により生成された改ざん検出子を検証するためのサーバ側改ざん検出子検証鍵と、上記クライアント側改ざん検出子検証鍵で検証できる改ざん検出子を生成できるサーバ側改ざん検出子生成鍵を持ち、
上記クライアント・サーバ間の相互認証において、
（ａ）上記サーバが、上記クライアントから上記Ｕを受け取った場合に、少なくとも上記Ｕと上記Ｙの組をログリストＰｓ’に記録すると共に、少なくとも上記Ｕと上記Ｙに対して上記サーバ側改ざん検出子生成鍵を用いて改ざん検出子ｍａｃ２を生成し、それを上記クライアントに送信するステップと、
（ｂ）上記クライアントが、該サーバから送られてきた改ざん検出子ｍａｃ２の検証に失

敗した場合に、上記クライアント認証情報Ｖｃを送信せず該クライアント・サーバ間の相互認証を中断するステップと、
（ｃ）同じく上記クライアントが、該サーバから送られてきた改ざん検出子ｍａｃ２の検証に成功した場合には、少なくとも上記Ｙと上記クライアント認証情報Ｖｃに対して該クライアント側改ざん検出子生成鍵を用いて改ざん検出子ｍａｃ３を生成し、その改ざん検出子を該サーバに送信すると共に、少なくとも上記Ｕと上記Ｙの組をログリストＰｃに記録するステップと、
（ｄ）該サーバが、該クライアントから送られてきた改ざん検出子ｍａｃ３の検証に失敗した場合に、該クライアント・サーバ間の相互認証を中断するステップと、
（ｅ）同じく該サーバが、該クライアントから送られてきた改ざん検出子ｍａｃ３の検証に成功し、かつ、該クライアントから送られてきた上記クライアント認証情報Ｖｃの検証に失敗した場合に、少なくとも上記Ｕと上記Ｙの組をログリストＰｓに記録し、該クライアント・サーバ間の相互認証を中断するステップと、
（ｆ）該クライアント・サーバ間の相互認証が正常に終了した際には、前回該クライアント・サーバ間の相互認証ステップが正常に終了した以降に該サーバと該クライアントが該クライアント・サーバ間の相互認証処理中に記録し続けたログリストＰｓ、Ｐｃ、Ｐｓ’中の少なくともＵとＹの組を第三者に改ざんされない方法で比較するステップと、
を有する、請求項１から７のいずれかに記載の方法。
【請求項９】
上記サーバ側及び上記クライアント側の少なくとも一方における上記改ざん検出子生成鍵及び上記検出子検証鍵をＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）鍵とする請求項８に記載の方法。
【請求項１０】
上記サーバ側及び上記クライアント側の両方の改ざん検出子生成鍵及び検出子検証鍵をＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）鍵とする場合、
上記サーバ側の改ざん検出子生成鍵、
上記サーバ側の改ざん検出子検証鍵、
上記クライアント側の改ざん検出子生成鍵、
上記クライアント側の改ざん検出子検証鍵、
は全て同じ鍵とし、
上記クライアントと上記サーバが同じＭＡＣ鍵を使っても異なる改ざん検出子を生成できるように、上記クライアントと上記サーバとで、異なるＭＡＣ生成アルゴリズム、又は、異なるメッセージフォーマットを用いる、請求項９に記載の方法。
【請求項１１】
上記改ざん検出子生成鍵を電子署名生成鍵、上記検出子検証鍵を電子署名検証鍵とする請求項８に記載の方法。
【請求項１２】
上記クライアントがクライアント側データ鍵ｃｄｋを持ち、上記サーバがサーバ側データ鍵ｓｄｋを第１識別情報又は第２識別情報と共に持ち、上記クライアント・サーバ間の相互認証ステップが正常に終了した際に、
（ａ）該サーバが、該クライアント・サーバ間の相互認証ステップにより生成された上記サーバ側マスター秘密Ｋｓに依存して生成される暗号化鍵を使って暗号化されたサーバ側データ鍵ｓｄｋを該クライアントに送信するステップと、
（ｂ）該クライアントが、該サーバから送信された暗号化されたデータ鍵ｓｄｋを該クライアント・サーバ間の相互認証ステップにより生成された該クライアント側マスター秘密ＫＣに依存して生成される暗号化鍵を使って復号するステップと、
（ｃ）該クライアントが上記クライアント側データ鍵ｃｄｋと上記サーバ側データ鍵ｓｄｋからデータｄｋを復元するステップを持つ
請求項１から１１のいずれかに記載の方法。
【請求項１３】

上記クライアントがクライアント側データ鍵ｃｄｋを持ち、上記サーバがサーバ側データ鍵ｓｄｋを第１識別情報又は第２識別情報と共に持ち、上記クライアント・サーバ間の相互認証ステップが正常に終了した際に、
（ａ’）該クライアントが、該クライアント・サーバ間の相互認証ステップにより生成された上記クライアント側マスター秘密Ｋｃに依存して生成される暗号化鍵を使って暗号化されたクライアント側データ鍵ｃｄｋを該サーバに送信するステップと、
（ｂ’）該サーバが、該クライアントから送信された暗号化されたクライアント側データ鍵ｃｄｋを該クライアント・サーバ間の相互認証ステップにより生成された該サーバ側マスター秘密Ｋｓに依存して生成される暗号化鍵を使って復号するステップと、
（ｃ’）該サーバが上記クライアント側データ鍵ｃｄｋと上記サーバ側データ鍵ｓｄｋからデータｄｋを復元するステップを持つ
請求項１から１２のいずれかに記載の方法。
【請求項１４】
上記データｄｋは、
ｄｋ’＝ｃｄｋ（＋）ｓｄｋ　又は
ｄｋ’＝ｃｄｋ＋ｓｄｋ　ｍｏｄ　ｑ　又は
ｄｋ’＝ｃｄｋ×ｓｄｋ　ｍｏｄ　ｑ
とし、
ｄｋ＝ｄｋ’　　又は
ｄｋ＝（ｄｋ’）（＋）ｈｐｗ　又は
ｄｋ＝（ｄｋ’）＋ｈｐｗ　ｍｏｄ　ｑ　又は
ｄｋ＝（ｄｋ’）×ｈｐｗ　ｍｏｄ　ｑ　又は
少なくともｄｋ’とｈｐｗを入力とする関数H_3（）の出力から計算できる値により復元できる、
請求項１２または１３に記載の方法。
ここで、（ｄｋ’）はｄｋ’の値を最初に計算するというステップを示しており、ｈｐｗは、上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力より計算できる値であり、（＋）は排他的論理和である。
【請求項１５】
上記クライアント・サーバ間の相互認証のために、上記クライアントが利用する可能性のある情報である、上記乱数ｔ、上記第１識別情報の全てあるいは一部を、該クライアントの記録装置に予め保存し、該クライアントが上記クライアント・サーバ間の相互認証を実行する祭に、上記クライアント・サーバ間の相互認証の要求に応じて呼び出す
請求項１から１４のいずれかに記載の方法。
【請求項１６】
上記クライアント・サーバ間の相互認証が成功した場合、
その相互認証で利用した該サーバの該記憶装置に記録されている情報である、上記第１識別情報、上記部分Ｗ又は上記パスワード情報ｖ、クライアントが生成した乱数ｔのコミット値Ｔ、の全てあるいは一部、並びに、
該クライアントの該記憶装置に記録されている情報である、上記乱数ｔ、上記第１識別情報、の全てあるいは一部を、
該クライアントと該サーバとの間でやりとりされた値、又は、該認証ステップで共有された上記マスター秘密ＫＣ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを有する、
請求項１から１５のいずれかに記載の方法。
【請求項１７】
上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
ｖ＝ｓ＋ｈｐｗ　ｍｏｄ　ｑ　または
ｖ＝ｓ×ｈｐｗ　ｍｏｄ　ｑ

により計算することができ、
上記更新するステップにおいて、上記サーバは、
$W' = W \cdot g^{ud}$ または
$W' = Wud$
と表記されうるように上記部分WをW'に更新し、上記クライアントは、
$s' = s + ud \mod q$ または
$s' = s \times ud \mod q$
と表記されうるように上記乱数ｓをｓ’に更新する、請求項１６に記載の方法。
ただし、ｈｐｗは上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力であり、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【請求項１８】
上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
$v = s + hpw \mod q$
により計算することができ、
上記更新するステップにおいて、上記サーバは、
$v' = v + ud \mod q$
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
$s' = s + ud \mod q$
と表記されうるように上記乱数ｓをｓ’に更新する、請求項１６に記載の方法。
ただし、ｈｐｗは上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力であり、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【請求項１９】
上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
$v = s (+) hpw$
により計算することができ、
上記更新するステップにおいて、上記サーバは、
$v' = v (+) ud$
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
$s' = s (+) ud$
と表記されうるように上記乱数ｓをｓ’に更新する、請求項１６に記載の方法。
ただし、ｈｐｗは上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力であり、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値であり、（＋）は排他的論理和を表す。
【請求項２０】
上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
$v = s \times hpw \mod q$
により計算することができ、
上記更新するステップにおいて、上記サーバは、
$v' = v \times ud \mod q$
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
$s' = s \times ud \mod q$
と表記されうるように上記乱数ｓをｓ’に更新する、請求項１６に記載の方法。
ただし、ｈｐｗは上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力であり、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【請求項２１】
上記クライアント・サーバ間の相互認証が成功した場合、

その相互認証で利用した該サーバの該記憶装置に記録されている情報である、該サーバ側の改ざん検出子生成鍵、該サーバ側の改ざん検出子検証鍵の全てあるいは一部、及び、
該クライアントの該記憶装置に記録されている情報である、上記クライアント側の改ざん検出子生成鍵、上記クライアント側の改ざん検出子検証鍵の全てあるいは一部を、
該クライアントと該サーバとの間でやりとりされた値、又は、該認証処理で共有された上記マスター秘密ＫＣ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを有し、
上記サーバと上記クライアントが同じ上記ＭＡＣ鍵ＭａｃＫを用いる場合、上記サーバと上記クライアントがそれぞれ上記ＭＡＣ鍵ＭａｃＫをＭａｃＫ’に更新するステップは、

ＭａｃＫ’＝ＭａｃＫ（＋）ｕｄ　又は
ＭａｃＫ’＝ＭａｃＫ＋ｕｄ　ｍｏｄ　ｑ　又は
ＭａｃＫ’＝ＭａｃＫ×ｕｄ　ｍｏｄ　ｑ　又は

と表記可能である、
請求項１０に記載の方法。
ただし、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【請求項２２】
上記クライアント・サーバ間の相互認証が成功した場合、
その相互認証で利用した該サーバの該記憶装置に記録されている情報である上記サーバ側データ鍵ｓｄｋ及び、
該クライアントの該記憶装置に記録されている情報である上記クライアント側データ鍵ｃｄｋを、
該クライアントと該サーバとの間でやりとりされた値、又は、該認証処理で共有された上記マスター秘密ＫＣ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを有し、
上記クライアントが上記クライアント側データ鍵ｃｄｋを持ち、上記サーバが上記サーバ側データ鍵ｓｄｋを持つ場合、上記サーバと上記クライアントがそれぞれｃｄｋとｓｄｋをｃｄｋ’とｓｄｋ’に更新するステップは、
上記ｄｋ’が

ｄｋ’＝ｃｄｋ（＋）ｓｄｋ

と表記可能な場合、

ｃｄｋ’＝ｃｄｋ（＋）ｕｄ
ｓｄｋ’＝ｓｄｋ（＋）ｕｄ

と表記可能であり、
上記ｄｋ’が

ｄｋ’＝ｃｄｋ＋ｓｄｋ　ｍｏｄ　ｑ

と表記可能な場合、

ｃｄｋ’＝ｃｄｋ＋ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ－ｕｄ　ｍｏｄ　ｑ

又は

ｃｄｋ’＝ｃｄｋ－ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ＋ｕｄ　ｍｏｄ　ｑ

と表記可能であり、
上記ｄｋ’が

ｄｋ’＝ｃｄｋ×ｓｄｋ　ｍｏｄ　ｑ

と表記可能な場合、

ｃｄｋ’＝ｃｄｋ×ｕｄ　ｍｏｄ　ｑ

ｓｄｋ’＝ｓｄｋ　／　ｕｄ　ｍｏｄ　ｑ
又は
ｃｄｋ’＝ｃｄｋ　／　ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ×ｕｄ　ｍｏｄ　ｑ
と表記可能である、
請求項１４に記載の方法。
ただし、ｕｄは上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値であり、－は整数上の引き算、「ａ＝ｃ／ｂ　ｍｏｄ　ｑ」はｃが０でない場合に、「ａ×ｂ≡ｃ　ｍｏｄ　ｑ」を満たす１以上ｑ未満の整数ａを示す。

【請求項２３】
クライアント装置とサーバ装置から構成されるシステムであって、該クライアント装置と該サーバ装置が請求項１から２２のいずれかに記載の相互認証方法を実行するように構成される、システム。

【請求項２４】
請求項１から２２のいずれかに記載の相互認証方法における、クライアント側で実行される処理を遂行しうるように構成される、コンピュータ装置。

【請求項２５】
請求項１から２２のいずれかに記載の相互認証方法における、サーバ側で実行される処理を遂行しうるように構成される、コンピュータ装置。

【請求項２６】
コンピュータ装置のＣＰＵで実行されることにより、該コンピュータ装置に、請求項１から２２のいずれかに記載の相互認証方法における、クライアント側で実行される処理を遂行させる、コンピュータ・プログラム。

【請求項２７】
コンピュータ装置のＣＰＵで実行されることにより、該コンピュータ装置に、請求項１から２２のいずれかに記載の相互認証方法における、サーバ側で実行される処理を遂行させる、コンピュータ・プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、パスワードを用いた認証方法に関し、特に、パスワードのみを用いてクライアント・サーバ間の相互認証を行う認証方法に関する。

【背景技術】
【０００２】
これまでパスワードのみを用いて認証を行う認証方法はいくつか提案されている。その中には攻撃者の通信盗聴によりパスワードの辞書攻撃が可能なものもある。より高い安全性を達するためには、パスワードのみを用いた認証方法では、インターネットのようなパブリックネットワーク上のあらゆる攻撃（通信盗聴、リプレイ攻撃、メッセージ改ざん、成りすまし、ｍａｎ－ｉｎ－ｔｈｅ－ｍｉｄｄｌｅ攻撃など）に耐性を有することが好ましい。かかる要求に関しては、離散対数問題に安全性の根拠をもつ認証方式であって、パブリックネットワーク上のあらゆる攻撃だけではなく、攻撃者がサーバの記録情報を用いてクライアントのなりすましを行うＫＣＩ（Ｋｅｙ　Ｃｏｍｐｒｏｍｉｓｅ　Ｉｍｐｅｒｓｏｎａｔｉｏｎ）攻撃にも安全な認証方式が知られている。しかし、このような安全性を有する従来の認証方式における問題点は、クライアント及びサーバともに計算量（すなわち、べき乗剰余演算の回数）を最低限に抑えられないことである。クライアント側においては、ユーザの端末装置が小型の遅い装置、旧世代のパーソナルコンピュータ、スマートカード、あるいは携帯型個人情報端末（ＰＤＡ）である場合があるため、計算量はできるだけ抑えることが望ましい。サーバ側においても、非常に多くのユーザを管理する必要があり、また計算能力が高くない場合もあるため、やはり計算量はできるだけ低いほうが

望ましい。
【０００３】
特許文献１に記載される方法では、Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ型鍵交換を利用し、パスワードのみを共有する二者間で、データネットワークを通じて安全に相互認証を行っている。しかしながら、特許文献１の図２及び図３の実施例はＫＣＩ攻撃に安全ではないし、図４及び図５のパスワードベリファイアを利用した実施例は、ＫＣＩ攻撃に安全であるものの、本発明者の提案による認証方法に比べてクライアントもサーバも多くの計算量を必要とする。
【０００４】
特許文献２に記載される方法は、特許文献１の図２に記載される方式の計算効率を改善したものであり、クライアント側の計算量を少なくとも半分に低減することができる。しかしながら、この方法はＫＣＩ攻撃に対して安全ではない。また、ＫＣＩ攻撃に対して安全にするために、特許文献１に記載された方式（図４及び図５のパスワードベリファイアを利用した実施例）のように、特許文献２の方法を変形することはできるものの、本発明者の提案による認証方法に比べてクライアントもサーバも多くの計算量を必要とする。
【０００５】
特許文献３に記載される方法では、端末と認証サーバが予めパスワードと暗号鍵を共有し、端末はパスワードを暗号鍵で暗号化して認証サーバへ送信することで認証を行い、認証が成功した場合は従来方式でデータ通信用暗号鍵を交換する。しかしながら、端末は暗号鍵を安全に保存するためにデバイスの耐タンパー性が必要となり、暗号鍵が漏えいした場合は、以前に通信した暗号文からパスワードを取り出すことができる。つまり、本発明者の提案による認証方法に比べて安全性が落ちるといえる。
【先行技術文献】
【特許文献】
【０００６】
【特許文献１】特開２００１－３１３６３４号公報
【特許文献２】特開２００２－３３５２３８号公報
【特許文献３】特開２００６－１９７０６５号公報
【非特許文献】
【０００７】
【非特許文献１】”ＡＭＰ”，ＩＥＥＥ Ｐ１３６３－２，Ｓｔａｎｄａｒｄ ｓｐｅｃｉｆｉｃａｔｉｏｎｓ ｆｏｒ ｐａｓｓｗｏｒｄ－ｂａｓｅｄ ｐｕｂｌｉｃ ｋｅｙ ｃｒｙｐｔｏｇｒａｐｈｉｃ ｔｅｃｈｎｉｑｕｅｓ
【非特許文献２】”ＫＡＭ－３”，ＩＳＯ／ＩＥＣ ＳＣ２７ ＦＣＤ １１７７０－４，Ｉｎｆｏｒｍａｔｉｏｎ ｔｅｃｈｎｏｌｏｇｙ － Ｓｅｃｕｒｉｔｙ ｔｅｃｈｎｉｑｕｅｓ － Ｋｅｙ ｍａｎａｇｅｍｅｎｔ － Ｐａｒｔ ４： Ｍｅｃｈａｎｉｓｍｓ ｂａｓｅｄ ｏｎ ｗｅａｋ ｓｅｃｒｅｔｓ
【発明の概要】
【発明が解決しようとする課題】
【０００８】
本発明は、ＫＣＩ攻撃を含めてパブリックネットワーク上の様々な攻撃に対して安全である認証方法であって、必要とされる計算量を従来に比べて減少させうる認証方法を提案しようとして生まれたものである。
【課題を解決するための手段】
【０００９】
本発明の認証方法は、認証処理を実行する時点においてはパスワード入力のみをユーザに要求するタイプの相互認証処理技術の認証方法であり、また、Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ型鍵交換を利用する相互認証処理技術の改良であると位置づけられる。
【００１０】
Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵交換技術では、一般に、クライアントが乱数ｕ（ｕ∈

（Ｚ／ｑＺ）＊）に基づいて、値Ｕ＝ｇ^{u}を計算し、サーバへ送信すると共に、サーバも乱数ｙ（ｙ∈（Ｚ／ｑＺ）＊）に基づいて、値Ｙ＝ｇ^{y}を計算し、クライアントへ送信する。その後、クライアント・サーバ共に、マスター秘密Ｋを、

Ｋ＝ｇ^{uy}　……（式１）

によって計算し、この秘密Ｋに基づいてセッション鍵を生成する。

【００１１】

Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ型鍵交換に基づく相互認証処理技術は、非特許文献１および非特許文献２に記載されているように、効率よく且つＫＣＩ攻撃に対して安全であるとして従来から知られている。これらの文献による認証技術では、クライアント・サーバ共に、マスター秘密Ｋを、

Ｋ＝ｇ$^{y(u+r')}$　……（式２）

により計算している。

ここで、

ｒ’＝Ｈ$_{2}$（Ｃ‖Ｓ‖Ｕ‖Ｙ）あるいはｒ’＝１；

Ｕ＝ｇ^{u}；

Ｙ＝Ｕ$^{y\cdot r}$・Ｗ^{y}；

ｒ＝Ｈ$_{2}$（Ｃ‖Ｓ‖Ｕ）；

Ｗ＝ｇ$^{H1(C\|S\|pw)}$；

Ｃ：クライアント装置の識別子；

Ｓ：サーバ装置の識別子；

ｐｗ：ユーザが覚えているパスワード

である。

これらの認証方法において、位数ｑの群を（Ｇ，・）とし、その集合Ｇの生成元をｇとし、Ｕ，Ｗ∈Ｇとする。また、”・”はＧ上の二項演算子であり、ｇ$_{1}$，ｇ$_{2}$∈Ｇとして、ｇ$_{1}$・ｇ$_{2}$をｇ$_{1}$ｇ$_{2}$として表記し、ｇ$_{1}$・ｇ$_{1}$をｇ$_{1}^{2}$として表記し、ｇ$_{1}^{i}$・ｇ$_{1}^{j}$をｇ$_{1}^{i+j}$として表記している。パスワードｐｗ及びパスワード認証データＷを利用することにより、ＫＣＩ攻撃等に対する安全性が確保されている。

【００１２】

ところで、本発明者は、サーバにおけるマスター秘密Ｋｓの計算式を、次のように修正しても、ＫＣＩ攻撃に対して安全となりうることを見出した。

Ｋｓ＝ｇ^{y}　……（式３）

ここで、ｙは前述の非特許文献１および非特許文献２の場合と同様であり、ｙはサーバで発生した乱数である（ｙ∈（Ｚ／ｑＺ）＊）。前述の場合と同様に、これらの認証方法においては、位数ｑの群を（Ｇ，・）とし、その集合Ｇの生成元をｇとし、Ｕ，Ｗ∈Ｇとする。また、”・”はＧ上の二項演算子であり、ｇ$_{1}$，ｇ$_{2}$∈Ｇとして、ｇ$_{1}$・ｇ$_{2}$をｇ$_{1}$ｇ$_{2}$として表記し，ｇ$_{1}$・ｇ$_{1}$をｇ$_{1}^{2}$として表記し，ｇ$_{1}^{i}$・ｇ$_{1}^{j}$をｇ$_{1}^{i+j}$として表記している。

【００１３】

上記の（式２）と（式３）を比べると分かるように、（式３）においては（ｕ＋ｒ’）のべき乗剰余演算が不要となっている。これは、サーバで遂行すべきマスター秘密Ｋｓの計算量に著しい違いをもたらす。（式３）を用いることにより、サーバにおける計算量を大幅に減少させることができる。

【００１４】

さらに、（式３）においてはマスター秘密Ｋｓの計算に、クライアントから供給される情報であるｕ，Ｕ，Ｃが不要であるため、マスター秘密Ｋｓを事前に計算しておくことが可能である。すなわち、認証を受けるべく、クライアントがサーバに接続してくる前に、予めマスター秘密Ｋｓを計算しておくことができる。従って、クライアントが接続してきてからのサーバの計算量が少なくてすみ、短時間に認証処理を終えることができる。

【００１５】

従来、Ｄｉｆｆｉｅ－Ｈｅｌｌｍａｎ鍵交換技術に基づく認証方法においては、安全性

を確保するために、ｕに関するべき乗計算は必須であると考えられていた。これに対して本発明者は、ｕに関するべき乗計算を除いても、安全性を確保しうる認証方法を発明するに至った。本発明に基づけば、ＫＣＩ攻撃等に対して安全な認証処理を、従来に比べて少ない計算量で実現することができ、且つ従来よりも短時間に処理を完了することを実現することができる。

【発明の効果】

【００１６】

本発明によれば、（式３）によってサーバ側マスター秘密Ｋｓを計算する場合、クライアントからの値Ｕの受信に応答してサーバが返信する値Ｙは、次のように計算される。

$$Y = U^{y \cdot W^{y \cdot r}} \quad \cdots\cdots(式４)$$ 又は

$$Y = U^{y \cdot r \cdot W^{y}} \quad \cdots\cdots(式４')$$

ここで、ｒは関数H_2（）にＵ（あるいはＵとほかの情報）を入力として求めた値であり、例えば、非特許文献１における方法と同様に、

$$r = H_2(C \| S \| U)$$

として計算することができる。ここで、Ｃはクライアント装置の識別子であり、Ｓはサーバ装置の識別子である。Ｗはクライアント識別子Ｃに対応してサーバの記憶装置に格納されるパスワード認証データ（パスワードベリファイア）であり、同様に、

$$W = g^{H1(C \| S \| pw)}$$

などによって計算することができる。Ｗは予め計算しておき、サーバの記憶装置に格納しておくことができる。

【００１７】

また、クライアントにおけるマスター秘密ＫＣの計算は、次の式によって行うことができる。

$$K_C = Y^{(1/b \bmod q)} \quad \cdots\cdots(式５)$$

ただし、

$b = u + PW \times r \bmod q$（$Y = U^{y \cdot W^{y \cdot r}}$である場合）　……（式６）又は

$b = u \times r + PW \bmod q$（$Y = U^{y \cdot r \cdot W^{y}}$である場合）　……（式６'）

である。ここで、ＰＷは、ユーザが入力するパスワードｐｗ（あるいはｐｗとほかの情報）を関数H_1（）に入力して求めた値であり、例えば、

$$PW = H_1(C \| S \| pw)$$

で計算することができる。なお、ｐｗは上記Ｗの計算の基礎になったパスワードと同じパスワードである。また、ｒは（式４）に関して説明したと同様に、関数H_1（）にＵ（あるいはＵとほかの情報）を入力として求めた値であり、例えば、

$$r = H_2(C \| S \| U)$$

として計算することができる。

【００１８】

上述のＹやｂの計算式において、（式４'）及び（式６'）は非特許文献１や２にも現れているが、（式４）及び（式６）は、本発明者の発明によるものである。

【００１９】

式５に示されるクライアント側マスター秘密ＫＣの計算量は、これまで知られていたＤｉｆｆｉｅ－Ｈｅｌｌｍａｎ型鍵交換に基づく相互認証処理技術において、最も計算量が少なかった非特許文献１と２に係る方法と同じ計算量である。従って、本明細書に開示される認証処理技術は、サーバ装置だけでなく、クライアント装置についても、計算量が極めて少ない方式であると言える。

【００２０】

さて、（式３）に示すとおりにマスター秘密に関わる計算量を減少させながら、ＫＣＩ攻撃に安全になることができる理由を説明する。はじめに（式４）から説明する。まず、攻撃者がサーバの記憶装置に格納されているパスワード認証データＷを取得したと仮定する。ＫＣＩ攻撃は、サーバの認証情報を得た攻撃者が、パスワードの辞書攻撃をせずに、クライアントになりすまして行う攻撃である。その攻撃を一般化して説明すると、攻撃者

はサーバにＵを以下のように計算して送信する。
$U = g^{c} \cdot g^{PW \cdot d}$
ここで、ｃとｄは攻撃者が生成した乱数（ｃ，ｄ∈（Ｚ／ｑＺ）＊）である。また、$W = g^{PW}$であり、$PW = H_1$（Ｃ‖Ｓ‖ｐｗ）である。このＵを受信したサーバは以下のようにＹを計算して返す。
$Y = U^{y} \cdot W^{y \cdot r}$
攻撃者がＫＣＩ攻撃を成功させるためには、$Y^{a} = K$になるようなａを探さなければならない。つまり、
（ｃ＋ＰＷ×ｄ＋ＰＷ×ｒ）ｙ×ａ＝ｙ　ｍｏｄ　ｑ
を解かなければならない。上の式は以下のように簡単になる。
（ｃ＋ＰＷ（ｄ＋ｒ））ａ＝１　ｍｏｄ　ｑ
上の式の解は
ｃ×ａ＝１と（ｄ＋ｒ）ａ＝０
になる。ここで、攻撃者はパスワードの辞書攻撃をやらずにＫＣＩ攻撃を行うため、ＰＷに係る項が０になる必要がある。ｃ×ａ＝１で、ａは０ではないので、（ｄ＋ｒ）ａ＝０でｄ＋ｒ＝０になるしかない。要するに、攻撃者はＵを計算する時に、ｄとして「－ｒ　ｍｏｄ　ｑ」のような値を使わなければならない。ところがｒはＵにより決まる（一方向性関数の）ハッシュした値であるため、ｄを求めることはできない。言い換えると、ＫＣＩ攻撃ができないということである。（式４’）についても同じように説明ができるが、ここでは省略する。

【００２１】
本発明によれば、認証処理に関わるＣＰＵの負荷を低下させることができ、処理能力の低い端末装置や、大量のクライアント装置からの要求を処理しなければならないサーバ装置などにとって、特に有益である。また、サーバにおいては、上述のようにマスター秘密を予め計算しておくことが可能になるため、クライアントからの認証要求を受けてからの処理を極めて短時間に完了することが可能となる。本発明は、ユーザやサーバ認証を必要とするサービスやアプリケーションにおいて広く利用することができ、例えばサーバやネットワークへのログイン、さらには電子商取引などで利用することができる。

【００２２】
本発明は、パスワードのみを用いた認証方式のみならず、他の認証手段を追加した認証方式にも応用することができ、例えば、二要素認証方式へ応用することができる。本発明の応用である二要素認証方式は、ユーザの端末装置が耐タンパモジュールなどを備えていない場合はユビキタス環境での認証方式として適しているし、ユーザの端末装置が耐タンパモジュールを用いる場合はもっと高い安全性を要求するインターネットバンキングなどに使うことができる。

【００２３】
本発明者の提案に基づく好適な具現化形態のいくつかは、添付の請求の範囲に特定されている。しかしながら、その具現化形態は、請求の範囲や明細書及び図面に明示的に記載されるものに限定されず、明細書に開示される発明の範囲を逸脱することなく、様々な態様を呈することが可能である。明細書に開示される発明の具現化形態は、請求の範囲や明細書及び図面に明示的に開示されるか否かにかかわらず、これらの書類から教示されうるあらゆる新規かつ有益な構成やそれらの組み合わせを、その範囲に含むものである。

【図面の簡単な説明】

【００２４】
【図１】実施例１の概要を説明するための図
【図２】実施例１における初期化段階を説明するための図
【図３】実施例１におけるクライアント装置の構成及び機能を説明するための図
【図４】実施例１におけるサーバ装置の構成及び機能を説明するための図
【図５】実施例２の概要を説明するための図
【図６】実施例２における初期化段階を説明するための図

【図７】実施例２におけるクライアント装置の構成及び機能を説明するための図
【図８】実施例２におけるサーバ装置の構成及び機能を説明するための図
【図９】実施例３の概要を説明するための図
【図１０】実施例３における初期化段階を説明するための図
【図１１】実施例３におけるクライアント装置の構成及び機能を説明するための図
【図１２】実施例３におけるサーバ装置の構成及び機能を説明するための図
【図１３】実施例４の概要を説明するための図
【図１４】実施例４における初期化段階を説明するための図
【図１５】実施例４におけるクライアント装置の構成及び機能を説明するための図
【図１６】実施例４におけるサーバ装置の構成及び機能を説明するための図
【図１７】実施例４におけるｊ番目プロトコル終了後の処理について説明するための図
【発明を実施するための形態】
【００２５】
本発明の実施形態は、クライアント・サーバ間における次のような相互認証方法を含む。この方法は、
サーバが実行する処理が、
（ａ）上記サーバで生成した乱数ｙ（ｙ∈（Ｚ／ｑＺ）＊）に基づいて、サーバ側マスター秘密Ｋｓを、式：

$$Ks = g^{y} \quad \cdots\cdots(式７)$$

によって計算するステップと、
（ｂ）上記クライアントで計算された第１クライアント情報Ｕを、第１識別情報（Ｃ，ＷＩＤ，ｉｄ）と共に該クライアントから受信するステップと、
（ｃ）上記受信した第１クライアント情報Ｕ及び第１識別情報を利用して、第１サーバ情報Ｙを、式：

$$Y = U^{y} \cdot W^{y \cdot r} \quad \cdots\cdots(式８)、又は$$

$$Y = U^{y \cdot r} \cdot W^{y} \quad \cdots\cdots(式８')$$

によって計算するステップと、
（ｄ）上記計算した第１サーバ情報Ｙを上記クライアントへ送信するステップと、
（ｅ）上記サーバ側マスター秘密Ｋｓを用いて、上記クライアントから受信したクライアント認証情報Ｖｃを認証するステップと、
（ｆ）上記サーバ側マスター秘密Ｋｓを用いて、サーバ認証情報Ｖｓを生成し、上記Ｖｓをクライアントに送信するステップと、
を含み、
ただし、上の記載において、ｑは群（Ｇ，・）の位数を表し、ｇはその集合Ｇの生成元を表し、”・”は上記Ｇ上の二項演算子を表し、
式（８）及び式（８'）における記号Ｗはパスワードｐｗに関する情報を含む部分を表し、上記クライアントが生成したパスワード情報ｖに基づいて

$$W = g^{v} \quad \cdots\cdots(式９)、$$

又は上記クライアントが生成したパスワード情報ｖと上記クライアントが生成した乱数ｔのコミット値Ｔに基づいて

$$W = T^{v} = g^{t \times v} \quad \cdots\cdots(式９')$$

により得ることができる部分であり、該パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする関数H_1（）の出力から計算される値であり、×は整数上の掛け算であり、式（８）及び式（８'）における記号ｒは、上記クライアントが上記第１クライアント情報Ｕを計算するより前の時刻に知ることのできない値であると共に、上記サーバ及び上記クライアントのいずれにおいても計算可能な値を表し、
Ｕ，Ｗ，Ｙ，Ｔ，ｇ∈Ｇであり、
クライアントが実行する処理が、
（ａ'）上記クライアントで生成した乱数ｕ（ｕ∈（Ｚ／ｑＺ）＊）に基づき、上記第１クライアント情報Ｕを、$U = g^{u}$ によって計算するステップと、

（ｂ’）上記計算した第１クライアント情報Ｕを、上記第１識別情報と共に上記サーバに送信するステップと、
（ｃ’）上記第１クライアント情報Ｕの送信に応じて、上記サーバから上記第１サーバ情報Ｙを受信するステップと、
（ｄ’）上記　Ｗ＝ｇ^{v}（式９）の場合に、
ブラインド値ｂを、上記パスワード情報ｖ及び上記値ｒに基づいて、次の計算式：
ｂ＝ｕ＋ｖ×ｒ　ｍｏｄ　ｑ（Ｙ＝Ｕ^{y}・Ｗ$^{y\cdot r}$である場合）　……（式１０）、又は
ｂ＝ｕ×ｒ＋ｖ　ｍｏｄ　ｑ（Ｙ＝Ｕ$^{y\cdot r}$・Ｗ^{y}である場合）　……（式１０’）
によって計算するステップ、又は
上記　Ｗ＝Ｔ^{v}＝ｇ$^{t\times v}$（式９’）の場合に、
上記クライアントが生成したパスワード情報ｖと上記クライアントが生成した乱数ｔに基づいて、前記ブラインド値ｂを、次の計算式：
ｂ＝ｕ＋ｔ×ｖ×ｒ　ｍｏｄ　ｑ（Ｙ＝Ｕ^{y}・Ｗ$^{y\cdot r}$である場合）……（式１１）、又は
ｂ＝ｕ×ｒ＋ｔ×ｖ　ｍｏｄ　ｑ（Ｙ＝Ｕ$^{y\cdot r}$・Ｗ^{y}である場合）……（式１１’）
によって計算するステップと、
（ｅ’）受信した上記第１サーバ情報Ｙに基づいて、クライアント側マスター秘密Ｋ$_{C}$を、
Ｋ$_{C}$＝Ｙ$^{（１／ｂ　ｍｏｄ　ｑ）}$
によって計算するステップと、
（ｆ’）上記クライアント側マスター秘密Ｋ$_{C}$を利用してクライアント認証情報Ｖｃを生成し、上記サーバに送信するステップと
（ｇ’）上記サーバから受信した上記サーバ認証情報Ｖｓを、上記クライアント側マスター秘密Ｋｃを利用して認証するステップと、
を含む。ただし、「１／ｂ　ｍｏｄ　ｑ」は「ａ×ｂ≡１　ｍｏｄ　ｑ」を満たす１以上ｑ未満の整数ａを示す。

【００２６】
実施形態によっては、上記値ｒは、上記第１クライアント情報Ｕがクライアントから送信された後にサーバからクライアントに送信される乱数、又は、少なくとも上記第１クライアント情報Ｕを一方向性関数Ｆ$_{2}$（）の入力として得られた出力より計算できる値であることができる。

【００２７】
実施形態によっては、上記パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする一方向性関数Ｆ$_{1}$（）の出力より計算できる値であることができる。

【００２８】
実施形態によっては、上記パスワード情報ｖは、上記パスワードｐｗを少なくとも入力とする関数Ｈ$_{1}$（）又は一方向性関数Ｆ$_{1}$（）の出力と乱数ｓとを少なくとも結合した値、又は、上記パスワードｐｗと上記乱数ｓとを少なくとも結合した値であることができる。

【００２９】
実施形態によっては、上記パスワード情報ｖは、
ｖ＝ｓ＋ｈｐｗ　ｍｏｄ　ｑ　又は
ｖ＝ｓ×ｈｐｗ　ｍｏｄ　ｑ　又は
ｖ＝ｓ（＋）ｈｐｗ　又は
上記パスワードｐｗと上記乱数ｓを少なくとも入力とする関数Ｈ$_{1}$（）又は一方向性関数Ｆ$_{1}$（）の出力より計算できる値と表記可能である。ただし、ｈｐｗは、上記パスワードｐｗを少なくとも入力とする関数Ｈ$_{1}$（）又は一方向性関数Ｆ$_{1}$（）の出力より計算できる値であり、（＋）は排他的論理和である。

【００３０】
実施形態によっては、上記部分Ｗ又は上記パスワード情報ｖが、上記第１識別情報（Ｃ

，ＷＩＤ）に関連づけられて上記サーバの記憶装置に予め格納されており、上記サーバが、上記受信した第１識別情報に基づいて該記憶装置から上記部分Ｗ（上記部分Ｗが格納されている場合）又は、上記パスワード情報ｖ（上記パスワード情報ｖが格納されている場合）を検索するステップを有することができる。

【００３１】

実施形態によっては、上記部分Ｗ又は上記パスワード情報ｖが、上記第１識別情報（Ｃ，ＷＩＤ）を少なくとも入力に含む一方向性関数H_3（）の出力から計算される第２識別情報に関連づけられて上記サーバの記憶装置に予め格納されており、上記サーバが、上記受信した第１識別情報（Ｃ，ＷＩＤ）を少なくとも入力に含む一方向性関数H_3（）の出力から計算した上記第２識別情報により該記憶装置から上記部分Ｗ（上記部分Ｗが格納されている場合）又は、上記パスワード情報ｖ（上記パスワード情報ｖが格納されている場合）を検索するステップを有することができる。

【００３２】

実施形態によっては、上記クライアントがクライアント側改ざん検出子生成鍵、クライアント側改ざん検出子検証鍵を持ち、上記サーバが、上記第１識別情報又は上記第２識別情報と共に、上記クライアント側改ざん検出子生成鍵により生成された改ざん検出子を検証するためのサーバ側改ざん検出子検証鍵と、上記クライアント側改ざん検出子検証鍵で検証できる改ざん検出子を生成できるサーバ側改ざん検出子生成鍵を持ち、

（ａ）上記クライアントが、少なくとも上記Ｕ（あるいは、上記Ｕと上記サーバから受信した乱数）に対して上記クライアント側改ざん検出子生成鍵を用いて改ざん検出子ｍａｃ１を生成し、それを上記サーバに送信するステップと、

（ｂ）上記サーバが、上記クライアントから受け取った改ざん検出子ｍａｃ１を上記サーバ側改ざん検出子検証鍵を用いて検証し、それが検証されなかった場合に、該クライアント・サーバ間の相互認証を中断するステップと、

（ｃ）同じく、上記サーバが、上記クライアントから受け取った改ざん検出子ｍａｃ１を上記サーバ側改ざん検出子検証鍵を用いて検証し、それが検証された場合に、少なくとも上記Ｕと上記Ｙの組をログリストＰｓ’に記録すると共に、少なくとも上記Ｕと上記Ｙに対して上記サーバ側改ざん検出子生成鍵を用いて改ざん検出子ｍａｃ２を生成し、それを上記クライアントに送信するステップと、

（ｄ）上記クライアントが、該サーバから送られてきた改ざん検出子ｍａｃ２の検証に失敗した場合に、上記クライアント認証情報Ｖｃを送信せず該クライアント・サーバ間の相互認証を中断するステップと、

（ｅ）同じく上記クライアントが、該サーバから送られてきた改ざん検出子ｍａｃ２の検証に成功した場合には、少なくとも上記Ｙと上記クライアント認証情報Ｖｃに対して該クライアント側改ざん検出子生成鍵を用いて改ざん検出子ｍａｃ３を生成し、その改ざん検出子を該サーバに送信すると共に、少なくとも上記Ｕと上記Ｙの組をログリストＰｃに記録するステップと、

（ｆ）該サーバが、該クライアントから送られてきた改ざん検出子ｍａｃ３の検証に失敗した場合に、該クライアント・サーバ間の相互認証を中断するステップと、

（ｇ）同じく該サーバが、該クライアントから送られてきた改ざん検出子ｍａｃ３の検証に成功し、かつ、該クライアントから送られてきた上記クライアント認証情報Ｖｃの検証に失敗した場合に、少なくとも上記Ｕと上記Ｙの組をログリストＰｓに記録し、該クライアント・サーバ間の相互認証を中断するステップと、

（ｈ）該クライアント・サーバ間の相互認証が正常に終了した際には、前回該クライアント・サーバ間の相互認証ステップが正常に終了した以降に該サーバと該クライアントが該クライアント・サーバ間の相互認証処理中に記録し続けたログリストＰｓ、Ｐｃ、Ｐｓ’中の少なくともＵとＹの組を第三者に改ざんされない方法で比較するステップと、

を有することができる。

【００３３】

この実施形態では、Ｐｃ中のＵとＹの組からＰｓ’中のＵとＹの組に一致するエントリ

を取り除いた残りエントリ数がサーバ側からのオンライン全数探索の数とみなし、Ｐｓ中のＵとＹの組からＰｃ中のＵとＹの組に一致するエントリを取り除いた残りエントリ数がクライアント側からのオンライン全数探索の数とみなすことができる。これによって、改ざん検出を用いたパスワードのオンライン全数探索検知機能を提供することができる。

【００３４】

実施形態によっては、上記サーバ側及び上記クライアント側の少なくとも一方における上記改ざん検出子生成鍵及び上記検出子検証鍵をＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）鍵とすることができる。

【００３５】

実施形態によっては、上記サーバ側及び上記クライアント側の両方の改ざん検出子生成鍵及び検出子検証鍵をＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）鍵とすることができる。この場合、

上記サーバ側の改ざん検出子生成鍵、

上記サーバ側の改ざん検出子検証鍵、

上記クライアント側の改ざん検出子生成鍵、

上記クライアント側の改ざん検出子検証鍵、

は全て同じ鍵とし、

上記クライアントと上記サーバが同じＭＡＣ鍵を使っても異なる改ざん検出子を生成できるように、上記クライアントと上記サーバとで、異なるＭＡＣ生成アルゴリズム、又は、異なるメッセージフォーマットを用いることとすることができる。

【００３６】

実施形態によっては、上記改ざん検出子生成鍵を電子署名生成鍵、上記検出子検証鍵を電子署名検証鍵とすることができる。

【００３７】

実施形態によっては、上記クライアントがクライアント側データ鍵ｃｄｋを持ち、上記サーバがサーバ側データ鍵ｓｄｋを第１識別情報又は第２識別情報と共に持ち、上記クライアント・サーバ間の相互認証ステップが正常に終了した際に、

（ａ）該サーバが、該クライアント・サーバ間の相互認証ステップにより生成された上記サーバ側マスター秘密Ｋｓに依存して生成される暗号化鍵を使って暗号化されたサーバ側データ鍵ｓｄｋを該クライアントに送信するステップと、

（ｂ）該クライアントが、該サーバから送信された暗号化されたデータ鍵ｓｄｋを該クライアント・サーバ間の相互認証ステップにより生成された該クライアント側マスター秘密ＫＣに依存して生成される暗号化鍵を使って復号するステップと、

（ｃ）該クライアントが上記クライアント側データ鍵ｃｄｋと上記サーバ側データ鍵ｓｄｋからデータｄｋを復元するステップを持つ

こととすることができる。

【００３８】

実施形態によっては、上記クライアントがクライアント側データ鍵ｃｄｋを持ち、上記サーバがサーバ側データ鍵ｓｄｋを第１識別情報又は第２識別情報と共に持ち、上記クライアント・サーバ間の相互認証ステップが正常に終了した際に、

（ａ’）該クライアントが、該クライアント・サーバ間の相互認証ステップにより生成された上記クライアント側マスター秘密Ｋｃに依存して生成される暗号化鍵を使って暗号化されたクライアント側データ鍵ｃｄｋを該サーバに送信するステップと、

（ｂ’）該サーバが、該クライアントから送信された暗号化されたクライアント側データ鍵ｃｄｋを該クライアント・サーバ間の相互認証ステップにより生成された該サーバ側マスター秘密Ｋｓに依存して生成される暗号化鍵を使って復号するステップと、

（ｃ’）該サーバが上記クライアント側データ鍵ｃｄｋと上記サーバ側データ鍵ｓｄｋからデータｄｋを復元するステップを持つ

こととすることができる。

【００３９】

実施形態によっては、上記データｄｋは、
ｄｋ’＝ｃｄｋ（＋）ｓｄｋ　又は
ｄｋ’＝ｃｄｋ＋ｓｄｋ　ｍｏｄ　ｑ　又は
ｄｋ’＝ｃｄｋ×ｓｄｋ　ｍｏｄ　ｑ
とし、
ｄｋ＝ｄｋ’　又は
ｄｋ＝（ｄｋ’）（＋）ｈｐｗ　又は
ｄｋ＝（ｄｋ’）＋ｈｐｗ　ｍｏｄ　ｑ　又は
ｄｋ＝（ｄｋ’）×ｈｐｗ　ｍｏｄ　ｑ　又は
少なくともｄｋ’とｈｐｗを入力とする関数H_3（）の出力から計算できる値
により復元できることとすることができる。ここで、（ｄｋ’）はｄｋ’の値を最初に計算するというステップを示しており、ｈｐｗ　は、上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力より計算できる値であり、（＋）は排他的論理和である。

【００４０】

実施形態によっては、上記クライアント・サーバ間の相互認証のために、上記クライアントが利用する可能性のある情報である、上記乱数ｔ、上記第１識別情報の全てあるいは一部を、該クライアントの記録装置に予め保存し、該クライアントが上記クライアント・サーバ間の相互認証を実行する祭に、上記クライアント・サーバ間の相互認証の要求に応じて呼び出すこととすることができる。

【００４１】

実施形態によっては、上記クライアント・サーバ間の相互認証が成功した場合、
その相互認証で利用した該サーバの該記憶装置に記録されている情報である、上記第１識別情報、上記部分Ｗ又は上記パスワード情報ｖ、クライアントが生成した乱数ｔのコミット値Ｔ、の全てあるいは一部、並びに、
該クライアントの該記憶装置に記録されている情報である、上記乱数ｔ、上記第１識別情報、の全てあるいは一部を、
該クライアントと該サーバとの間でやりとりされた値、又は、該認証ステップで共有された上記マスター秘密ＫＣ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを有することとすることができる。

【００４２】

実施形態によっては、上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
「ｖ＝ｓ＋ｈｐｗ　ｍｏｄ　ｑ」または「ｖ＝ｓ×ｈｐｗ　ｍｏｄ　ｑ」
により計算することができ、
上記更新するステップにおいて、上記サーバは、
「Ｗ’＝Ｗ・g^{ud}」または「Ｗ’＝W^{ud}」
と表記されうるように上記部分ＷをＷ’に更新し、上記クライアントは、
「ｓ’＝ｓ＋ｕｄ　ｍｏｄ　ｑ」または「ｓ’＝ｓ×ｕｄ　ｍｏｄ　ｑ」
と表記されうるように上記乱数ｓをｓ’に更新することとすることができる。

【００４３】

実施形態によっては、上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
「ｖ＝ｓ＋ｈｐｗ　ｍｏｄ　ｑ」
により計算することができ、
上記更新するステップにおいて、上記サーバは、
「ｖ’＝ｖ＋ｕｄ　ｍｏｄ　ｑ」
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
「ｓ’＝ｓ＋ｕｄ　ｍｏｄ　ｑ」

と表記されうるように上記乱数ｓをｓ’に更新することとすることができる。
【００４４】
実施形態によっては、上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
ｖ＝ｓ（＋）ｈｐｗ
により計算することができ、
上記更新するステップにおいて、上記サーバは、
ｖ’＝ｖ（＋）ｕｄ
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
ｓ’＝ｓ（＋）ｕｄ
と表記されうるように上記乱数ｓをｓ’に更新することとすることができる。
【００４５】
実施形態によっては、上記パスワード情報ｖが、上記パスワードｐｗと乱数ｓに基づいて、次の計算式：
ｖ＝ｓ×ｈｐｗ　ｍｏｄ　ｑ
により計算することができ、
上記更新するステップにおいて、上記サーバは、
ｖ’＝ｖ×ｕｄ　ｍｏｄ　ｑ
と表記されうるように上記パスワード情報ｖをｖ’に更新し、上記クライアントは、
ｓ’＝ｓ×ｕｄ　ｍｏｄ　ｑ
と表記されうるように上記乱数ｓをｓ’に更新することとすることができる。なお、上記において、ｈｐｗ　は上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力であり、ｕｄ　は上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【００４６】
実施形態によっては、上記クライアント・サーバ間の相互認証が成功した場合、
その相互認証で利用した該サーバの該記憶装置に記録されている情報である、該サーバ側の改ざん検出子生成鍵、該サーバ側の改ざん検出子検証鍵の全てあるいは一部、及び、該クライアントの該記憶装置に記録されている情報である、上記クライアント側の改ざん検出子生成鍵、上記クライアント側の改ざん検出子検証鍵の全てあるいは一部を、該クライアントと該サーバとの間でやりとりされた値、又は、該認証処理で共有された上記マスター秘密Ｋｃ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを有し、
上記サーバと上記クライアントが同じ上記ＭＡＣ鍵ＭａｃＫを用いる場合、上記サーバと上記クライアントがそれぞれ上記ＭＡＣ鍵ＭａｃＫをＭａｃＫ’に更新するステップは、
ＭａｃＫ’＝ＭａｃＫ（＋）ｕｄ　又は
ＭａｃＫ’＝ＭａｃＫ＋ｕｄ　ｍｏｄ　ｑ　又は
ＭａｃＫ’＝ＭａｃＫ×ｕｄ　ｍｏｄ　ｑ　又は
と表記可能である。ただし、ｕｄ　は上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値である。
【００４７】
実施形態によっては、上記クライアント・サーバ間の相互認証が成功した場合、
その相互認証で利用した該サーバの該記憶装置に記録されている情報である上記サーバ側データ鍵ｓｄｋ及び、
該クライアントの該記憶装置に記録されている情報である上記クライアント側データ鍵ｃｄｋを、
該クライアントと該サーバとの間でやりとりされた値、又は、該認証処理で共有された上記マスター秘密Ｋｃ（クライアント側）Ｋｓ（サーバ側）、又は上記マスター秘密と上記該サーバと該クライアントの間でやりとりされた値の両方を使って更新するステップを

有し、
上記クライアントが上記クライアント側データ鍵ｃｄｋを持ち、上記サーバが上記サーバ側データ鍵ｓｄｋを持つ場合、上記サーバと上記クライアントがそれぞれｃｄｋとｓｄｋをｃｄｋ’とｓｄｋ’に更新するステップは、
上記ｄｋ’が
ｄｋ’＝ｃｄｋ（＋）ｓｄｋ
と表記可能な場合、
ｃｄｋ’＝ｃｄｋ（＋）ｕｄ
ｓｄｋ’＝ｓｄｋ（＋）ｕｄ
と表記可能であり、
上記ｄｋ’が
ｄｋ’＝ｃｄｋ＋ｓｄｋ　ｍｏｄ　ｑ
と表記可能な場合、
ｃｄｋ’＝ｃｄｋ＋ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ－ｕｄ　ｍｏｄ　ｑ
又は
ｃｄｋ’＝ｃｄｋ－ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ＋ｕｄ　ｍｏｄ　ｑ
と表記可能であり、
上記ｄｋ’が
ｄｋ’＝ｃｄｋ×ｓｄｋ　ｍｏｄ　ｑ
と表記可能な場合、
ｃｄｋ’＝ｃｄｋ×ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ／ｕｄ　ｍｏｄ　ｑ
又は
ｃｄｋ’＝ｃｄｋ／ｕｄ　ｍｏｄ　ｑ
ｓｄｋ’＝ｓｄｋ×ｕｄ　ｍｏｄ　ｑ
と表記可能である。ただし、ｕｄ　は上記サーバと上記クライアントが共有している上記マスター秘密Ｋｓ（サーバ側）Ｋｃ（クライアント側）から生成される値であり、－は整数上の引き算、「ａ＝ｃ／ｂ　ｍｏｄ　ｑ」はｃが０でない場合「ａ×ｂ≡ｃ　ｍｏｄ　ｑ」を満たす１以上ｑ未満の整数ａを示す。

【００４８】
本発明の実施形態は、クライアント装置とサーバ装置から構成されるシステムであって、該クライアント装置と該サーバ装置が上記の相互認証方法を実行するように構成される、システムを含む。

【００４９】
本発明の実施形態は、上記の相互認証方法における、クライアント側で実行される処理を遂行しうるように構成される、コンピュータ装置を含む。

【００５０】
本発明の実施形態は、上記の相互認証方法における、サーバ側で実行される処理を遂行しうるように構成される、コンピュータ装置を含む。

【００５１】
本発明の実施形態は、コンピュータ装置のＣＰＵで実行されることにより、該コンピュータ装置に、上記の相互認証方法における、クライアント側で実行される処理を遂行させる、コンピュータ・プログラムを含む。

【００５２】
本発明の実施形態は、コンピュータ装置のＣＰＵで実行されることにより、該コンピュータ装置に、上記の相互認証方法における、サーバ側で実行される処理を遂行させる、コンピュータ・プログラムを含む。

【００５３】

以下、本発明の理解に資するために、本発明の更なる実施形態の例をいくつか説明する。ただし、これらの実施例は、本発明の請求の範囲を限定する意図で説明するものではなく、あくまで本発明の理解を深めるために説明するものであることを留意されたい。

【００５４】

パスワードのみを用いた認証システム及びその応用システムの実施例を説明する前に、以下の説明において用いる背景知識と記号についてまず説明しておく。

【００５５】

以下の方法において位数ｑの群を（Ｇ，・）とし、その集合Ｇの生成元をｇ、Ｕ，Ｗ∈Ｇとする。また、”・”はＧ上の二項演算子であり$g_1, g_2 \in G$として、$g_1 \cdot g_2$を$g_1 g_2$と表記し、$g_1 \cdot g_1$をg_1^2と表記し、$g_1^i \cdot g_1^j$をg_1^{i+j}と表記している。本発明は、離散対数問題を解くことが困難な様々な群を用いて実施可能であるため、以下の説明および請求の範囲において、素体の群及びある種の楕円曲線群に限定されないことに留意してほしい。

【００５６】

ハッシュ関数Ｈのセキュリティパラメータをｋとする。ただし、$1/2^k$は無視できるほど小さいと仮定する。また、$\{0,1\}^*$は有限の２進数のストリングの集合を、$\{0,1\}^k$は長さｋの２進数のストリングの集合を表す。ハッシュ関数Ｈは$\{0,1\}^*$の入力から$\{0,1\}^k$の出力を出す安全な一方向関数であり、ＦＤＨ（Ｆｕｌｌ－Ｄｏｍａｉｎ　Ｈａｓｈ）関数H_1とH_2は$\{0,1\}^*$の入力から$(Z/qZ)^*$の出力を出す安全な一方向関数である。ここで、$(Z/qZ)^*$は$\{1,2,\cdots,q\}$の集合（部分群）を表す。また、乱数発生器から発生される乱数は$R \in (Z/qZ)^*$を無作為に生成する。また、‖は値を連結（ｃｏｎｃａｔｅｎａｔｉｏｎ）するという意味である。また、ＣとＳ（あるいはS_i）はそれぞれユーザとサーバ（あるいは、多数のサーバの中でｉ番目のサーバ）を表すＩＤである。

【００５７】

〔実施例１：パスワードのみを用いた認証システム〕

図１は、実施例１として以下に説明する、パスワードのみを用いて認証を行う認証システム１００の全体構成を説明するための図である。認証システム１００は、ユーザの端末装置３００とサーバの認証装置４００とで構成される。ユーザの端末装置３００は、ユーザから入力されたパスワードに基づいて、ある特徴的な演算を行う。サーバの認証装置４００は、そのデータベース４０２にユーザのＩＤとパスワード認証データとを保持している。認証システム１００において、ユーザの端末装置３００は、サーバの認証装置４００とインターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した場合のみ、お互いに同じセッション鍵を確保する。共有されたセッション鍵はユーザの端末装置３００とサーバの認証装置４００が後で行う通信内容を保護するために使われる。

【００５８】

なお、以下の全ての図において、そこに描かれる機能要素のそれぞれは、専用のハードウェアによって実現されることもできるが、ＣＰＵとコンピュータ・プログラムを用いたソフトウェア処理によって実現されることができる場合がある。従って、以下の全ての図において、例えば「乱数発生器」のように「器」「装置」のような用語が用いられているとしても、その実現手段はハードウェアに限定されるものではなく、ソフトウェア処理による手段によっても実装可能であることに注意されたい。また、２つ以上の機能要素を１つのハードウェア回路にまとめたり、あるいは２つ以上の機能要素をそれぞれサブプログラムとして含んだ１つのプログラムにまとめたりすることも可能である。例えば、以下の実施例で紹介されるユーザの端末装置やサーバの認証装置の機能の全部又は一部を、プロセッサとメモリとプログラムコードを用いて実現することも可能である。また、各機能要素をＦＰＧＡのようなプログラマブルな回路を用いて実現することも可能である。当業者であれば、当然ながら、実施形態の具体的要求に応じて、適切な実装手段を選択することができるだろう。

【００５９】
［１．パスワードのみを用いた認証システム１００の初期化］
次に、図２を参照して、認証システム１００における初期化処理について説明する。この初期化処理において、ユーザの端末装置３００は、ユーザから入力されたパスワードに基づいて、サーバの認証装置４００と安全な通信路（例えば、直接に登録したり、郵便で送付したり、あるいは電話で知らせるなど）を用いて初期化処理を行い、サーバの認証装置４００は内部にあるメモリあるいはデータベース４０２へユーザのＩＤとパスワード認証データを保存する。
【００６０】
＜ユーザの端末装置３００における作業＞
図２に描かれるように、ユーザの端末装置３００は、パスワード認証データ生成器３０２を有している。端末装置３００における初期化処理において、パスワード認証データ生成器３０２は、ユーザから入力されたパスワードｐｗを入力として、パスワード認証データWを計算式：
$W = g^{H1(C \| S \| pw)}$
により計算して出力する。その後端末装置３００は、ユーザのＩＤとパスワード認証データ「Ｃ，Ｗ」とをサーバの認証装置４００へ送信する。
【００６１】
＜サーバの認証装置４００における作業＞
図２に示すように、サーバの認証装置４００における初期化処理において、認証装置４００は、ユーザの端末装置３００から受信したユーザのＩＤとパスワード認証データ「Ｃ，Ｗ」を、認証装置４００の内部にあるメモリあるいはデータベース４０２へ格納する。
【００６２】
［２．パスワードのみを用いた認証システム１００のプロトコル実行］
次に、図３および図４を参照して、パスワードのみを用いた認証システム１００におけるプロトコル実行処理について説明する。このプロトコル実行処理は、図２を用いて説明した初期化処理が完了した後に行われる。当該プロトコル実行処理において、ユーザの端末装置３００は、ユーザから入力されたパスワードに基づいてある特徴的な演算を行い、サーバの認証装置４００とインターネットのようなパブリックネットワークを通じてお互いに相互認証する。前述のようにサーバの認証装置４００は、ユーザのＩＤ及びパスワード認証データをデータベース４０２などに保持している。ユーザの端末装置３００とサーバの認証装置４００は、相互認証が成功した時のみ、互いに同じセッション鍵を確保する。
【００６３】
＜ユーザの端末装置３００の動作＞
図３は、認証システム１００のプロトコル実行を行うユーザの端末装置３００の機能構成及び動作を説明するためのブロック図である。まず、この図を参照して、端末装置３００における認証システム１００のプロトコル実行処理について説明する。
【００６４】
公開値演算器３０４は、乱数発生器３０６によりランダムに発生させた乱数ｕ（$u \in (Z/qZ)^*$）を入力として、公開値Uを計算式：
$U = g^u$
により計算して出力する。ブラインド生成器３０８は、ユーザから入力されたパスワードｐｗと乱数発生器３０６によりランダムに発生させた乱数ｕと公開値演算器３０４により出力された公開値Ｕとを入力として、ブラインドｂを計算式：
$b = u + H_1(C \| S \| pw) \times r \mod q$
により計算して出力する。ここで、
$r = H_2(C \| S \| U)$
である。逆ブラインド演算器３１０は、ブラインド生成器３０８により出力されたブラインドｂを入力として、逆ブラインドb^{-1}を計算式：

$b^{-1} \mod q$
により計算して出力する。ユーザの端末装置３００は、図示しない通信処理部を介して、サーバの認証装置４００に対して、ユーザのＩＤであるＣと公開値演算器３０４により出力された公開値Ｕを送信する。

【００６５】

しばらくすると、ユーザの端末装置３００は、サーバの認証装置４００からメッセージ「Ｓ，Ｙ」を受信する。マスター秘密生成器３１２は、サーバの認証装置４００から受信した値Ｙと、逆ブラインド演算器３１０により出力された逆ブラインドb^{-1}を入力として、マスター秘密Ｋを計算式：
$K = Y^{(1/b \mod q)}$
により計算して出力する。

【００６６】

続いて、認証子生成器３１４は、公開値演算器３０４により出力された公開値Ｕとサーバの認証装置４００から受信した値Ｙとマスター秘密生成器３１２により出力されたマスター秘密Ｋを入力として、認証子Ｖｃを計算式：
Ｖｃ＝Ｈ（１‖Ｃ‖Ｓ‖Ｕ‖Ｙ‖Ｋ）
により計算、入力メッセージＩｎｐｕｔＭｓｇとともに出力する。ここで、入力メッセージは
ＩｎｐｕｔＭｓｇ＝Ｃ‖Ｓ‖Ｕ‖Ｙ‖Ｋ
である。ここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）を使ってもよい。ユーザの端末装置３００は、図示しない通信処理部を介して、認証子生成器３１４により出力された認証子Ｖｃをサーバの認証装置４００へ送信する。

【００６７】

しばらくすると、ユーザの端末装置３００は、サーバの認証装置４００からメッセージ「Ｖｓ」を受信する。認証子判断部３１６は、サーバの認証装置４００から受信した認証子Ｖｓが正しく生成された値であるかどうかを確認する。認証子判断部３１６は、認証子生成器３１４から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいてハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）を計算し、サーバの認証装置４００から受信した認証子Ｖｓと比較する。認証子判断部３１６の判断処理において、認証子Ｖｓとハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部３１６は、エラーメッセージ発生器３１８に対して一致しないことを通知する。これを受けて、エラーメッセージ発生器３１８はエラーメッセージを生成して処理を中断する。一方、認証子判断部３１６は、認証子Ｖｓとハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致しているとを判断した場合は、サーバの認証装置４００は正当な装置であるとして認証し、セッション鍵生成器３２０へその旨を通知する。セッション鍵生成器３２０は、認証子生成器３１４から供給されるメッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵ＳＫを計算式：
ＳＫ＝Ｈ（３‖ＩｎｐｕｔＭｓｇ）
により計算して出力する。

【００６８】

＜サーバの認証装置４００の動作＞

図４は、認証システム１００のプロトコル実行を行うサーバの認証装置４００の機能構成及び動作を説明するためのブロック図である。次に、この図を参照して、サーバの認証装置４００における認証システム１００のプロトコル実行処理について説明する。

【００６９】

サーバの認証装置４００は、プロトコル実行のためのユーザのＩＤとパスワード認証データ「Ｃ，Ｗ」を格納装置の内部にあるメモリあるいはデータベース４０２に保持している。

【００７０】

まず、マスター秘密生成器４１２は、乱数発生器４０６によりランダムに発生させた乱

数ｙ（ｙ∈（Ｚ／ｑＺ）＊）を入力として、マスター秘密Ｋを計算式：
$K = g^{y}$
により計算して出力する。
【００７１】
しばらくすると、サーバの認証装置４００は、ユーザの端末装置３００からメッセージ「Ｃ，Ｕ」を受信する。第１ブラインド生成器４０４は、端末装置３００から受信した公開値Ｕとデータベース４０２から読み出したパスワード認証データＷと乱数発生器４０６によりランダムに発生させた乱数ｙとを入力として、第１ブラインド$W^{y \cdot r}$を計算式：
$W^{y \cdot r}$
により計算して出力する。ここで、
$r = H_2(C \| S \| U)$
である。別の方法としてｒをサーバが生成してからクライアントに送ってもよい。第２ブラインド生成器４０５は、ユーザの端末装置３００から受信した公開値Ｕと乱数発生器４０６によりランダムに発生させた乱数ｙとを入力として、第２ブラインドU^{y}を計算式：
U^{y}
により計算して出力する。
【００７２】
マスク演算器４０８は、第１ブラインド生成器４０４により出力された第１ブラインド$W^{y \cdot r}$と第２ブラインド生成器４０５により出力された第２ブラインドU^{y}を入力として、値Ｙを
$Y = U^{y} \cdot W^{y \cdot r}$
により計算して出力する。サーバの認証装置４００は、図示しない通信処理部を介して、ユーザの端末装置３００に対して、サーバのＩＤであるＳとマスク演算器４０８により出力された値Ｙを送信する。
【００７３】
続いて、入力メッセージ集合器４１４は、ユーザの端末装置３００から受信した公開値Ｕとマスク演算器４０８により出力された値Ｙとマスター秘密生成器４１２により出力されたマスター秘密Ｋとを入力として、入力メッセージＩｎｐｕｔＭｓｇを出力する。ここで、入力メッセージは
$InputMsg = C \| S \| U \| Y \| K$
である。
【００７４】
しばらくすると、サーバの認証装置４００は、ユーザの端末装置３００からメッセージ「Ｖｃ」を受信する。認証子判断部４１６は、ユーザの端末装置３００から受信した認証子Ｖｃが正しく生成された値であるかどうかを確認する。認証子判断部４１６は、入力メッセージ集合器４１４から供給された入力メッセージＩｎｐｕｔＭｓｇに基づいてハッシュ関数Ｈ（１∥ＩｎｐｕｔＭｓｇ）を計算し、ユーザの端末装置３００から受信した認証子Ｖｃと比較する。認証子判断部４１６の判断処理において、認証子Ｖｃとハッシュ関数Ｈ（１∥ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部４１６は、エラーメッセージ発生器４１８に対して一致しないことを通知する。これを受けて、エラーメッセージ発生器４１８はエラーメッセージを生成して処理を中断する。
【００７５】
一方、認証子判断部４１６の判断処理において、認証子Ｖｃとハッシュ関数Ｈ（１∥ＩｎｐｕｔＭｓｇ）が一致したと判断された場合は、ユーザの端末装置３００が正当な装置として認証される。ここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）を使ってもよい。認証子生成器４２０は、入力メッセージ集合器４１４から供給された入力メッセージＩｎｐｕｔＭｓｇを入力として、認証子Ｖｓを計算式：
Ｖｓ＝Ｈ（２∥ＩｎｐｕｔＭｓｇ）
により計算して出力する。サーバの認証装置４００は、図示しない通信処理部を介して、

ユーザの端末装置３００に対して、認証子生成器４２０により出力された認証子Ｖｓを送信する。セッション鍵生成器４２２は、入力メッセージ集合器４１４から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵ＳＫを計算式：
ＳＫ＝Ｈ（３‖ＩｎｐｕｔＭｓｇ）
により計算して出力する。

【００７６】

［３．認証システム１００の変形例］

認証システム１００において、サーバの認証装置４００の第１ブラインド生成器４０４及び第２ブラインド生成器４０５は、第１ブラインド及び第２ブラインドを$W^{y \cdot r}$，U^{y}によって計算していた。しかしながら、これらは次のように計算してもよい。

第１ブラインド：W^{y}
第２ブラインド：$U^{y \cdot r}$　（ただしｒ＝H_2（Ｃ‖Ｓ‖Ｕ））
すなわち、はじめの例ではｒがパスワード認証データＷに関するべき乗計算に用いられていたのに対し、この変形例ではｒが公開値Ｕに関するべき乗計算に用いられている。

【００７７】

第１ブラインド及び第２ブラインドの計算方法の変形に伴い、マスク演算器４０８で計算される値Ｙの計算も、次のように変形される。
$Y = U^{y \cdot r} \cdot W^{y}$

【００７８】

かかる変形例の場合、ユーザの端末装置３００のブラインド生成器３０８におけるブラインドｂの計算式も、次のように変形される。
ｂ＝ｕ×ｒ＋H_1（Ｃ‖Ｓ‖ｐｗ）　ｍｏｄ　ｑ（ただしｒ＝H_2（Ｃ‖Ｓ‖Ｕ））
すなわち、はじめの例ではｒがパスワードｐｗに関する項に乗じられていたのに対し、この変形例ではｒが乱数ｕに乗じられている。

【００７９】

実施例１に係る認証システム１００では、サーバにおけるマスター秘密の計算を、式：
$K = g^{y}$
で行うことができるため、g^{uy}のべき乗計算が必要な従来技術よりも、サーバの計算量を大きく抑えることができる。また、クライアントにおけるマスター秘密の計算も、式：
$K = Y^{(1/b \bmod q)}$
で行うことができるため、g^{uy}のべき乗計算が必要な従来技術よりも、やはり計算量を抑えることができる。これらの利点は上述の変形例においても失われない。パスワード及びパスワード認証データＷを用いたことによる安全性と相まって、認証システム１００は、ＫＣＩ攻撃などパブリックネットワーク上の様々な攻撃に対して安全であり、且つクライアント及びサーバの計算量を従来方式に比べて減少させることに成功している。

【００８０】

〔実施例２：実施例１のシステムの応用〕

次に、実施例１で紹介したパスワードのみを用いた認証方式を、二要素認証方式に応用した例を説明する。図５は、この応用例を説明するために例示として用いる認証システム５００の全体構成を描いている。

【００８１】

認証システム５００は、ユーザの端末装置７００とサーバの認証装置８００とで構成される。認証システム５００において、ユーザの端末装置７００はユーザから入力されたパスワードに加えてメモリ７０２などに保持されている記録情報に基づき、ある特徴的な演算を行う。サーバの認証装置８００は、識別子ＷＩＤと認証データをデータベース８０２などに保持している。識別子ＷＩＤは、例えば端末装置７００の識別子とカウント値とを含むものであることができる。端末装置７００とサーバの認証装置８００はインターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保すると共に、各自の記録情報を次のセッションのために更新していく。それにより、サーバの認証装置８００は、ユーザの認証データ

に対してパスワードを全数探索することができなくなり、ユーザの端末装置７００は記録情報が漏洩したとしても安全性が落ちることがない。

【００８２】

［１．その認証システム５００の初期化］

初めに、図６を参照して、認証システム５００における初期化処理について説明する。図６は、認証システム５００の初期化処理に係る、ユーザの端末装置７００とサーバの認証装置８００の機能構成及び動作を説明するためのブロック図である。認証システム５００の初期化処理において、ユーザの端末装置７００は、ユーザから入力されたパスワードに基づいて、サーバの認証装置８００と安全な通信路（例えば、直接に登録したり、郵便で送付したり、あるいは電話で知らせるなど）を用いて初期化処理を行い、ユーザの端末装置７００は内部にあるメモリ７０２へ記録情報としてＣＳ１を保存し、サーバの認証装置８００は内部にあるメモリあるいはデータベース８０２へ記録情報としてＳＳ１を保存する。

【００８３】

＜ユーザの端末装置７００における作業＞

図６に示すように、ユーザの端末装置７００における初期化処理では、結合器７０４は、ユーザから入力されたパスワードｐｗと乱数発生器７０６によりランダムに発生させた乱数s_{i1}（$s_{i1}\in(Z/qZ)^*$）を入力として、結合値v_{i1}を計算式：

$$v_{i1}=s_{i1}+H_1(C\|S_i\|pw) \mod q$$

により計算して出力する。ここで、S_iはｉ番目のサーバを表す。二要素認証データ生成器７０８は、結合器７０４により出力された結合値v_{i1}を入力として、認証データW_{i1}を計算式：

$$W_{i1}=g^{v_{i1}}$$

により計算して出力し、識別子ＷＩＤと認証データ「ＷＩＤ，W_{i1}」をサーバの認証装置８００へ送信する。ユーザの端末装置７００は、サーバのＩＤと識別子ＷＩＤと乱数発生器７０６により発生させた乱数s_{i1}を、ユーザの端末装置７００の内部にあるメモリ７０２へ記録情報「ＣＳ１」として「S_i，ＷＩＤ，s_{i1}」を保存する。

【００８４】

＜サーバの認証装置８００における作業＞

図６に示すように、サーバの認証装置８００における初期化処理において、サーバの認証装置８００は、ユーザの端末装置７００から受信した識別子ＷＩＤと認証データ「ＷＩＤ，W_{i1}」を、サーバの認証装置８００の内部にあるメモリあるいはデータベース８０２へ記録情報「ＳＳ１」として「ＷＩＤ，W_{i1}」を保存する。

【００８５】

［２．認証システム５００のｊ番目プロトコル実行］

次に、図７および図８を参照して、認証システム５００におけるｊ番目プロトコル実行処理について説明する。図７および図８は、認証システム５００のｊ番目プロトコル実行を行う、ユーザの端末装置７００とサーバの認証装置８００の機能構成及び動作を、それぞれ説明するためのブロック図である。図７および図８において、ｊはｊ≧１になるような整数である。認証システム５００のｊ番目プロトコル実行において、ユーザの端末装置７００は、ユーザから入力されたパスワードｐｗとメモリ７０２に保持している記録情報ＣＳｊとに基づいて、ある特徴的な演算を行う。そして、記録情報ＳＳｊをデータベースなどに保持しているサーバの認証装置８００と、インターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保する。以下に説明されるｊ番目プロトコル実行は、認証システム５００の初期化が完了した後（ｊ＝１の時）、あるいは認証システム５００のｊ－１番目プロトコル実行が終了した後（すなわち、ＣＳｊ＝（S_i，ＷＩＤ，s_{ij}）とＳＳｊ＝（ＷＩＤ，W_{ij}）の時）に、ユーザの端末装置７００及びサーバの認証装置８００によって実行される。

【００８６】

＜ユーザの端末装置７００における作業＞
まず、図７を参照して、ユーザの端末装置７００における認証システム５００のｊ番目プロトコル実行処理について説明する。前述した認証システム５００の初期化処理の後、認証システム５００のｊ番目プロトコル実行の前には、ユーザの端末装置７００は、記録情報「ＣＳｊ」として「S_i，ＷＩＤ，s_{ij}」を内蔵するメモリ７０２に保持している。

【００８７】
結合器７０４は、ユーザから入力されたパスワードｐｗとメモリ７０２から読み出した乱数s_{ij}とを入力として、結合値v_{ij}を計算式：
$$v_{ij}=s_{ij}+H_1(C \| S_i \| pw) \mod q$$
により計算して出力する。公開値演算器７１２は、乱数発生器７０６によりランダムに発生させた乱数ｕ（ｕ∈（Ｚ／ｑＺ）＊）を入力として、公開値Ｕを計算式：
$$U=g^u$$
により計算して出力する。

【００８８】
ブラインド生成器７１４は、結合器７０４により出力された結合値v_{ij}と乱数発生器７０６によりランダムに発生させた乱数ｕと、公開値演算器７１２により出力された公開値Ｕとを入力として、ブラインドｂを計算式：
$$b=u+v_{ij}\times r \mod q$$
により計算して出力する。ここで、ｒは：
$$r=H_2(C \| S_i \| U)$$
である。逆ブラインド演算器７１６は、ブラインド生成器７１４により出力されたブラインドｂを入力として、逆ブラインドb^{-1}を計算式：
$$b^{-1} \mod q$$
により計算して出力する。

【００８９】
ユーザの端末装置７００は、サーバの認証装置８００に対して、メモリ７０２から読み出した識別子ＷＩＤと公開値演算器７１２により出力された公開値Ｕを、図示しない通信処理部を介して送信する。

【００９０】
しばらくすると、ユーザの端末装置７００は、サーバの認証装置８００からメッセージ「S_i，Ｙ」を受信する。マスター秘密生成器７１８は、サーバの認証装置８００から受信した値Ｙと、逆ブラインド演算器７１６により出力された逆ブラインドb^{-1}を入力として、マスター秘密Ｋを計算式：
$$K=Y^{(1/b \mod q)}$$
により計算して出力する。

【００９１】
続いて、認証子生成器７２０は、メモリ７０２から読み出した識別子ＷＩＤと公開値演算器７１２により出力された公開値Ｕとサーバの認証装置８００から受信した値Ｙとマスター秘密生成器７１８により出力されたマスター秘密Ｋとを入力として、認証子Ｖｃを計算式：
$$Vc=H(1 \| WID \| S_i \| U \| Y \| K)$$
により計算して入力メッセージＩｎｐｕｔＭｓｇとともに出力する。ここで、入力メッセージは
$$InputMsg=WID \| S_i \| U \| Y \| K$$
である。ここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）を使ってもよい。ユーザの端末装置７００は、図示しない通信処理部を介して、認証子生成器７２０により出力された認証子Ｖｃをサーバの認証装置８００へ送信する。

【００９２】

しばらくすると、ユーザの端末装置７００は、サーバの認証装置８００からメッセージ「V_{Si}」を受信する。認証子判断部７２２は、サーバの認証装置８００から受信した認証子V_{Si}が正しく生成された値であるかどうかを確認する。認証子判断部７２２は、認証子生成器７２０から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、サーバの認証装置８００から受信した認証子V_{Si}と比較する。この比較処理において、認証子V_{Si}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部７２２は、エラーメッセージ発生器７２４に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器７２４はエラーメッセージを生成して処理を中断する。

【００９３】

一方、認証子判断の判断処理において、認証子V_{Si}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致したことを判断した場合は、サーバの認証装置８００が正当な装置として認証して、続けて次の処理を行う。セッション鍵生成器７２６は、認証子生成器７２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵SK_{ij}を計算式：

$SK_{ij}=H$（３‖ＩｎｐｕｔＭｓｇ）

により計算して出力する。秘密値更新器７２８は、メモリ７０２から読み出した識別子ＷＩＤと乱数s_{ij}と認証子生成器７２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、ｊ＋１番目の識別子ＷＩＤと乱数$s_{i(j+1)}$を計算式：

$WID=H_1$（ＷＩＤ‖ＩｎｐｕｔＭｓｇ）

$s_{i(j+1)}=s_{ij}+H_1$（ＩｎｐｕｔＭｓｇ） ｍｏｄ ｑ

により計算して出力する。ユーザの端末装置７００は、メモリ７０２に保持している現在の識別子ＷＩＤと乱数との組である「ＷＩＤ，s_{ij}」を、秘密値更新器７２８により出力された次の識別子ＷＩＤと乱数の組である「ＷＩＤ，$s_{i(j+1)}$」に書き換える。

【００９４】

生成したセッション鍵ＳＫは、ユーザの端末装置７００がクライアント側のデータ鍵ｃｄｋを暗号化してサーバの認証装置８００へ送信するために用いることができる。また、サーバの認証装置８００がユーザの端末装置７００へ暗号化して送信してくる、サーバの認証装置８００が暗号化されたサーバ側のデータ鍵ｓｄｋを復号するためにも用いることができる。

【００９５】

セッション鍵ＳＫが生成されると、ユーザの端末装置７００は、データ鍵ｃｄｋをセッション鍵ＳＫを用いて暗号化し、サーバの認証装置８００へ送信する。或いは、サーバの認証装置８００は、認証装置８００が生成したセッション鍵を用いてサーバ側データ鍵ｓｄｋを暗号化し、ユーザの端末装置７００へ送信する。ユーザの端末装置７００は、自身で生成したセッション鍵ＳＫを用いて受信したサーバ側データ鍵ｓｄｋを復号する。成功すると、ユーザの端末装置７００は、データ鍵ｄｋをｃｄｋとｓｄｋとを用いて復元することができる。

【００９６】

上記データｄｋは、

ｄｋ’＝ｃｄｋ（＋）ｓｄｋ 又は

ｄｋ’＝ｃｄｋ＋ｓｄｋ ｍｏｄ ｑ 又は

ｄｋ’＝ｃｄｋ×ｓｄｋ ｍｏｄ ｑ

とし、

ｄｋ＝ｄｋ’ 又は

ｄｋ＝（ｄｋ’）（＋）ｈｐｗ 又は

ｄｋ＝（ｄｋ’）＋ｈｐｗ ｍｏｄ ｑ 又は

ｄｋ＝（ｄｋ’）×ｈｐｗ ｍｏｄ ｑ 又は

少なくともｄｋ’とｈｐｗを入力とする関数H_3（）の出力から計算できる値

により復元できる。

ここで、（ｄｋ’）はｄｋ’の値を最初に計算するというステップを示しており、

ｈｐｗは、上記パスワードｐｗを少なくとも入力とする関数H_1（）又は一方向性関数F_1（）の出力より計算できる値であり、（＋）は排他的論理和である。

【００９７】

＜サーバの認証装置８００における作業＞

次に、図８を参照して、サーバの認証装置８００における認証システム５００のｊ番目プロトコル実行処理について説明する。前述した認証システム５００の初期化処理の後、認証システム５００のｊ番目プロトコル実行の前には、サーバの認証装置８００は、記録情報「ＳＳｊ」として「ＷＩＤ，W_{ij}」を格納装置の内部にあるメモリあるいはデータベース８０２に保持している。

【００９８】

まず、サーバの認証装置８００は、ユーザの端末装置７００からメッセージ「ＷＩＤ，Ｕ」を受信する。ＷＩＤ判断部８０４は、ユーザの端末装置７００から受信した識別子ＷＩＤの正確性を確認する。ＷＩＤ判断部８０４は、データベースから読み出した識別子ＷＩＤを、ユーザの端末装置７００から受信した識別子ＷＩＤと比較して一致しない場合、エラーメッセージ発生器８０６に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器８０６はエラーメッセージを生成して処理を中断する。一方、ＷＩＤ判断部８０４の判断処理において、データベースから読み出した識別子ＷＩＤをユーザの端末装置７００から受信した識別子ＷＩＤと比較して、一致した場合には、続けて次の処理を行う。

【００９９】

マスター秘密生成器８０８は、乱数発生器８１０によりランダムに発生させた乱数ｙ（$y \in (Z/qZ)^*$）を入力として、マスター秘密Ｋを計算式：

$$K = g^y$$

により計算して出力する。

【０１００】

第１ブラインド生成器８１２は、ユーザの端末装置７００から受信した公開値Ｕとデータベースから読み出した認証データW_{ij}と乱数発生器８１０によりランダムに発生させた乱数ｙとを入力として、第１ブラインド$W_{ij}^{y \cdot r}$を計算式：

$$W_{ij}^{y \cdot r}$$

により計算して出力する。ここで、ｒは

$$r = H_2(C \| S_i \| U)$$

である。第２ブラインド生成器８１４は、ユーザの端末装置７００から受信した公開値Ｕと乱数発生器８１０によりランダムに発生させた乱数ｙとを入力として、第２ブラインドU^yを計算式：

$$U^y$$

により計算して出力する。マスク演算器８１６は、第１ブラインド生成器８１２により出力された第１ブラインド$W_{ij}^{y \cdot r}$と第２ブラインド生成器８１４により出力された第２ブラインドU^yとを入力として、値Ｙを

$$Y = U^y \cdot W_{ij}^{y \cdot r}$$

により計算して出力する。サーバの認証装置８００は、図示しない通信処理部を介して、ユーザの端末装置７００に対して、サーバのＩＤであるＳｉとマスク演算器８１６により出力された値Ｙを送信する。

【０１０１】

続いて、入力メッセージ集合器８１８が、ユーザの端末装置７００から受信した識別子ＷＩＤと公開値Ｕとマスク演算器８１６により出力された値Ｙとマスター秘密生成器８０８により出力されたマスター秘密Ｋとを入力として、入力メッセージＩｎｐｕｔＭｓｇを出力する。ここで、入力メッセージは

$$InputMsg = WID \| Si \| U \| Y \| K$$

である。しばらくすると、サーバの認証装置８００は、ユーザの端末装置７００からメッセージ「Ｖｃ」を受信する。

【０１０２】
認証子判断部８２０は、ユーザの端末装置７００から受信した認証子Ｖｃが正しく生成された値であるかどうかを確認する。認証子判断部８２０は、入力メッセージ集合器８１８から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、ユーザの端末装置７００から受信した認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）との比較処理を行う。この比較処理において、認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部８２０は、エラーメッセージ発生器８２２に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器８２２はエラーメッセージを生成して処理を中断する。一方、上記比較処理において、受信した認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致すると判断された場合は、ユーザの端末装置７００が正当な装置として認証されるので、続けて次の処理が行われる。なおここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）を使ってもよい。

【０１０３】
認証子生成器８２４は、入力メッセージ集合器８１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、認証子$Ｖ_{Ｓｉ}$を計算式：
$Ｖ_{Ｓｉ}＝Ｈ（２‖ＩｎｐｕｔＭｓｇ）$
により計算して出力する。サーバの認証装置８００は、図示しない通信処理部を介して、ユーザの端末装置７００に対して、認証子生成器８２４により出力された認証子$Ｖ_{Ｓｉ}$を送信する。

【０１０４】
セッション鍵生成器８２６は、入力メッセージ集合器８１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵$ＳＫ_{ｉｊ}$を計算式：
$ＳＫ_{ｉｊ}＝Ｈ（３‖ＩｎｐｕｔＭｓｇ）$
により計算して出力する。秘密値更新器８２８は、データベース８０２から読み出した識別子ＷＩＤと認証データ$Ｗ_{ｉｊ}$と入力メッセージ集合器８１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、ｊ＋１番目の識別子ＷＩＤと認証データ$Ｗ_{ｉ（ｊ＋１）}$を計算式：
$ＷＩＤ＝Ｈ_{１}（ＷＩＤ‖ＩｎｐｕｔＭｓｇ）$
$Ｗ_{ｉ（ｊ＋１）}＝Ｗ_{ｉｊ}・ｇ^{Ｈ１（ＩｎｐｕｔＭｓｇ）}$
により計算して出力する。サーバの認証装置８００は、データベース８０２に保持している現在の識別子ＷＩＤと認証データの組である「ＷＩＤ，$Ｗ_{ｉｊ}$」を、秘密値更新器８２８により出力された次の識別子ＷＩＤと認証データの組である「ＷＩＤ，$Ｗ_{ｉ（ｊ＋１）}$」に書き換える。

【０１０５】
生成したセッション鍵ＳＫは、サーバの認証装置８００がサーバ側のデータ鍵ｓｄｋを暗号化してユーザの端末装置７００へ送信するために用いることができる。また、ユーザの端末装置７００がサーバの認証装置８００へ暗号化して送信してくる、端末装置７００が暗号化されたクライアント側のデータ鍵ｃｄｋを復号するためにも用いることができる。

【０１０６】
セッション鍵ＳＫが生成されると、サーバの認証装置８００は、データ鍵ｓｄｋをセッション鍵ＳＫを用いて暗号化し、ユーザの端末装置７００へ送信する。或いは、ユーザの端末装置７００は、端末装置７００が生成したセッション鍵を用いてクライアント側データ鍵ｃｄｋを暗号化し、サーバの認証装置８００へ送信する。サーバの認証装置８００は、自身で生成したセッション鍵ＳＫを用いて受信したクライアント側データ鍵ｃｄｋを復号する。成功すると、サーバの認証装置８００は、データ鍵ｄｋをｃｄｋとｓｄｋとを用いて復元することができる。

【０１０７】
［３．認証システム５００の変形例］

実施例１に係る認証システム１００の場合と同様に、実施例２に係る認証システム５００においても、サーバの認証装置８００のマスク演算器８１６における値Ｙの計算方法を以下のように変形することができる。この変形においては、第１ブラインド及び第２ブラインドが次のように計算される。

第１ブラインド：$W_{ij}{}^{y}$

第２ブラインド：$U^{y \cdot r}$（ただしｒ＝H_2（Ｃ‖S_i‖Ｕ））

すなわち、はじめの例ではｒがパスワード認証データW_{ij}に関するべき乗計算に用いられていたのに対し、この変形例ではｒが公開値Ｕに関するべき乗計算に用いられている。そして、マスク演算器８１６で得られる値Ｙは、次のように求められる。

$Y = U^{y \cdot r} \cdot W_{ij}{}^{y}$

【０１０８】

これらの変形に伴い、ユーザの端末装置７００のブラインド生成器７１４におけるブラインドｂの計算式も、次のように変形される。

$b = u \times r + v_{ij} \mod q$　（ただしｒ＝H_2（Ｃ‖S_i‖Ｕ））

すなわち、はじめの例ではｒが結合値v_{ij}に関する項に乗じられていたのに対し、この変形例ではｒが乱数ｕに乗じられる。

【０１０９】

［４．認証システム５００の更なる変形例］

実施例２に係る認証システム５００を次のように変形することにより、攻撃者のパスワードオンライン攻撃を検出する機能を付加することができる。

【０１１０】

前述した認証システム５００の初期化処理に加えて、ユーザの端末装置７００は、ＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）生成用の鍵ＭａｃＫを、安全な通信路を通じてサーバの認証装置８００へ送信する。ユーザの端末装置７００は、内部にあるメモリ７０２に他の記録情報とともに鍵ＭａｃＫを保存する。サーバの認証装置８００は、ユーザの端末装置７００から受信した鍵ＭａｃＫを、内部にあるメモリあるいはデータベース８０２に他の記録情報とともに保存する。

【０１１１】

前述した認証システム５００のｊ番目プロトコル実行処理に加えて、ユーザの端末装置７００は、サーバの認証装置８００へ送信するメッセージに対して、内部にあるメモリ７０２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをサーバの認証装置８００へ送信する。同じように、サーバの認証装置８００も、ユーザの端末装置７００へ送信するメッセージに対して、内部にあるメモリあるいはデータベース８０２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをユーザの端末装置７００へ送信する。送信されたＭＡＣは、クライアント及びサーバの各々において、各々に保存されている鍵ＭａｃＫを用いてそれぞれ検証される。

【０１１２】

認証システム５００のｊ番目プロトコル実行処理において、ＭＡＣの検証が失敗した場合など何かのエラーが発生して処理が中断された場合は、ユーザの端末装置７００とサーバの認証装置８００は各自のメモリあるいはデータベースに、その時送受信したメッセージと他の情報（例えば、時間、ＩＰアドレスなど）をログとして保存する。

【０１１３】

認証システム５００のｊ番目プロトコル終了後、ユーザの端末装置７００とサーバの認証装置８００がお互いに認証してセッション鍵を共有した場合、サーバの認証装置８００は、内部にあるメモリあるいはデータベース８０２に保存していたこれまでのログ情報を、セッション鍵により保護される安全な通信路を通じて、ユーザの端末装置７００へ送信するとともにそれらのログ情報を削除する。ユーザの端末装置７００は、サーバの認証装置８００から受信したログ情報と内部にあるメモリ７０２に保存していたこれまでのログ情報を比較することで攻撃者のパスワードに関するオンライン辞書攻撃の回数をユーザに表示させる。ユーザの端末装置７００は内部にあるメモリ７０２にこれまで保存していた

ログ情報を削除する。

【０１１４】

上記の認証システム５００のオンライン辞書攻撃検出機能は、ＭＡＣの代わりに署名（Ｄｉｇｉｔａｌ　Ｓｉｇｎａｔｕｒｅ）を使ってもよい。

【０１１５】

実施例２に係る認証システム５００も、実施例１に係る認証システム１００と同様に、サーバにおけるマスター秘密の計算を、式：

$K = g^{y}$

で行うことができるため、g^{uy}のべき乗計算が必要な従来技術よりも、サーバの計算量を大きく抑えることができる。また、クライアントにおけるマスター秘密の計算も、式：

$K = \underline{Y^{(1/b \bmod q)}}$

で行うことができるため、g^{uy}のべき乗計算が必要な従来技術よりも、やはり計算量を抑えることができる。これらの利点は上述の変形例においても失われない。パスワード及びパスワード認証データＷに加えて二要素認証方式を用いたことによる安全性と相まって、認証システム５００は、ＫＣＩ攻撃を含めてパブリックネットワーク上の様々な攻撃に対して非常に高度の安全性を提供することができ、且つクライアント及びサーバの計算量を従来方式に比べて減少させることに成功している。

【０１１６】

〔実施例３：実施例１のシステムの別の応用例〕

次に、実施例１で紹介したパスワードのみを用いた認証方式を、二要素認証方式に応用した別の例を説明する。図９は、この応用例を説明するために例示として用いる認証システム５００’の全体構成を描いている。

【０１１７】

認証システム５００’は、ユーザの端末装置９００とサーバの認証装置１０００とで構成される。認証システム５００’において、ユーザの端末装置９００はユーザから入力されたパスワードに加えてメモリ９０２などに保持されている記録情報に基づき、ある特徴的な演算を行う。サーバの認証装置１０００は、識別子ＷＩＤと認証データと公開値Ｔをデータベース１００２などに保持している。識別子ＷＩＤは、例えば、端末装置９００の識別子とカウント値とを含むものであることができる。端末装置９００とサーバの認証装置１０００はインターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保すると共に、各自の記録情報を次のセッションのために更新していく。それにより、サーバの認証装置１０００は、ユーザの認証データに対してパスワードを全数探索することができなくなり、ユーザの端末装置９００は記録情報が漏洩したとしても安全性が落ちることがない。

【０１１８】

［１．認証システム５００’の初期化］

初めに図１０を参照して、認証システム５００’における初期化処理について説明する。図１０は、認証システム５００’の初期化処理に係る、ユーザの端末装置９００とサーバの認証装置１０００の機能構成及び動作を説明するためのブロック図である。認証システム５００’の初期化処理において、ユーザの端末装置９００は、ユーザから入力されたパスワードに基づいて、サーバの認証装置１０００と安全な通信路（例えば、直接に登録したり、郵便で送付したり、あるいは電話で知らせるなど）を用いて初期化処理を行い、ユーザの端末装置９００は内部にあるメモリ９０２へ記録情報としてＣＳ１を保存し、サーバの認証装置１０００は内部にあるメモリあるいはデータベース１００２へ記録情報としてＳＳ１を保存する。

【０１１９】

＜ユーザの端末装置９００における作業＞

図１０に示すように、ユーザの端末装置９００における初期化処理では、結合器９０４は、ユーザから入力されたパスワードｐｗと乱数発生器９０６によりランダムに発生させた乱数s_{i1}（$s_{i1} \in (Z/qZ)^{*}$）を入力として、結合値v_{i1}を計算式：

$v_{i1}=s_{i1}+H_1(C \| S_i \| pw) \mod q$
により計算して出力する。ここで、S_iはｉ番目のサーバを表す。公開値演算器９１２は、乱数発生器９０６によりランダムに発生させた乱数ｔ（$t \in (Z/qZ)^*$）を入力として、公開値Ｔを計算式：
$T=g^t$
により計算して出力し、識別子ＷＩＤと認証データと公開値「ＷＩＤ，v_{i1}，Ｔ」をサーバの認証装置１０００へ送信する。ユーザの端末装置９００は、サーバのＩＤと識別子ＷＩＤと乱数発生器９０６により発生させた乱数s_{i1}とｔを、ユーザの端末装置９００の内部にあるメモリ９０２へ記録情報「ＣＳ１」として「S_i，ＷＩＤ，s_{i1}，ｔ」を保存する。

【０１２０】
＜サーバの認証装置１０００における作業＞
図１０に示すように、サーバの認証装置１０００における初期化処理において、サーバの認証装置１０００は、ユーザの端末装置９００から受信した識別子ＷＩＤと認証データと公開値「ＷＩＤ，v_{i1}，Ｔ」を、サーバの認証装置１０００の内部にあるメモリあるいはデータベース１００２へ記録情報「ＳＳ１」として「ＷＩＤ，v_{i1}，Ｔ」を保存する。

【０１２１】
［２．認証システム５００’のｊ番目プロトコル実行］
次に、図１１および図１２を参照して、認証システム５００’におけるｊ番目プロトコル実行処理について説明する。図１１および図１２は、認証システム５００’のｊ番目プロトコル実行を行う、ユーザの端末装置９００とサーバの認証装置１０００の機能構成及び動作を、それぞれ説明するためのブロック図である。図１１および図１２において、ｊはｊ≧１になるような整数である。認証システム５００’のｊ番目プロトコル実行において、ユーザの端末装置９００は、ユーザから入力されたパスワードｐｗとメモリ９０２に保持している記録情報ＣＳｊとに基づいて、ある特徴的な演算を行う。そして、記録情報ＳＳｊをデータベースなどに保持しているサーバの認証装置１０００と、インターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保する。以下に説明されるｊ番目プロトコル実行は、認証システム５００’の初期化が完了した後（ｊ＝１の時）、あるいは認証システム５００’のｊ－１番目プロトコル実行が終了した後（すなわち、ＣＳｊ＝（S_i，ＷＩＤ，s_{ij}，ｔ）とＳＳｊ＝（ＷＩＤ，v_{ij}，Ｔ）の時）に、ユーザの端末装置９００及びサーバの認証装置１０００によって実行される。

【０１２２】
＜ユーザの端末装置９００における作業＞
まず、図１１を参照して、ユーザの端末装置９００における認証システム５００’のｊ番目プロトコル実行処理について説明する。前述した認証システム５００’の初期化処理の後、認証システム５００’のｊ番目プロトコル実行に先だって、ユーザの端末装置９００は、記録情報「ＣＳｊ」として「S_i，ＷＩＤ，s_{ij}，ｔ」を内蔵するメモリ９０２に予め保持している。

【０１２３】
結合器９０４は、ユーザから入力されたパスワードｐｗとメモリ９０２から読み出した乱数s_{ij}とを入力として、結合値v_{ij}を計算式：
$v_{ij}=s_{ij}+H_1(C \| S_i \| pw) \mod q$
により計算して出力する。公開値演算器９１２は、乱数発生器９０６によりランダムに発生させた乱数ｕ（$u \in (Z/qZ)^*$）を入力として、公開値Ｕを計算式：
$U=g^u$
により計算して出力する。

【０１２４】
ブラインド生成器９１４は、結合器９０４により出力された結合値v_{ij}と乱数発生器

９０６によりランダムに発生させた乱数ｕとメモリ９０２から読み出した乱数ｔと、公開値演算器９１２により出力された公開値Ｕとを入力として、ブラインドｂを計算式：
$b = u + t \times v_{ij} \times r \mod q$
により計算して出力する。ここで、ｒは：
$r = H_2(C \| S_i \| U)$
である。逆ブラインド演算器９１６は、ブラインド生成器９１４により出力されたブラインドｂを入力として、逆ブラインドb^{-1}を計算式：
$b^{-1} \mod q$
により計算して出力する。

【０１２５】

ユーザの端末装置９００は、サーバの認証装置１０００に対して、メモリ９０２から読み出した識別子ＷＩＤと公開値演算器９１２により出力された公開値Ｕを、図示しない通信処理部を介して送信する。

【０１２６】

しばらくすると、ユーザの端末装置９００は、サーバの認証装置１０００からメッセージ「S_i，Ｙ」を受信する。マスター秘密生成器９１８は、サーバの認証装置１０００から受信した値Ｙと、逆ブラインド演算器９１６により出力された逆ブラインドb^{-1}を入力として、マスター秘密Ｋを計算式：
$K = Y^{(1/b \mod q)}$
により計算して出力する。

【０１２７】

続いて、認証子生成器９２０は、メモリ９０２から読み出した識別子ＷＩＤと公開値演算器９１２により出力された公開値Ｕとサーバの認証装置１０００から受信した値Ｙと結合器９０４により出力された結合値v_{ij}とマスター秘密生成器９１８により出力されたマスター秘密Ｋとを入力として、認証子Ｖｃを計算式：
$Vc = H(1 \| WID \| S_i \| U \| Y \| v_{ij} \| K)$
により計算して入力メッセージＩｎｐｕｔＭｓｇとともに出力する。ここで、入力メッセージは
$InputMsg = WID \| S_i \| U \| Y \| v_{ij} \| K$
である。ここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）を使ってもよい。ユーザの端末装置９００は、図示しない通信処理部を介して、認証子生成器９２０により出力された認証子Ｖｃをサーバの認証装置１０００へ送信する。

【０１２８】

しばらくすると、ユーザの端末装置９００は、サーバの認証装置１０００からメッセージ「V_{S_i}」を受信する。認証子判断部９２２は、サーバの認証装置１０００から受信した認証子V_{S_i}が正しく生成された値であるかどうかを確認する。認証子判断部９２２は、認証子生成器９２０から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、サーバの認証装置１０００から受信した認証子V_{S_i}と比較する。この比較処理において、認証子V_{S_i}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部９２２は、エラーメッセージ発生器９２４に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器９２４はエラーメッセージを生成して処理を中断する。

【０１２９】

一方、認証子判断の判断処理において、認証子V_{S_i}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致したことを判断した場合は、サーバの認証装置１０００が正当な装置として認証して、続けて次の処理を行う。セッション鍵生成器９２６は、認証子生成器９２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵SK_{ij}を計算式：
$SK_{ij} = H(3 \| InputMsg)$
により計算して出力する。秘密値更新器９２８は、メモリ９０２から読み出した識別子Ｗ

ＩＤと乱数s_{ij}と認証子生成器９２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、ｊ＋１番目の識別子ＷＩＤと乱数$s_{i(j+1)}$を計算式：
$WID=H_1(WID \| InputMsg)$
$s_{i(j+1)}=s_{ij}+H_1(InputMsg) \mod q$
により計算して出力する。ユーザの端末装置９００は、メモリ９０２に保持している現在の識別子ＷＩＤと乱数との組である「ＷＩＤ，s_{ij}」を、秘密値更新器９２８により出力された次の識別子ＷＩＤと乱数の組である「ＷＩＤ，$s_{i(j+1)}$」に書き換える。

【０１３０】

＜サーバの認証装置１０００における作業＞

次に、図１２を参照して、サーバの認証装置１０００における認証システム５００’のｊ番目プロトコル実行処理について説明する。前述した認証システム５００’の初期化処理の後、認証システム５００’のｊ番目プロトコル実行の前には、サーバの認証装置１０００は、記録情報「ＳＳｊ」として「ＷＩＤ，v_{ij}，Ｔ」を格納装置の内部にあるメモリあるいはデータベース１００２に保持している。

【０１３１】

まず、サーバの認証装置１０００は、ユーザの端末装置９００からメッセージ「ＷＩＤ，Ｕ」を受信する。ＷＩＤ判断部１００４は、ユーザの端末装置９００から受信した識別子ＷＩＤの正確性を確認する。ＷＩＤ判断部１００４は、データベースから読み出した識別子ＷＩＤを、ユーザの端末装置９００から受信した識別子ＷＩＤと比較して一致しない場合、エラーメッセージ発生器１００６に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器１００６はエラーメッセージを生成して処理を中断する。一方、ＷＩＤ判断部１００４の判断処理において、データベースから読み出した識別子ＷＩＤをユーザの端末装置９００から受信した識別子ＷＩＤと比較して、一致した場合には、続けて次の処理を行う。

【０１３２】

マスター秘密生成器１００８は、乱数発生器１０１０によりランダムに発生させた乱数ｙ（$y \in (Z/qZ)^*$）を入力として、マスター秘密Ｋを計算式：
$K=g^y$
により計算して出力する。

【０１３３】

第１ブラインド生成器１０１２は、ユーザの端末装置９００から受信した公開値Ｕとデータベースから読み出した認証データv_{ij}と公開値Ｔと乱数発生器１０１０によりランダムに発生させた乱数ｙとを入力として、第１ブラインド$T^{y \cdot v_{ij} \cdot r}$を計算式：
$T^{y \cdot v_{ij} \cdot r}$
により計算して出力する。ここで、ｒは
$r=H_2(C \| S_i \| U)$
である。第２ブラインド生成器１０１４は、ユーザの端末装置９００から受信した公開値Ｕと乱数発生器１０１０によりランダムに発生させた乱数ｙとを入力として、第２ブラインドU^yを計算式：
U^y
により計算して出力する。マスク演算器１０１６は、第１ブラインド生成器１０１２により出力された第１ブラインド$T^{y \cdot v_{ij} \cdot r}$と第２ブラインド生成器１０１４により出力された第２ブラインドU^yとを入力として、値Ｙを
$Y=U^y \cdot T^{y \cdot v_{ij} \cdot r}$
により計算して出力する。サーバの認証装置１０００は、図示しない通信処理部を介して、ユーザの端末装置９００に対して、サーバのＩＤであるＳｉとマスク演算器１０１６により出力された値Ｙを送信する。

【０１３４】

続いて、入力メッセージ集合器１０１８が、ユーザの端末装置９００から受信した識別子ＷＩＤと公開値Ｕとマスク演算器１０１６により出力された値Ｙとデータベースから読

み出した認証データv_{ij}とマスター秘密生成器１００８により出力されたマスター秘密Ｋとを入力として、入力メッセージＩｎｐｕｔＭｓｇを出力する。ここで、入力メッセージは

$InputMsg = WID \| S_i \| U \| Y \| v_{ij} \| K$

である。しばらくすると、サーバの認証装置１０００は、ユーザの端末装置９００からメッセージ「Ｖｃ」を受信する。

【０１３５】

認証子判断部１０２０は、ユーザの端末装置９００から受信した認証子Ｖｃが正しく生成された値であるかどうかを確認する。認証子判断部１０２０は、入力メッセージ集合器１０１８から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、ユーザの端末装置９００から受信した認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）との比較処理を行う。この比較処理において、認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部１０２０は、エラーメッセージ発生器１０２２に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器１０２２はエラーメッセージを生成して処理を中断する。一方、上記比較処理において、認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致すると判断された場合は、ユーザの端末装置９００が正当な装置として認証されるので、続けて次の処理が行われる。なおここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）を使ってもよい。

【０１３６】

認証子生成器１０２４は、入力メッセージ集合器１０１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、認証子V_{S_i}を計算式：

$V_{S_i} = H(2 \| InputMsg)$

により計算して出力する。サーバの認証装置１０００は、図示しない通信処理部を介して、ユーザの端末装置９００に対して、認証子生成器１０２４により出力された認証子V_{S_i}を送信する。

【０１３７】

セッション鍵生成器１０２６は、入力メッセージ集合器１０１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵SK_{ij}を計算式：

$SK_{ij} = H(3 \| InputMsg)$

により計算して出力する。

秘密値更新器１０２８は、データベース１００２から読み出した識別子ＷＩＤと認証データv_{ij}と入力メッセージ集合器１０１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、ｊ＋１番目の識別子ＷＩＤと認証データ$v_{i(j+1)}$を計算式：

$WID = H_1(WID \| InputMsg)$

$v_{i(j+1)} = v_{ij} + H_1(InputMsg) \mod q$

により計算して出力する。サーバの認証装置１０００は、データベース１００２に保持している現在の識別子ＷＩＤと認証データの組である「ＷＩＤ，v_{ij}」を、秘密値更新器１０２８により出力された次の識別子ＷＩＤと認証データの組である「ＷＩＤ，$v_{i(j+1)}$」に書き換える。

【０１３８】

［３．認証システム５００’の変形例１］

実施例１に係る認証システム１００の場合と同様に、実施例３に係る認証システム５００’においても、サーバの認証装置１０００のマスク演算器１０１６における値Ｙの計算方法を以下のように変形することができる。この変形においては、第１ブラインド及び第２ブラインドが次のように計算される。

第１ブラインド：$T^{y \cdot v_{ij}}$

第２ブラインド：$U^{y \cdot r}$　（ただし$r = H_2(C \| S_i \| U)$）

【０１３９】

すなわち、はじめの例ではｒが公開値Ｔに関するべき乗計算に用いられていたのに対し

、この変形例ではｒが公開値Ｕに関するべき乗計算に用いられている。そして、マスク演算器１０１６で得られる値Ｙは、次のように求められる。
$Y = U^{y \cdot r \cdot T^{y \cdot v_{ij}}}$
【０１４０】
これらの変形に伴い、ユーザの端末装置９００のブラインド生成器９１４におけるブラインドｂの計算式も、次のように変形される。
$b = u \times r + t \times v_{ij} \mod q$ （ただし$r = H_2(C \| S_i \| U)$）
すなわち、はじめの例ではｒが結合値v_{ij}に関する項に乗じられていたのに対し、この変形例ではｒが乱数ｕに乗じられる。
【０１４１】
［４．認証システム５００’の変形例２］
次に、実施例３に係る認証システム５００’において、サーバの認証装置１０００のマスク演算器１０１６における値Ｙの計算方法を以下のように変形することができる。この変形においては、第１ブラインド及び第２ブラインドが次のように計算される。
第１ブラインド：$T^{y \cdot r}$ （ただし$r = H_2(C \| S_i \| U \| v_{ij})$）
第２ブラインド：U^{y}
【０１４２】
すなわち、はじめの例ではｒが$r = H_2(C \| S_i \| U)$で計算されていたのに対し、この変形例ではｒが$r = H_2(C \| S_i \| U \| v_{ij})$で計算されている。そして、マスク演算器１０１６で得られる値Ｙは、次のように求められる。
$Y = U^{y \cdot T^{y \cdot r}}$
【０１４３】
これらの変形に伴い、ユーザの端末装置９００のブラインド生成器９１４におけるブラインドｂの計算式も、次のように変形される。
$b = u + t \times r \mod q$ （ただし$r = H_2(C \| S_i \| U \| v_{ij})$）
すなわち、はじめの例ではｒが$r = H_2(C \| S_i \| U)$で計算されていたのに対し、この変形例ではｒが$r = H_2(C \| S_i \| U \| v_{ij})$で計算されている。
【０１４４】
［５．認証システム５００’のオンライン辞書攻撃検出機能］
次に、実施例３に係る認証システム５００’において、攻撃者のパスワードオンライン攻撃を検出する機能について説明する。
【０１４５】
前述した認証システム５００’の初期化処理に加えて、ユーザの端末装置９００は、ＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）生成用の鍵ＭａｃＫを、安全な通信路を通じてサーバの認証装置１０００へ送信する。ユーザの端末装置９００は、内部にあるメモリ９０２に他の記録情報とともに鍵ＭａｃＫを保存する。サーバの認証装置１０００は、ユーザの端末装置９００から受信した鍵ＭａｃＫを、内部にあるメモリあるいはデータベース１００２に他の記録情報とともに保存する。
【０１４６】
前述した認証システム５００’のｊ番目プロトコル実行処理に加えて、ユーザの端末装置９００は、サーバの認証装置１０００へ送信するメッセージに対して、内部にあるメモリ９０２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをサーバの認証装置１０００へ送信する。同じように、サーバの認証装置１０００も、ユーザの端末装置９００へ送信するメッセージに対して、内部にあるメモリあるいはデータベース１００２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをユーザの端末装置９００へ送信する。認証システム５００’のｊ番目プロトコル実行処理で、ＭＡＣの検証が失敗した場合など何かのエラーが発生して処理が中断された場合は、ユーザの端末装置９００とサーバの認証装置１０００は各自のメモリあるいはデータベースに、その時送受信したメッセージと他の情報（例えば、時間、ＩＰアドレスなど）をログとして保存する。

【０１４７】
認証システム５００’のｊ番目プロトコル終了後、ユーザの端末装置９００とサーバの認証装置１０００がお互いに認証してセッション鍵を共有した場合、サーバの認証装置１０００は、内部にあるメモリあるいはデータベース１００２に保存していたこれまでのログ情報を、セッション鍵により保護される安全な通信路を通じて、ユーザの端末装置９００へ送信するとともにそれらのログ情報を削除する。ユーザの端末装置９００は、サーバの認証装置１０００から受信したログ情報と内部にあるメモリ９０２に保存していたこれまでのログ情報を比較することで、攻撃者のパスワードに関するオンライン辞書攻撃の回数をユーザに表示させる。ユーザの端末装置９００は内部にあるメモリ９０２にこれまで保存していたログ情報を削除する。

【０１４８】
上記の認証システム５００’のオンライン辞書攻撃検出機能は、ＭＡＣの代わりに署名（Ｄｉｇｉｔａｌ　Ｓｉｇｎａｔｕｒｅ）を使ってもよい。

【０１４９】
〔実施例４：実施例１のシステムの更に別の応用例〕
次に、実施例１で紹介したパスワードのみを用いた認証方式を、二要素認証方式に応用した、更に別の例を説明する。図１３は、この応用例を説明するために例示として用いる認証システム５００”の全体構成を描いている。

【０１５０】
認証システム５００”は、ユーザの端末装置１１００とサーバの認証装置１２００とで構成される。認証システム５００”において、ユーザの端末装置１１００はユーザから入力されたパスワードに加えてメモリ１１０２などに保持されている記録情報に基づき、ある特徴的な演算を行う。サーバの認証装置１２００は、ハッシュされた一時ＩＤと認証データをデータベース１２０２などに保持している。端末装置１１００とサーバの認証装置１２００はインターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保すると共に、各自の記録情報を次のセッションのために更新していく。それにより、サーバの認証装置１２００は、ユーザの認証データに対してパスワードを全数探索することができなくなり、ユーザの端末装置１１００は記録情報が漏洩したとしても安全性が落ちることがない。

【０１５１】
［１．認証システム５００”の初期化］
初めに図１４を参照して、認証システム５００”における初期化処理について説明する。図１４は、認証システム５００”の初期化処理に係る、ユーザの端末装置１１００とサーバの認証装置１２００の機能構成及び動作を説明するためのブロック図である。認証システム５００”の初期化処理において、ユーザの端末装置１１００は、ユーザから入力されたパスワードに基づいて、サーバの認証装置１２００と安全な通信路（例えば、直接に登録したり、郵便で送付したり、あるいは電話で知らせるなど）を用いて初期化処理を行い、ユーザの端末装置１１００は内部にあるメモリ１１０２へ記録情報としてＣＳ１を保存し、サーバの認証装置１２００は内部にあるメモリあるいはデータベース１２０２へ記録情報としてＳＳ１を保存する。

【０１５２】
＜ユーザの端末装置１１００における作業＞
図１４に示すように、ユーザの端末装置１１００における初期化処理では、結合器１１０４は、ユーザから入力されたパスワードｐｗと乱数発生器１１０６によりランダムに発生させた乱数s_{i1}（$s_{i1} \in (Z/qZ)^*$）を入力として、結合値v_{i1}を計算式：
$$v_{i1} = s_{i1} + H_1(C \| S_i \| pw) \mod q$$
により計算して出力する。ここで、S_iはｉ番目のサーバを表す。ＨＩＤ生成器１１０７は、ＩＤ用乱数発生器１１０５によりランダムに発生させたＩＤ値id_{i1}（$id_{i1} \in \{0,1\}^k$）を入力として、ＨＩＤ値hid_{i1}を計算式：
$$hid_{i1} = H(4 \| id_{i1})$$

により計算して出力し、ＨＩＤ値と認証データ「hid_{i1}，v_{i1}」をサーバの認証装置１２００へ送信する。ユーザの端末装置１１００は、サーバのＩＤとＩＤ用乱数発生器１１０５により発生させたＩＤ値と乱数発生器１１０６により発生させた乱数s_{i1}を、ユーザの端末装置１１００の内部にあるメモリ１１０２へ記録情報「ＣＳ１」として「S_i，id_{i1}，s_{i1}」を保存する。

【０１５３】

＜サーバの認証装置１２００における作業＞

図１４に示すように、サーバの認証装置１２００における初期化処理において、サーバの認証装置１２００は、ユーザの端末装置１１００から受信したＨＩＤ値と認証データ「hid_{i1}，v_{i1}」を、サーバの認証装置１２００の内部にあるメモリあるいはデータベース１２０２へ記録情報「ＳＳ１」として「hid_{i1}，v_{i1}」を保存する。

【０１５４】

［２．認証システム５００”のｊ番目プロトコル実行］

次に、図１５および図１６を参照して、認証システム５００”におけるｊ番目プロトコル実行処理について説明する。図１５および図１６は、認証システム５００”のｊ番目プロトコル実行を行う、ユーザの端末装置１１００とサーバの認証装置１２００の機能構成及び動作を、それぞれ説明するためのブロック図である。図１５および図１６において、ｊはｊ≧１になるような整数である。認証システム５００”のｊ番目プロトコル実行において、ユーザの端末装置１１００は、ユーザから入力されたパスワードｐｗとメモリ１１０２に保持している記録情報ＣＳｊとに基づいて、ある特徴的な演算を行う。そして、記録情報ＳＳｊをデータベースなどに保持しているサーバの認証装置１２００と、インターネットのようなパブリックネットワークを通じてお互いに相互認証し、その相互認証が成功した時のみ、お互いに同じセッション鍵を確保する。以下に説明されるｊ番目プロトコル実行は、認証システム５００”の初期化が完了した後（ｊ＝１の時）、あるいは認証システム５００”のｊ－１番目プロトコル実行が終了した後（すなわち、ＣＳｊ＝（S_i，id_{ij}，s_{ij}）とＳＳｊ＝（hid_{ij}，v_{ij}）の時）に、ユーザの端末装置１１００及びサーバの認証装置１２００によって実行される。

【０１５５】

＜ユーザの端末装置１１００における作業＞

まず、図１５を参照して、ユーザの端末装置１１００における認証システム５００”のｊ番目プロトコル実行処理について説明する。前述した認証システム５００”の初期化処理の後、認証システム５００”のｊ番目プロトコル実行の前には、ユーザの端末装置１１００は、記録情報「ＣＳｊ」として「S_i，id_{ij}，s_{ij}」を内蔵するメモリ１１０２に保持している。

【０１５６】

結合器１１０４は、ユーザから入力されたパスワードｐｗとメモリ１１０２から読み出した乱数s_{ij}とを入力として、結合値v_{ij}を計算式：

$$v_{ij}=s_{ij}+H_1(C \| S_i \| pw) \mod q$$

により計算して出力する。公開値演算器１１１２は、乱数発生器１１０６によりランダムに発生させた乱数ｕ（ｕ∈（Ｚ／ｑＺ）＊）を入力として、公開値Ｕを計算式：

$$U=g^u$$

により計算して出力する。ＨＩＤ生成器１１０７は、メモリ１１０２から読み出したＩＤ値id_{ij}を入力として、ＨＩＤ値を計算式：

$$hid_{ij}=H(4 \| id_{ij})$$

により計算して出力する。

【０１５７】

ブラインド生成器１１１４は、結合器１１０４により出力された結合値v_{ij}と乱数発生器１１０６によりランダムに発生させた乱数ｕとＨＩＤ生成器１１０７により出力されたＨＩＤ値hid_{ij}と、公開値演算器１１１２により出力された公開値Ｕとを入力として、ブラインドｂを計算式：

$b = u + v_{ij} \times r \mod q$
により計算して出力する。ここで、ｒは：
$r = H_2(hid_{ij} \| S_i \| U)$
である。逆ブラインド演算器１１１６は、ブラインド生成器１１１４により出力されたブラインドｂを入力として、逆ブラインドb^{-1}を計算式：
$b^{-1} \mod q$
により計算して出力する。
【０１５８】
ユーザの端末装置１１００は、サーバの認証装置１２００に対して、メモリ１１０２から読み出したＩＤ値id_{ij}と公開値演算器１１１２により出力された公開値Ｕを、図示しない通信処理部を介して送信する。
【０１５９】
しばらくすると、ユーザの端末装置１１００は、サーバの認証装置１２００からメッセージ「S_i，Ｙ」を受信する。マスター秘密生成器１１１８は、サーバの認証装置１２００から受信した値Ｙと、逆ブラインド演算器１１１６により出力された逆ブラインドb^{-1}を入力として、マスター秘密Ｋを計算式：
$K = Y^{(1/b \mod q)}$
により計算して出力する。
【０１６０】
続いて、認証子生成器１１２０は、ＨＩＤ生成器１１０７により出力されたＨＩＤ値hid_{ij}と公開値演算器１１１２により出力された公開値Ｕとサーバの認証装置１２００から受信した値Ｙと結合器１１０４により出力された結合値v_{ij}とマスター秘密生成器１１１８により出力されたマスター秘密Ｋとを入力として、認証子Ｖｃを計算式：
$Vc = H(1 \| hid_{ij} \| S_i \| U \| Y \| v_{ij} \| K)$
により計算して入力メッセージＩｎｐｕｔＭｓｇとともに出力する。ここで、入力メッセージは
$InputMsg = hid_{ij} \| S_i \| U \| Y \| v_{ij} \| K$
である。ここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）を使ってもよい。ユーザの端末装置１１００は、図示しない通信処理部を介して、認証子生成器１１２０により出力された認証子Ｖｃをサーバの認証装置１２００へ送信する。
【０１６１】
しばらくすると、ユーザの端末装置１１００は、サーバの認証装置１２００からメッセージ「V_{S_i}」を受信する。認証子判断部１１２２は、サーバの認証装置１２００から受信した認証子V_{S_i}が正しく生成された値であるかどうかを確認する。認証子判断部１１２２は、認証子生成器１１２０から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、サーバの認証装置１２００から受信した認証子V_{S_i}と比較する。この比較処理において、認証子V_{S_i}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部１１２２は、エラーメッセージ発生器１１２４に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器１１２４はエラーメッセージを生成して処理を中断する。
【０１６２】
一方、認証子判断の判断処理において、認証子V_{S_i}とハッシュ関数Ｈ（２‖ＩｎｐｕｔＭｓｇ）が一致したことを判断した場合は、サーバの認証装置１２００が正当な装置として認証して、続けて次の処理を行う。セッション鍵生成器１１２６は、認証子生成器１１２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵SK_{ij}を計算式：
$SK_{ij} = H(3 \| InputMsg)$
により計算して出力する。秘密値更新器１１２８は、メモリ１１０２から読み出した乱数s_{ij}と認証子生成器１１２０から入力された入力メッセージＩｎｐｕｔＭｓｇを入力と

して、ｊ＋１番目の乱数$s_{i(j+1)}$を計算式：
$s_{i(j+1)}=s_{ij}+H_1(InputMsg) \mod q$
により計算して出力する。ユーザの端末装置１１００は、メモリ１１０２に保持している現在の乱数である「s_{ij}」を、秘密値更新器１１２８により出力された次の乱数である「$s_{i(j+1)}$」に書き換える。

【０１６３】

＜サーバの認証装置１２００における作業＞

次に、図１６を参照して、サーバの認証装置１２００における認証システム５００”のｊ番目プロトコル実行処理について説明する。前述した認証システム５００”の初期化処理の後、認証システム５００”のｊ番目プロトコル実行の前には、サーバの認証装置１２００は、記録情報「ＳＳｊ」として「hid_{ij}，v_{ij}」を格納装置の内部にあるメモリあるいはデータベース１２０２に保持している。

【０１６４】

まず、サーバの認証装置１２００は、ユーザの端末装置１１００からメッセージ「id_{ij}，Ｕ」を受信する。ＨＩＤ生成器１２０７は、ユーザの端末装置１１００から受信したＩＤ値id_{ij}を入力として、ＨＩＤ値hid_{ij}を計算式：
$hid_{ij}=H(4\|id_{ij})$
により計算して出力する。ＨＩＤ判断部１２０４は、データベースから読み出したＨＩＤ値hid_{ij}を、ＨＩＤ生成器１２０７により出力されたＨＩＤ値と比較して一致しない場合、エラーメッセージ発生器１２０６に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器１２０６はエラーメッセージを生成して処理を中断する。一方、ＨＩＤ判断部１２０４の判断処理において、データベースから読み出したＨＩＤ値hid_{ij}をＨＩＤ生成器１２０７により出力されたＨＩＤ値hid_{ij}と比較して、一致した場合には、続けて次の処理を行う。

【０１６５】

マスター秘密生成器１２０８は、乱数発生器１２１０によりランダムに発生させた乱数ｙ（$y\in(Z/qZ)^*$）を入力として、マスター秘密Ｋを計算式：
$K=g^y$
により計算して出力する。

【０１６６】

第１ブラインド生成器１２１２は、ユーザの端末装置１１００から受信した公開値Ｕとデータベースから読み出したＨＩＤ値hid_{ij}と認証データv_{ij}と乱数発生器１２１０によりランダムに発生させた乱数ｙとを入力として、第１ブラインド$g^{y\cdot v_{ij}\cdot r}$を計算式：
$g^{y\cdot v_{ij}\cdot r}$
により計算して出力する。ここで、ｒは
$r=H_2(hid_{ij}\|S_i\|U)$
である。第２ブラインド生成器１２１４は、ユーザの端末装置１１００から受信した公開値Ｕと乱数発生器１２１０によりランダムに発生させた乱数ｙとを入力として、第２ブラインドU^yを計算式：
U^y
により計算して出力する。マスク演算器１２１６は、第１ブラインド生成器１２１２により出力された第１ブラインド$g^{y\cdot v_{ij}\cdot r}$と第２ブラインド生成器１２１４により出力された第２ブラインドU^yとを入力として、値Ｙを
$Y=U^y\cdot g^{y\cdot v_{ij}\cdot r}$
により計算して出力する。サーバの認証装置１２００は、図示しない通信処理部を介して、ユーザの端末装置１１００に対して、サーバのＩＤであるS_iとマスク演算器１２１６により出力された値Ｙを送信する。

【０１６７】

続いて、入力メッセージ集合器１２１８が、ユーザの端末装置１１００から受信した公

開値Ｕとマスク演算器１２１６により出力された値Ｙとデータベースから読み出したＨＩＤ値hid_{ij}と認証データv_{ij}とマスター秘密生成器１２０８により出力されたマスター秘密Ｋとを入力として、入力メッセージＩｎｐｕｔＭｓｇを出力する。ここで、入力メッセージは

$InputMsg = hid_{ij} \| S_i \| U \| Y \| v_{ij} \| K$

である。しばらくすると、サーバの認証装置１２００は、ユーザの端末装置１１００からメッセージ「Ｖｃ」を受信する。

【０１６８】

認証子判断部１２２０は、ユーザの端末装置１１００から受信した認証子Ｖｃが正しく生成された値であるかどうかを確認する。認証子判断部１２２０は、入力メッセージ集合器１２１８から入力された入力メッセージＩｎｐｕｔＭｓｇに基づいて、ユーザの端末装置１１００から受信した認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）との比較処理を行う。この比較処理において、認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致しない場合、認証子判断部１２２０は、エラーメッセージ発生器１２２２に対して、一致しないことを通知する。これを受けて、エラーメッセージ発生器１２２２はエラーメッセージを生成して処理を中断する。一方、上記比較処理において、認証子Ｖｃとハッシュ関数Ｈ（１‖ＩｎｐｕｔＭｓｇ）が一致すると判断された場合は、ユーザの端末装置１１００が正当な装置として認証されるので、続けて次の処理が行われる。なおここで、ハッシュ関数Ｈの代わりにＭＡＣ（Ｍｅｓｓａｇｅ Ａｕｔｈｅｎｔｉｃａｔｉｏｎ Ｃｏｄｅ）を使ってもよい。

【０１６９】

認証子生成器１２２４は、入力メッセージ集合器１２１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、認証子V_{S_i}を計算式：

$V_{S_i} = H(2 \| InputMsg)$

により計算して出力する。サーバの認証装置１２００は、図示しない通信処理部を介して、ユーザの端末装置１１００に対して、認証子生成器１２２４により出力された認証子V_{S_i}を送信する。

【０１７０】

セッション鍵生成器１２２６は、入力メッセージ集合器１２１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、セッション鍵SK_{ij}を計算式：

$SK_{ij} = H(3 \| InputMsg)$

により計算して出力する。秘密値更新器１２２８は、データベース１２０２から読み出した認証データv_{ij}と入力メッセージ集合器１２１８から入力された入力メッセージＩｎｐｕｔＭｓｇを入力として、ｊ＋１番目の認証データ$v_{i(j+1)}$を計算式：

$v_{i(j+1)} = v_{ij} + H_1(InputMsg) \mod q$

により計算して出力する。サーバの認証装置１２００は、データベース１２０２に保持している現在の認証データである「v_{ij}」を、秘密値更新器１２２８により出力された次の認証データである「$v_{i(j+1)}$」に書き換える。

【０１７１】

［３．認証システム５００”のｊ番目プロトコル終了後］

次に図１７を参照して、認証システム５００”におけるｊ番目プロトコル終了後の処理について説明する。図１７は、認証システム５００”のｊ番目プロトコル終了後の処理に係る、ユーザの端末装置１１００とサーバの認証装置１２００の機能構成及び動作を説明するためのブロック図である。認証システム５００”のｊ番目プロトコル終了後の処理において、ユーザの端末装置１１００は、セッション鍵生成器１１２６により出力されたセッション鍵SK_{ij}を用いて、次のセッションのためのＨＩＤ値$hid_{i(j+1)}$を、サーバの認証装置１２００へ安全に送信し、ユーザの端末装置１１００は内部にあるメモリ１１０２へ次のＩＤ値である$id_{i(j+1)}$を保存し、サーバの認証装置１２００は内部にあるメモリあるいはデータベース１２０２へ次のＨＩＤ値である$hid_{i(j+1)}$を保存する。

【０１７２】
＜ユーザの端末装置１１００における作業＞
図１７に示すように、ユーザの端末装置１１００におけるｊ番目プロトコル終了後の処理では、ＨＩＤ生成器１１０７は、ＩＤ用乱数発生器１１０５によりランダムに発生させたＩＤ値$id_{i(j+1)}$（$id_{i(j+1)} \in \{0,1\}^k$）を入力として、ｊ＋１番目のＨＩＤ値$hid_{i(j+1)}$を計算式：
$hid_{i(j+1)}=H(4 \| id_{i(j+1)})$
により計算して出力し、ＨＩＤ値「$hid_{i(j+1)}$」をサーバの認証装置１２００へ送信する。ユーザの端末装置１１００は、ユーザの端末装置１１００の内部にあるメモリ１１０２に保持している現在のＩＤ値である「id_{ij}」を、ＩＤ用乱数発生器１１０５により発生させた次のＩＤ値である「$id_{i(j+1)}$」に書き換える。
＜サーバの認証装置１２００における作業＞
【０１７３】
図１７に示すように、サーバの認証装置１２００におけるｊ番目プロトコル終了後の処理において、サーバの認証装置１２００は、サーバの認証装置１２００の内部にあるメモリあるいはデータベース１２０２に保持している現在のＨＩＤ値である「hid_{ij}」を、ユーザの端末装置１１００から受信したｊ＋１番目のＨＩＤ値である「$hid_{i(j+1)}$」に書き換える。
【０１７４】
［４．認証システム５００”の変形例］
実施例１に係る認証システム１００の場合と同様に、実施例４に係る認証システム５００”においても、サーバの認証装置１２００のマスク演算器１２１６における値Ｙの計算方法を以下のように変形することができる。この変形においては、第１ブラインド及び第２ブラインドが次のように計算される。
第１ブラインド：$g^{y \cdot v_{ij}}$
第２ブラインド：$U^{y \cdot r}$　（ただし$r=H_2(hid_{ij} \| S_i \| U)$）
【０１７５】
すなわち、はじめの例ではｒがｇに関するべき乗計算に用いられていたのに対し、この変形例ではｒが公開値Ｕに関するべき乗計算に用いられている。そして、マスク演算器１２１６で得られる値Ｙは、次のように求められる。
$Y=U^{y \cdot r} \cdot g^{y \cdot v_{ij}}$
【０１７６】
これらの変形に伴い、ユーザの端末装置１１００のブラインド生成器１１１４におけるブラインドｂの計算式も、次のように変形される。
$b=u \times r+v_{ij} \mod q$　（ただし$r=H_2(hid_{ij} \| S_i \| U)$）
すなわち、はじめの例ではｒが結合値v_{ij}に関する項に乗じられていたのに対し、この変形例ではｒが乱数ｕに乗じられる。
【０１７７】
［５．認証システム５００”のオンライン辞書攻撃検出機能］
次に、実施例４に係る認証システム５００”において、攻撃者のパスワードオンライン攻撃を検出する機能について説明する。
【０１７８】
前述した認証システム５００”の初期化処理に加えて、ユーザの端末装置１１００は、ＭＡＣ（Ｍｅｓｓａｇｅ　Ａｕｔｈｅｎｔｉｃａｔｉｏｎ　Ｃｏｄｅ）生成用の鍵ＭａｃＫを、安全な通信路を通じてサーバの認証装置１２００へ送信する。ユーザの端末装置１１００は、内部にあるメモリ１１０２に他の記録情報とともに鍵ＭａｃＫを保存する。サーバの認証装置１２００は、ユーザの端末装置１１００から受信した鍵ＭａｃＫを、内部にあるメモリあるいはデータベース１２０２に他の記録情報とともに保存する。
【０１７９】
前述した認証システム５００”のｊ番目プロトコル実行処理に加えて、ユーザの端末装

置１１００は、サーバの認証装置１２００へ送信するメッセージに対して、内部にあるメモリ１１０２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをサーバの認証装置１２００へ送信する。同じように、サーバの認証装置１２００も、ユーザの端末装置１１００へ送信するメッセージに対して、内部にあるメモリあるいはデータベース１２０２から読み出した鍵ＭａｃＫを用いてＭＡＣを生成し、メッセージとともにＭＡＣをユーザの端末装置１１００へ送信する。認証システム５００”のｊ番目プロトコル実行処理で、ＭＡＣの検証が失敗した場合など何かのエラーが発生して処理が中断された場合は、ユーザの端末装置１１００とサーバの認証装置１２００は各自のメモリあるいはデータベースに、その時送受信したメッセージと他の情報（例えば、時間、ＩＰアドレスなど）をログとして保存する。

【０１８０】

前述した認証システム５００”のｊ番目プロトコル終了後の処理に加えて、ユーザの端末装置１１００とサーバの認証装置１２００がお互いに認証してセッション鍵を共有した場合、サーバの認証装置１２００は、内部にあるメモリあるいはデータベース１２０２に保存していたこれまでのログ情報を、セッション鍵により保護される安全な通信路を通じて、ユーザの端末装置１１００へ送信するとともにそれらのログ情報を削除する。ユーザの端末装置１１００は、サーバの認証装置１２００から受信したログ情報と内部にあるメモリ１１０２に保存していたこれまでのログ情報を比較することで攻撃者のパスワードに関するオンライン辞書攻撃の回数をユーザに表示させる。ユーザの端末装置１１００は内部にあるメモリ１１０２にこれまで保存していたログ情報を削除する。

【０１８１】

上記の認証システム５００”のオンライン辞書攻撃検出機能は、ＭＡＣの代わりに署名（Ｄｉｇｉｔａｌ　Ｓｉｇｎａｔｕｒｅ）を使ってもよい。

【０１８２】

以上、本発明の具現化形態のいくつかの例を説明したが、これらの例は本発明のアイディアの具現化形態を限定する目的で挙げられたのではなく、あくまで本発明によって開示される新たな技術思想のより深い理解に資するために挙げられたものである。ここで開示される技術思想の実施形態は上記の例に限定されるものではなく、その思想を逸脱することなく、様々な形態をとり得ることは言うまでもない。

【符号の説明】

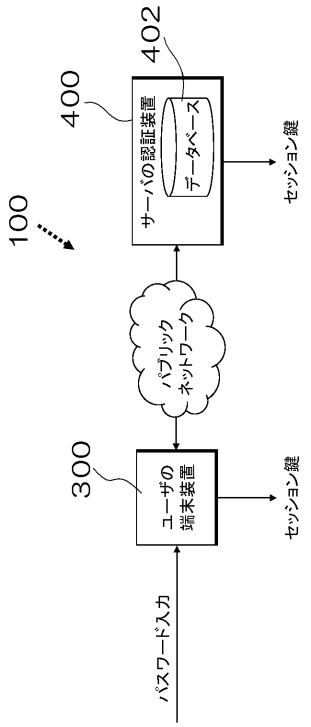
【０１８３】

１００　認証システム
３００　端末装置
３０２　パスワード認証データ生成器
３０４　公開値演算器
３０６　乱数発生器
３０８　ブラインド生成器
３１０　逆ブラインド演算器
３１２　マスター秘密生成器
３１４　認証子生成器
３１６　認証子判断部
３１８　エラーメッセージ発生器
３２０　セッション鍵生成器
４００　認証装置
４０２　データベース
４０４　第１ブラインド生成器
４０５　第２ブラインド生成器
４０６　乱数発生器
４０８　マスク演算器
４１２　マスター秘密生成器

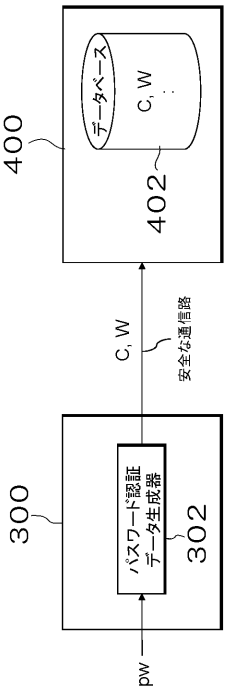
４１４　メッセージ集合器
４１６　認証子判断部
４１８　エラーメッセージ発生器
４２０　認証子生成器
４２２　セッション鍵生成器
５００　認証システム
５００’　認証システム
５００”　認証システム
７００　端末装置
７０２　メモリ
７０４　結合器
７０６　乱数発生器
７０８　二要素認証データ生成器
７１２　公開値演算器
７１４　ブラインド生成器
７１６　逆ブラインド演算器
７１８　マスター秘密生成器
７２０　認証子生成器
７２２　認証子判断部
７２４　エラーメッセージ発生器
７２６　セッション鍵生成器
７２８　秘密値更新器
８００　認証装置
８０２　データベース
８０４　ＷＩＤ判断部
８０６　エラーメッセージ発生器
８０８　マスター秘密生成器
８１０　乱数発生器
８１２　第１ブラインド生成器
８１４　第２ブラインド生成器
８１６　マスク演算器
８１８　メッセージ集合器
８２０　認証子判断部
８２２　エラーメッセージ発生器
８２４　認証子生成器
８２６　セッション鍵生成器
８２８　秘密値更新器
９００　端末装置
９０２　メモリ
９０４　結合器
９０６　乱数発生器
９１２　公開値演算器
９１４　ブラインド生成器
９１６　逆ブラインド演算器
９１８　マスター秘密生成器
９２０　認証子生成器
９２２　認証子判断部
９２４　エラーメッセージ発生器
９２６　セッション鍵生成器
９２８　秘密値更新器

１０００　認証装置
１００２　データベース
１００４　ＷＩＤ判断部
１００６　エラーメッセージ発生器
１００８　マスター秘密生成器
１０１０　乱数発生器
１０１２　第１ブラインド生成器
１０１４　第２ブラインド生成器
１０１６　マスク演算器
１０１８　メッセージ集合器
１０２０　認証子判断部
１０２２　エラーメッセージ発生器
１０２４　認証子生成器
１０２６　セッション鍵生成器
１０２８　秘密値更新器
１１００　端末装置
１１０２　メモリ
１１０４　結合器
１１０５　ＩＤ用乱数発生器
１１０６　乱数発生器
１１０７　生成器
１１１２　公開値演算器
１１１４　ブラインド生成器
１１１６　逆ブラインド演算器
１１１８　マスター秘密生成器
１１２０　認証子生成器
１１２２　認証子判断部
１１２４　エラーメッセージ発生器
１１２６　セッション鍵生成器
１１２８　秘密値更新器
１２００　認証装置
１２０２　データベース
１２０４　判断部
１２０６　エラーメッセージ発生器
１２０７　生成器
１２０８　マスター秘密生成器
１２１０　乱数発生器
１２１２　第１ブラインド生成器
１２１４　第２ブラインド生成器
１２１６　マスク演算器
１２１８　メッセージ集合器
１２２０　認証子判断部
１２２２　エラーメッセージ発生器
１２２４　認証子生成器
１２２６　セッション鍵生成器
１２２８　秘密値更新器

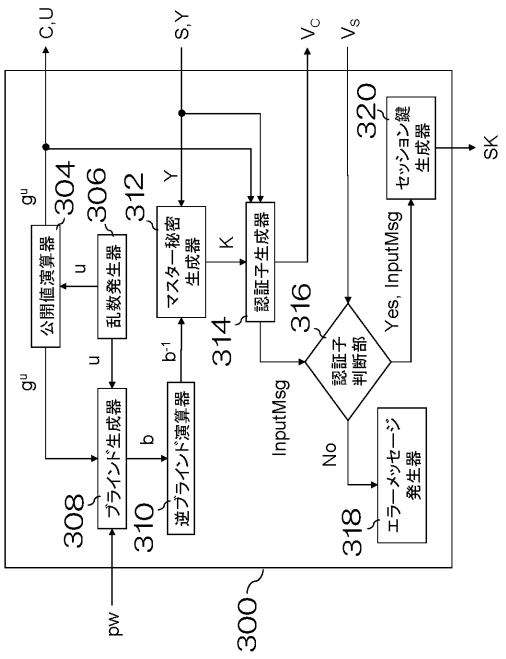
【図１】


100
400
402
サーバの認証装置
データベース
セッション鍵
パブリック
ネットワーク
300
ユーザの
端末装置
セッション鍵
パスワード入力

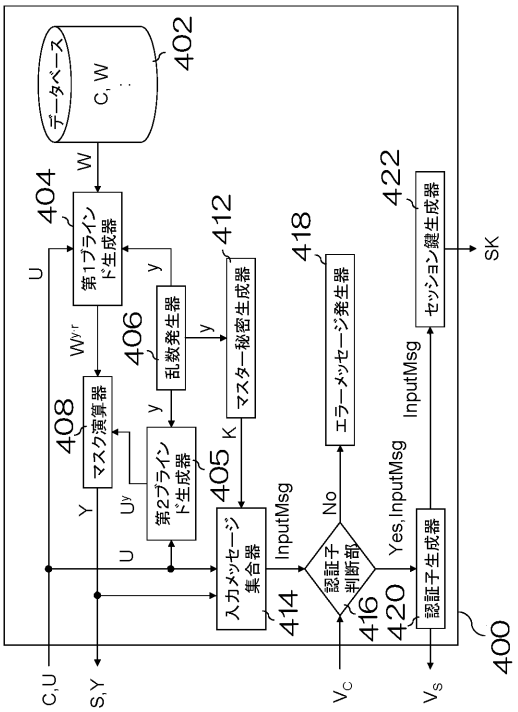
【図２】


400
データベース
C, W
402
C, W
安全な通信路
300
パスワード認証
データ生成器
302
pw

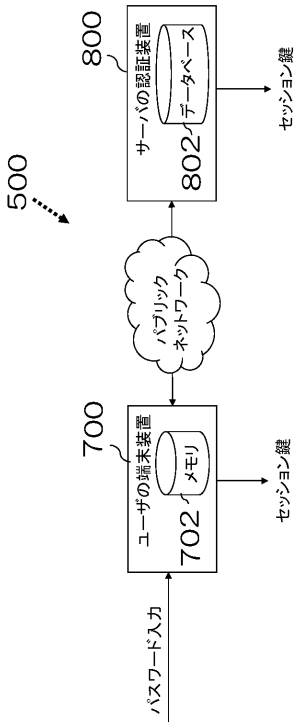
【図３】


C,U
S,Y
V_C
V_S
g^u
304
公開値演算器
306
乱数発生器
312
マスター秘密
生成器
Y
K
認証子生成器
316
320
セッション鍵
生成器
SK
u
b^{-1}
314
認証子
判断部
Yes, InputMsg
InputMsg
No
308
プラインド生成器
b
310
逆プラインド演算器
318
エラーメッセージ
発生器
pw
300

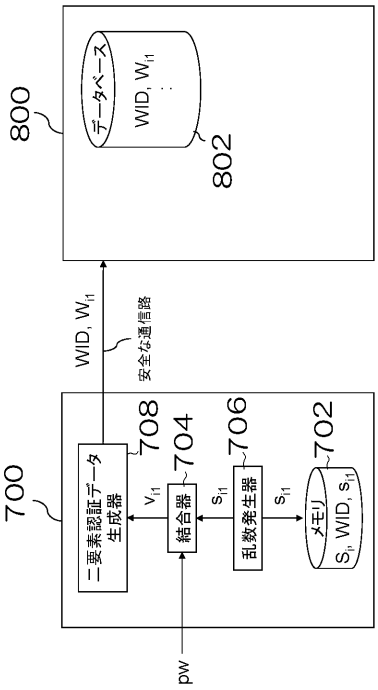
【図４】


データベース
C, W
402
404
第1プライン
ド生成器
W
U
W^{y_r}
406
乱数発生器
y
412
マスター秘密生成器
K
418
エラーメッセージ発生器
422
セッション鍵生成器
SK
408
マスク演算器
Y
U^y
第2プライン
ド生成器
405
入力メッセージ
集合器
InputMsg
414
認証子
判断部
No
Yes,InputMsg
InputMsg
416
420
認証子生成器
C,U
S,Y
V_C
V_S
400

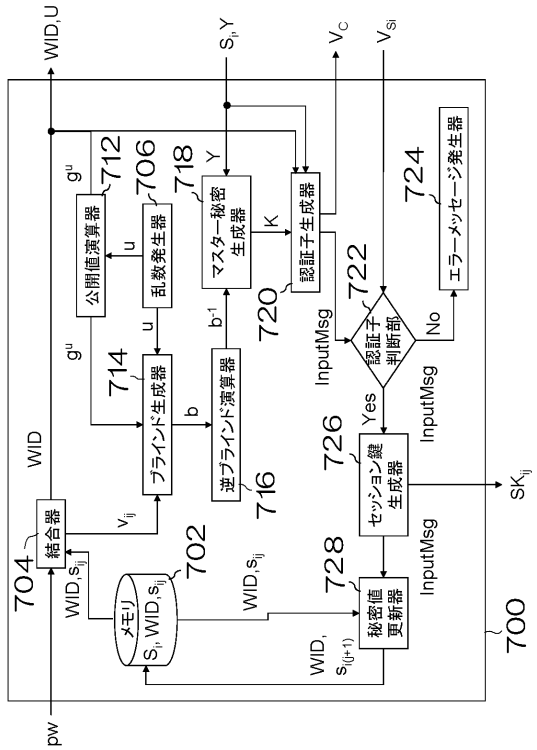
【図５】


500
800
サーバの認証装置
データベース
802
セッション鍵
パブリック
ネットワーク
700
ユーザの端末装置
メモリ
702
セッション鍵
パスワード入力

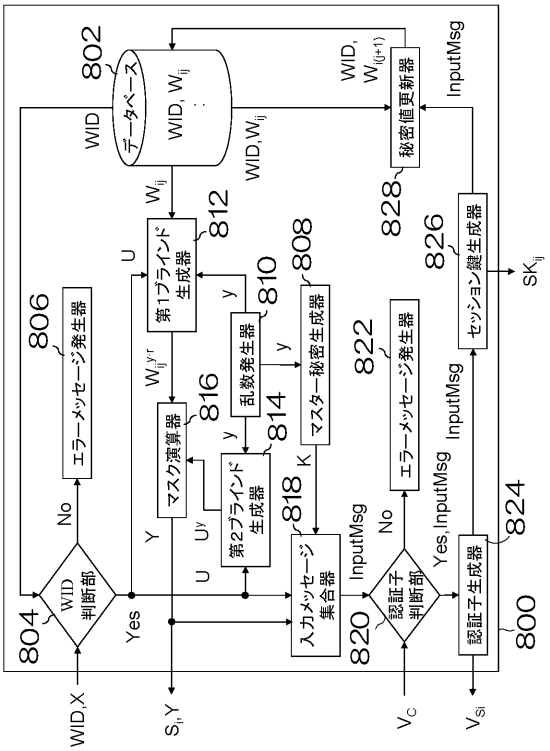
【図６】


800
データベース
WID, W_f1
802
WID, W_f1
安全な通信路
700
二要素認証データ生成器
708
704
706
702
v_f1
s_f1
s_f1
結合器
乱数発生器
メモリ
S_i, WID, s_f1
pw

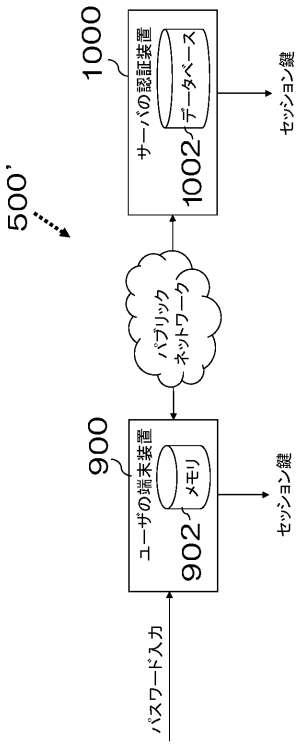
【図７】


WID,U
S_i,Y
V_C
V_Si
WID
g^u
712
706
718
Y
公開値演算器
乱数発生器
マスター秘密生成器
714
u
u
K
ブラインド生成器
b
b^-1
逆ブラインド演算器
認証子生成器
720
722
724
InputMsg
認証子判断部
No
エラーメッセージ発生器
704
結合器
v_ij
Yes
726
InputMsg
InputMsg
セッション鍵生成器
SK_ij
WID,s_ij
メモリ
S_i, WID,s_ij
702
716
728
秘密値更新器
InputMsg
WID,
s_i(j+1)
700
pw

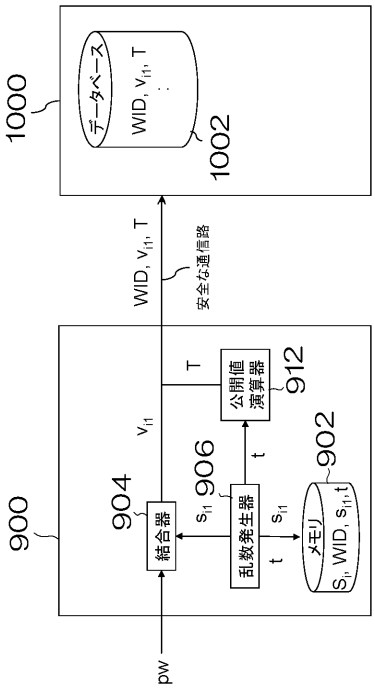
【図８】


802
WID
データベース
WID, W_ij
WID,W_ij
WID,
W_i(j+1)
秘密値更新器
InputMsg
828
826
セッション鍵生成器
SK_ij
806
エラーメッセージ発生器
No
812
U
第1ブラインド生成器
W_ij
808
y
乱数発生器
810
W_ij^y^r
816
マスク演算器
814
y
マスター秘密生成器
K
804
WID
判断部
Yes
Y
U
U^y
第2ブラインド生成器
818
822
入力メッセージ集合器
InputMsg
No
エラーメッセージ発生器
820
認証子判断部
Yes, InputMsg
InputMsg
824
認証子生成器
800
WID,X
S_i,Y
V_C
V_Si

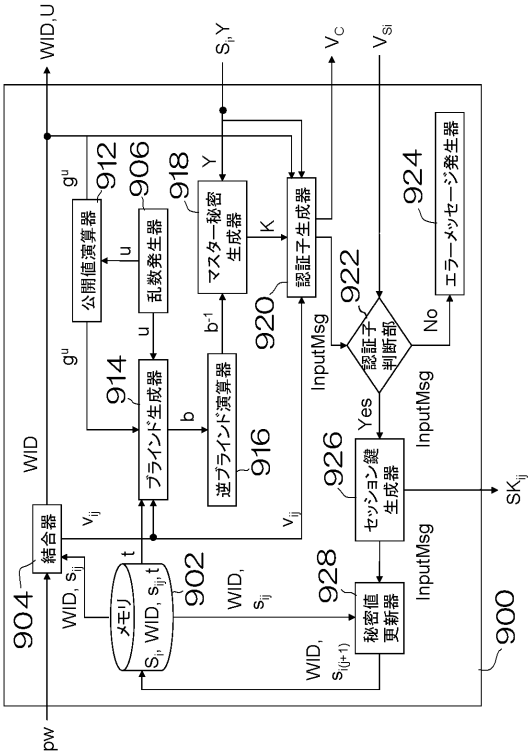
【図９】



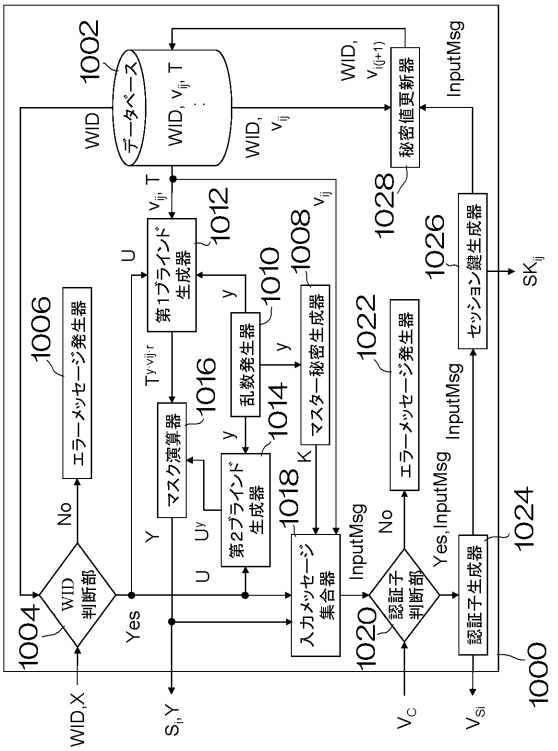
【図１０】



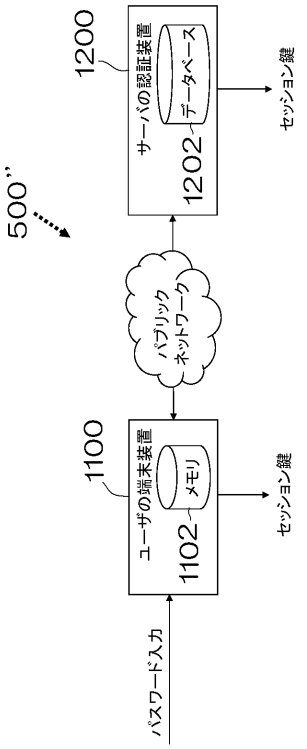
【図１１】



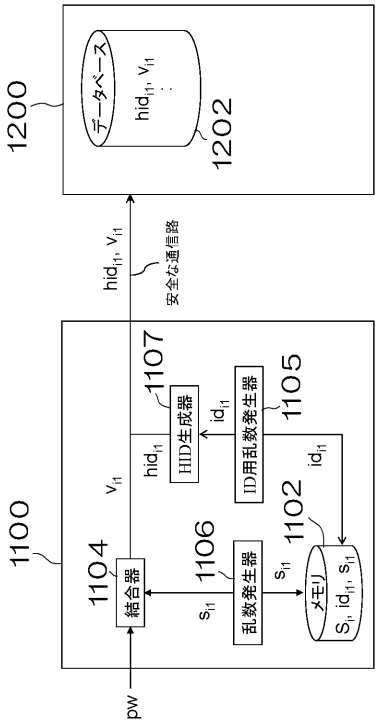
【図１２】



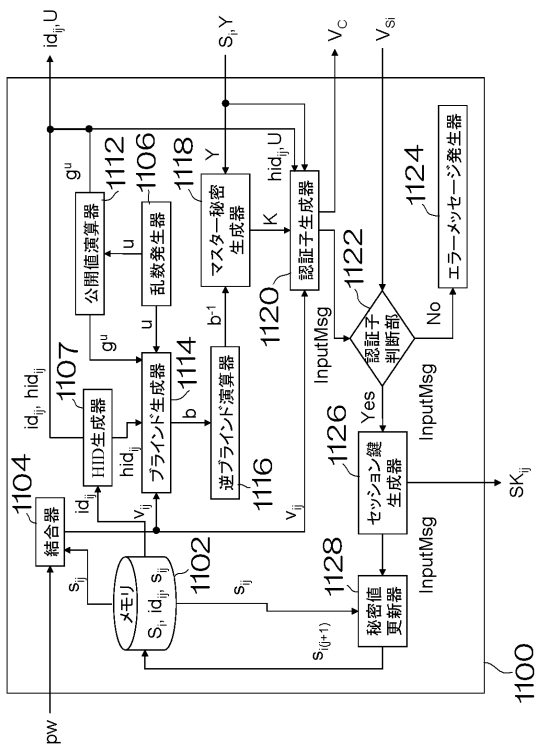
【図１３】



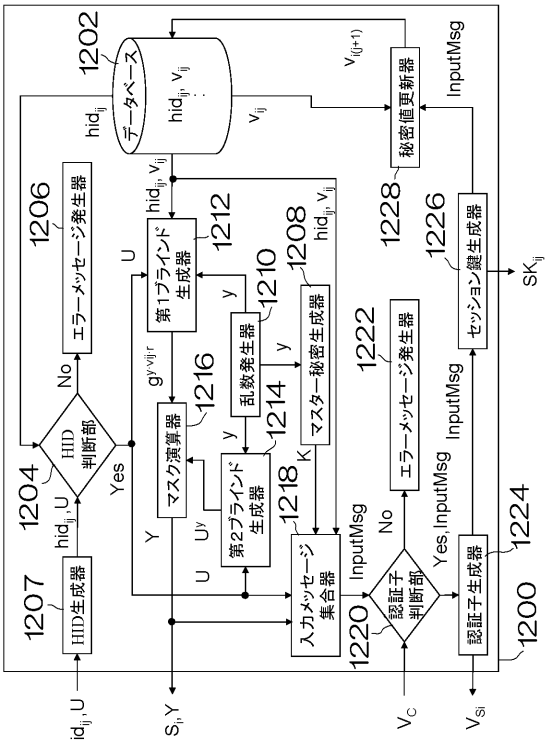
【図１４】



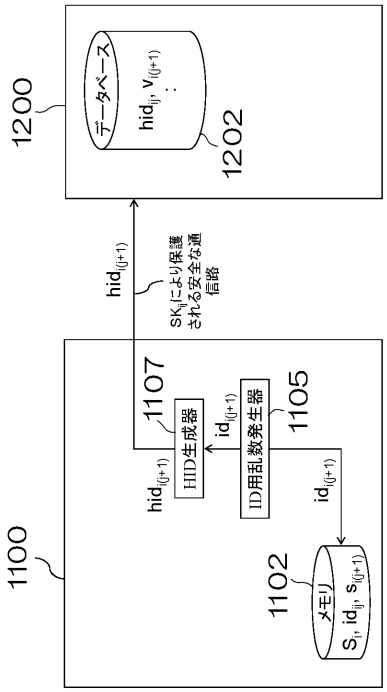
【図１５】



【図１６】



【図１７】


1200
データベース
hid_j, v_(j+1)
1202
hid_(j+1)
SK_jにより保護される安全な通信路
1100
1107
HID生成器
hid_(j+1)
id_(j+1)
ID用乱数発生器
1105
id_(j+1)
1102
メモリ
S_j, id_j, s_(j+1)

フロントページの続き

(72)発明者　辛　星漢
東京都千代田区霞が関１丁目３番１号　独立行政法人産業技術総合研究所内
(72)発明者　古原　和邦
東京都千代田区霞が関１丁目３番１号　独立行政法人産業技術総合研究所内

審査官　金沢　史明

(56)参考文献　国際公開第２００５／０４１４７４（WO，Ａ１）
特許第５０９９７７１（ＪＰ，Ｂ２）
特許第５００４０８６（ＪＰ，Ｂ２）
T. Kwon, Revision of AMP in IEEE P1363.2 and ISO/IEC 11770-4, Password-Based Public-Key Cryptography: Submissions and Research Contributions, IEEE P1363, ２００５年　６月８日, ［２０１２年１２月７日検索］, インターネット, , ＵＲＬ, http://grouper.ieee.org/groups/1363/passwdPK/contributions.html#Kwon05
辛星漢, 他, An Efficient Anonymous Password-Authenticated Key Exchange Protocol, 電子情報通信学会技術研究報告, 電子情報通信学会, ２００６年　７月１４日, vol. 106, no. 176, pp. 107-114
辛星漢, 他, On Anonymous Password-Authenticated Key Exchange, 電子情報通信学会技術研究報告, 電子情報通信学会, ２００７年　７月１２日, vol. 107, no. 140, pp. 145-151

(58)調査した分野(Int.Cl., ＤＢ名)
Ｈ０４Ｌ　　９／３２,　９／０８