

明 細 書

発明の名称： 認証装置、認証方法、及びプログラム

技術分野

[0001] 本開示は、認証装置、認証方法、及びプログラムに関する。

背景技術

[0002] 情報処理技術や通信技術の急速な発展に伴い、公文書、私文書を問わず、文書の電子化が急速に進んでいる。これに伴い、多くの個人や企業は、電子文書の安全管理に大きな関心を寄せている。こうした関心の高まりを受け、各方面で電子文書の盗聴や偽造等のタンパリング行為に対する安全性が盛んに議論されるようになってきた。電子文書の盗聴に対する安全性は、例えば、電子文書を暗号化することにより確保される。また、電子文書の偽造に対する安全性は、例えば、電子署名を利用することにより確保される。但し、暗号や電子署名には十分なタンパリング耐性が求められる。

[0003] 電子署名は、電子文書の作成者を特定するために利用される。そのため、電子署名は、電子文書の作成者しか生成できないようにすべきである。仮に、悪意ある第三者が同じ電子署名を生成できてしまうと、その第三者が電子文書の作成者に成りすますことができてしまう。つまり、悪意ある第三者により電子文書が偽造されてしまう。こうした偽造を防止するため、電子署名の安全性については様々な議論が交わされてきた。現在広く利用されている電子署名方式としては、例えば、RSA署名方式やDSA署名方式を利用する方式が知られている。

[0004] RSA署名方式は、「大きな合成数に対する素因数分解の困難性（以下、素因数分解問題）」を安全性の根拠とする。また、DSA署名方式は、「離散対数問題に対する解の導出の困難性」を安全性の根拠とする。これらの根拠は、古典的なコンピュータを利用して素因数分解問題や離散対数問題を効率的に解くアルゴリズムが存在しないことに起因する。つまり、上記の困難性は、古典的なコンピュータにおける計算量的な困難性を意味する。しかし

ながら、量子コンピュータを用いると、素因数分解問題や離散対数問題に対する解答が効率的に算出されてしまうと言われている。

[0005] 現在利用されている電子署名方式や公開鍵認証方式の多くは、RSA署名方式やDSA署名方式と同様、素因数分解問題や離散対数問題の困難性に安全性の根拠をおいている。そのため、こうした電子署名方式や公開鍵認証方式は、量子コンピュータが実用化された場合に、その安全性が確保されないことになる。そこで、素因数分解問題や離散対数問題など、量子コンピュータにより容易に解かれてしまう問題とは異なる問題に安全性の根拠をおく新たな電子署名方式及び公開鍵認証方式が求められている。量子コンピュータにより容易に解くことが難しい問題としては、例えば、多変数多項式に対する解答の困難性（以下、多変数多項式問題）がある。

[0006] その他にも、量子コンピュータにより解くことが困難とされている問題として、Syndrome Decoding問題、Constrained Linear Equation問題、Permuted Kernel問題、Permuted Perception問題、代数曲面上の求セクション問題などがある。

[0007] これらの問題のうち、代数曲面上の求セクション問題以外の問題はNP困難であることが知られている。その応用例として、例えば、下記の非特許文献1、2には、Syndrome Decoding問題に基づく公開鍵認証方式が開示されている。また、下記の非特許文献3には、Permuted Kernel問題に基づく公開鍵認証方式が開示されている。これら以外にも、Constrained Linear Equations問題に基づく公開鍵認証方式やPermuted Perceptions問題に基づく公開鍵認証方式などが提案されている。

先行技術文献

非特許文献

[0008] 非特許文献1: Jacques Stern, A New Identification Scheme Based on Syndrome D

ecoding, CRYPTO 1993, p13-21.

非特許文献2: Jacques Stern, A New Paradigm for Public Key Identification, IEEE Transactions on Information Theory, 1996, p13-21.

非特許文献3: Adi Shamir, An Efficient Identification Scheme Based on Permuted Kernels (Extended Abstract), CRYPTO 1989, p606-609.

発明の概要

発明が解決しようとする課題

[0009] ここで、公開鍵認証方式の安全性レベルについて言及しておきたい。公開鍵認証方式の安全性レベルには、2つの安全性レベルが存在する。その1つは、受動的攻撃 (passive attack) に対する安全性レベルである。もう1つは、能動的攻撃 (active attack) に対する安全性レベルである。受動的攻撃に対する安全性レベルは、正当なプロトコルに従う証明者と検証者の対話を盗聴することのみが可能な攻撃者を想定した安全性のレベルである。一方、能動的攻撃に対する安全性レベルは、攻撃者自身が直接的に証明者と対話プロトコルを実行することができる状況を想定した安全性のレベルである。つまり、能動的攻撃に対する安全性レベルは、攻撃者が好きなように証明者と対話プロトコルを行なえるということが想定されているのである。

[0010] 上記の非特許文献1～3に記載されているような従来の公開鍵認証方式は、受動的攻撃に対する安全性レベルを保証している。しかし、これら従来の公開鍵認証方式は、並列的繰り返し構成を行なった場合、能動的攻撃に対する安全性レベルを確実に保証しているか否かが知られていない。なぜならば、並列的繰り返し構成を行なった場合、零知識性は保存されないということが一般に知られているからである。そのため、並列的繰り返し構成において

能動的攻撃に対する安全性レベルを確実に保証するためには、さらに別の性質を保証する必要がある。

[0011] 従来の公開鍵認証方式は、1組の鍵ペア（公開鍵 y 、秘密鍵 s ）を用いて「 y について $y = F(s)$ となる s を知っていること」を証明者が検証者に対して証明する方式であった。そのため、検証で受理される対話を行った場合、「対話を行った証明者が s を用いた」という情報を検証者に知られてしまうことが避けられなかった。また、この形式の公開鍵認証方式の並列的繰り返し構成では、 F に衝突困難性の保証がない場合、能動的攻撃に対する安全性レベルを確実に保証しているか否かが知られていなかった。実際、前述の公開鍵認証方式で用いられている関数 F では、衝突困難性は保証されていない。

[0012] そこで、本開示は、上記問題に鑑みてなされたものであり、本開示の目的とするところは、関数 F に衝突困難性が保証されていなくても、対話プロトコルを並列的に繰り返し実行したとしても、能動的攻撃に対する安全性レベルを保証することが可能な、新規かつ改良された認証装置、認証方法、及びプログラムを提供することにある。

課題を解決するための手段

[0013] 上記課題を解決するために、本開示のある観点によれば、 L 個（ $L \geq 2$ ）の秘密鍵 s_i （ $i = 1 \sim L$ ）、及び n 次多変数多項式の組 F （ $n \geq 2$ ）について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持部と、（ $L - 1$ ）個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行する対話プロトコル実行部と、を備え、前記対話プロトコル実行部は、前記検証者との間で前記対話プロトコルを実行する際に、どの秘密鍵 s_i を使用したかを当該検証者に知られないようにする、認証装置が提供される。

[0014] また、前記対話プロトコル実行部は、前記検証者から L 個のチャレンジ $C h_i$ を受信するチャレンジ受信部と、前記チャレンジ受信部により受信された L 個のチャレンジ $C h_i$ の中から（ $L - 1$ ）個のチャレンジ $C h_i$ を任意に選

択するチャレンジ選択部と、前記秘密鍵 s_i を用いて、前記チャレンジ選択部により選択された $(L-1)$ 個のチャレンジ Ch_i のそれぞれに対する $(L-1)$ 個の回答 Rsp_i を生成する回答生成部と、前記回答生成部により生成された $(L-1)$ 個の回答 Rsp_i を前記検証者に送信する回答送信部と、を含んでいてもよい。

[0015] また、前記対話プロトコル実行部は、前記検証者に対し、 L 個の前記秘密鍵 s_i のそれぞれに対応するメッセージ Cmt_i を送信するメッセージ送信部をさらに含んでいてもよい。この場合、前記チャレンジ受信部は、前記メッセージ送信部により送信された各メッセージ Cmt_i に応じて、前記検証者により k 通り ($k \geq 2$) の検証パターンの中から選択された検証パターンを示すチャレンジ Ch_i を受信する。

[0016] また、前記メッセージ Cmt_i が $Cmt_i = (c_{i,1}, \dots, c_{i,N})$ である場合、前記メッセージ送信部は、一方向性関数 H を用いて新たなメッセージ $Cmt' = H(Cmt_1, \dots, Cmt_L)$ を算出して当該メッセージ Cmt' を前記検証者に送信し、前記回答送信部は、前記回答 Rsp_i と共に、当該回答 Rsp_i を利用しても前記検証者が復元できない前記メッセージ Cmt_i の要素を送信する、ように構成されていてもよい。

[0017] また、前記鍵保持部は、前記 L 個の秘密鍵 s_i のうち、1つの秘密鍵 s_{i_0} ($1 \leq i_0 \leq L$) を保持しないようにしてもよい。この場合、前記対話プロトコル実行部は、前記対話プロトコルの中で実行される前記秘密鍵 s_{i_0} に関する処理を偽証アルゴリズムに基づいて実行する。

[0018] また、上記課題を解決するために、本開示の別の観点によれば、 L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持部と、検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $Ch_i^{(j)}$ ($j = 1 \sim Q$) を受信するチャレンジ受信部と、前記チャレンジ受信部により受信された Q 組の L 個のチャレンジ $Ch_i^{(j)}$ の中から1組の L 個のチャレンジ $Ch_i^{(j)}$ を任意に選択するチャレンジ選択部と、前記秘密鍵 s_i を用いて、前記チャレンジ選択部により選択

された L 個のチャレンジ $Ch_i^{(j)}$ のそれぞれに対する L 個の回答 Rsp_i を生成する回答生成部と、前記回答生成部により生成された L 個の回答 Rsp_i を前記検証者に送信する回答送信部と、を備える、認証装置が提供される。

[0019] また、前記対話プロトコル実行部は、前記検証者に対し、 L 個の前記秘密鍵 s_i のそれぞれに対応するメッセージ Cmt_i を送信するメッセージ送信部をさらに含んでもよい。この場合、前記チャレンジ受信部は、前記メッセージ送信部により送信された各メッセージ Cmt_i に応じて、前記検証者により k 通り ($k \geq 2$) の検証パターンの中から選択された検証パターンを示すチャレンジ $Ch_i^{(j)}$ を受信する。

[0020] また、前記メッセージ Cmt_i が $Cmt_i = (c_{i,1}, \dots, c_{i,N})$ である場合、前記メッセージ送信部は、一方向性関数 H を用いて新たなメッセージ $Cmt' = H(Cmt_1, \dots, Cmt_L)$ を算出して当該メッセージ Cmt' を前記検証者に送信し、前記回答送信部は、前記回答 Rsp_i と共に、当該回答 Rsp_i を利用しても前記検証者が復元できない前記メッセージ Cmt_i の要素を送信する、ように構成されていてもよい。

[0021] また、上記課題を解決するために、本開示の別の観点によれば、 L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を生成する鍵生成ステップと、($L - 1$) 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行する対話プロトコル実行ステップと、を含み、前記対話プロトコル実行ステップでは、前記検証者との間で前記対話プロトコルを実行する際に、どの秘密鍵 s_i を使用したかを当該検証者に知られないようにする、認証方法が提供される。

[0022] また、上記課題を解決するために、本開示の別の観点によれば、 L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持機能と、($L - 1$) 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行する対話プロトコル実行機能

と、をコンピュータに実現させるためのプログラムであり、前記対話プロトコル実行機能は、前記検証者との間で前記対話プロトコルを実行する際に、どの秘密鍵 s_i を使用したかを当該検証者に知られないようにする、プログラムが提供される。

[0023] また、上記課題を解決するために、本開示の別の観点によれば、 L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を生成する鍵生成ステップと、検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $Ch_i^{(j)}$ ($j = 1 \sim Q$) を受信するチャレンジ受信ステップと、前記チャレンジ受信ステップで受信された Q 組の L 個のチャレンジ $Ch_i^{(j)}$ の中から 1 組の L 個のチャレンジ $Ch_i^{(j)}$ を任意に選択するチャレンジ選択ステップと、前記秘密鍵 s_i を用いて、前記チャレンジ選択ステップで選択された L 個のチャレンジ $Ch_i^{(j)}$ のそれぞれに対する L 個の回答 Rsp_i を生成する回答生成ステップと、前記回答生成ステップで生成された L 個の回答 Rsp_i を前記検証者に送信する回答送信ステップと、を含む、認証方法が提供される。

[0024] また、上記課題を解決するために、本開示の別の観点によれば、 L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持機能と、検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $Ch_i^{(j)}$ ($j = 1 \sim Q$) を受信するチャレンジ受信機能と、前記チャレンジ受信機能により受信された Q 組の L 個のチャレンジ $Ch_i^{(j)}$ の中から 1 組の L 個のチャレンジ $Ch_i^{(j)}$ を任意に選択するチャレンジ選択機能と、前記秘密鍵 s_i を用いて、前記チャレンジ選択機能により選択された L 個のチャレンジ $Ch_i^{(j)}$ のそれぞれに対する L 個の回答 Rsp_i を生成する回答生成機能と、前記回答生成機能により生成された L 個の回答 Rsp_i を前記検証者に送信する回答送信機能と、をコンピュータに実現させるためのプログラムが提供される。また、上記課題を解決するために、本開示の別の観点によれば、上記のプログラムが記録された、コンピュータにより読み取り可能な記録媒体が提供される。

発明の効果

[0025] 以上説明したように本開示によれば、対話プロトコルを並列的に繰り返し実行したとしても、能動的攻撃に対する安全性レベルを保証することが可能になる。

図面の簡単な説明

[0026] [図1]公開鍵認証方式のアルゴリズム構成について説明するための説明図である。

[図2]ハッシュ関数の公開鍵認証方式について説明するための説明図である。

[図3]SSH 1.0 a 公開鍵認証方式の対話プロトコルについて説明するための説明図である。

[図4]SSH 1.0 b 公開鍵認証方式の対話プロトコルについて説明するための説明図である。

[図5]対話プロトコルの直列的繰り返し構成について説明するための説明図である。

[図6]対話プロトコルの並列的繰り返し構成について説明するための説明図である。

[図7]SSH 1.0 a 公開鍵認証方式の対話プロトコルに対する偽証明アルゴリズムについて説明するための説明図である。

[図8]SSH 1.0 b 公開鍵認証方式の対話プロトコルに対する偽証明アルゴリズムについて説明するための説明図である。

[図9]SSH 1.0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法について説明するための説明図である。

[図10]SSH 1.0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法 (変形例 1) について説明するための説明図である。

[図11]SSH 1.0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法 (変形例 2) について説明するための説明図である。

[図12]SSH 1.0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法 (変形例 3) について説明するための説明図である。

[図13] S S H 1 0 b 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法について説明するための説明図である。

[図14] S S H 1 0 b 公開鍵認証方式の対話プロトコルに対する本手法 # 1 の適用方法（変形例）について説明するための説明図である。

[図15] S S H 1 0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 2 の適用方法について説明するための説明図である。

[図16] S S H 1 0 a 公開鍵認証方式の対話プロトコルに対する本手法 # 2 の適用方法（変形例）について説明するための説明図である。

[図17] S S H 1 0 b 公開鍵認証方式の対話プロトコルに対する本手法 # 2 の適用方法について説明するための説明図である。

[図18] S S H 1 0 b 公開鍵認証方式の対話プロトコルに対する本手法 # 2 の適用方法（変形例）について説明するための説明図である。

[図19] S S H 1 0 a 公開鍵認証方式の対話プロトコルにおける通信量の削減方法について説明するための説明図である。

[図20] S S H 1 0 b 公開鍵認証方式の対話プロトコルにおける通信量の削減方法について説明するための説明図である。

[図21] 本実施形態に係る対話プロトコルを実現することが可能な情報処理装置のハードウェア構成例について説明するための説明図である。

発明を実施するための形態

[0027] 以下に添付図面を参照しながら、本開示の好適な実施の形態について詳細に説明する。なお、本明細書及び図面において、実質的に同一の機能構成を有する構成要素については、同一の符号を付することにより重複説明を省略する。

[0028] [説明の流れについて]

ここで、以下に記載する本開示の実施形態に関する説明の流れについて簡単に述べる。まず、図 1 を参照しながら、公開鍵認証方式のアルゴリズム構成について説明する。次いで、図 2 を参照しながら、n パスの公開鍵認証方式について説明する。次いで、図 3 を参照しながら、S S H 1 0 a 公開鍵認

証方式の対話プロトコルについて説明する。次いで、図4を参照しながら、SSH10b公開鍵認証方式の対話プロトコルについて説明する。次いで、図5、図6を参照しながら、対話プロトコルの繰り返し構成について説明する。この中で、能動的攻撃に対する安全性レベルについて簡単に説明する。

[0029] 次いで、図7を参照しながら、SSH10a公開鍵認証方式の対話プロトコルに対する偽証アルゴリズムについて説明する。次いで、図8を参照しながら、SSH10b公開鍵認証方式の対話プロトコルに対する偽証アルゴリズムについて説明する。次いで、図9～図12を参照しながら、本開示の第1実施形態に係る手法（本手法#1）をSSH10a公開鍵認証方式の対話プロトコルに適用する方法について説明する。次いで、図13、図14を参照しながら、本手法#1をSSH10b公開鍵認証方式の対話プロトコルに適用する方法について説明する。

[0030] 次いで、図15、図16を参照しながら、本開示の第2実施形態に係る手法（本手法#2）をSSH10a公開鍵認証方式の対話プロトコルに適用する方法について説明する。次いで、図17、図18を参照しながら、本手法#2をSSH10b公開鍵認証方式の対話プロトコルに適用する方法について説明する。次いで、図19、図20を参照しながら、本実施形態に係る対話プロトコルにおける通信量の削減方法について説明する。次いで、図21を参照しながら、本実施形態に係る対話プロトコルを実現することが可能な情報処理装置のハードウェア構成例について説明する。最後に、同実施形態の技術的思想について纏め、当該技術的思想から得られる作用効果について簡単に説明する。

[0031] （説明項目）

1：はじめに

1-1：公開鍵認証方式のアルゴリズム構成

1-2：nパスの公開鍵認証方式について

1-3：SSH10a公開鍵認証方式の対話プロトコル

1-4：SSH10b公開鍵認証方式の対話プロトコル

- 1-5 : 対話プロトコルの繰り返し構成
- 1-6 : SSH 1.0 a 公開鍵認証方式に対する偽証アルゴリズム
- 1-7 : SSH 1.0 b 公開鍵認証方式に対する偽証アルゴリズム
- 2 : 第1実施形態（本手法#1）
 - 2-1 : 概要
 - 2-2 : SSH 1.0 a 公開鍵認証方式への適用
 - 2-3 : SSH 1.0 a 公開鍵認証方式への適用（変形例1）
 - 2-4 : SSH 1.0 a 公開鍵認証方式への適用（変形例2）
 - 2-5 : SSH 1.0 a 公開鍵認証方式への適用（変形例3）
 - 2-6 : SSH 1.0 b 公開鍵認証方式への適用
 - 2-7 : SSH 1.0 b 公開鍵認証方式への適用（変形例）
- 3 : 第2実施形態（本手法#2）
 - 3-1 : 概要
 - 3-2 : SSH 1.0 a 公開鍵認証方式への適用
 - 3-3 : SSH 1.0 a 公開鍵認証方式への適用（変形例）
 - 3-4 : SSH 1.0 b 公開鍵認証方式への適用
 - 3-5 : SSH 1.0 b 公開鍵認証方式への適用（変形例）
- 4 : 補足
 - 4-1 : 方式の拡張について
 - 4-2 : 非対話型の公開鍵認証方式について
 - 4-3 : 通信量の削減方法について
- 5 : ハードウェア構成
- 6 : まとめ

[0032] <1 : はじめに>

はじめに、本開示に係る実施形態について詳細に説明するに先立ち、一般的な公開鍵認証方式のアルゴリズム構成、及びnパスの公開鍵認証方式について簡単に説明する。

[0033] [1-1 : 公開鍵認証方式のアルゴリズム構成]

まず、図 1 を参照しながら、公開鍵認証方式のアルゴリズム構成について説明する。図 1 は、公開鍵認証方式のアルゴリズム構成について説明するための説明図である。

[0034] (概要)

公開鍵認証方式とは、ある人（証明者）が、公開鍵 p_k 及び秘密鍵 s_k を利用して、他の人（検証者）に本人であることを納得させるための認証方式である。例えば、証明者 A の公開鍵 p_{k_A} は、検証者に公開される。一方、証明者 A の秘密鍵 s_{k_A} は、証明者により秘密に管理される。公開鍵認証方式では、公開鍵 p_{k_A} に対応する秘密鍵 s_{k_A} を知る者が証明者 A 本人であるとみなされる。

[0035] 証明者 A が検証者 B に対して本人であることを証明しようとする場合、証明者 A は、検証者 B と対話プロトコルを実行して、自身が公開鍵 p_{k_A} に対応する秘密鍵 s_{k_A} を知っていることを証明すればよい。そして、対話プロトコルにより証明者 A が秘密鍵 s_{k_A} を知っていることが検証者 B により証明された場合、証明者 A の正当性（本人であること）が証明される。

[0036] なお、公開鍵認証方式の安全性を確保するには、次に示す 2 つの条件が求められる。

[0037] 1 つ目の条件は、対話プロトコルを実行した際に秘密鍵 s_k を持たない偽証者により偽証が成立してしまう確率を限りなく小さくすることである。この 1 つ目の条件が成り立つことを「健全性」と呼ぶ。つまり、健全性を有する対話プロトコルにおいては、秘密鍵 s_k を持たない偽証者により、無視できない確率で偽証が成立することはないと言い換えられる。2 つ目の条件は、対話プロトコルを実行したとしても、証明者 A が有する秘密鍵 s_{k_A} の情報が検証者 B に一切漏れることがないようにすることである。この 2 つ目の条件が成り立つことを「零知識性」と呼ぶ。

[0038] 上記の健全性と零知識性を有する対話プロトコルを利用することにより、公開鍵認証方式の安全性が確保される。

[0039] (モデル)

公開鍵認証方式のモデルには、図 1 に示すように、証明者と検証者という 2 つのエンティティが存在する。証明者は、鍵生成アルゴリズム G_{en} を用いて、証明者固有の秘密鍵 s_k と公開鍵 p_k の組を生成する。次いで、証明者は、鍵生成アルゴリズム G_{en} を用いて生成した秘密鍵 s_k と公開鍵 p_k の組を利用して検証者と対話プロトコルを実行する。このとき、証明者は、証明者アルゴリズム P を利用して対話プロトコルを実行する。上記の通り、対話プロトコルにおいて、証明者は、証明者アルゴリズム P を利用して、秘密鍵 s_k を保有していることを検証者に証明する。

[0040] 一方、検証者は、検証者アルゴリズム V を利用して対話プロトコルを実行し、証明者が公開している公開鍵に対応する秘密鍵を、その証明者が保有しているか否かを検証する。つまり、検証者は、証明者が公開鍵に対応する秘密鍵を保有しているか否かを検証するエンティティである。このように、公開鍵認証方式のモデルは、証明者と検証者という 2 つのエンティティ、及び、鍵生成アルゴリズム G_{en} 、証明者アルゴリズム P 、検証者アルゴリズム V という 3 つのアルゴリズムにより構成される。

[0041] なお、以下の説明において、「証明者」「検証者」という表現を用いるが、これらの表現はあくまでもエンティティを意味するものである。従って、鍵生成アルゴリズム G_{en} 、証明者アルゴリズム P を実行する主体は、「証明者」のエンティティに対応する情報処理装置である。同様に、検証者アルゴリズム V を実行する主体は、情報処理装置である。これら情報処理装置のハードウェア構成は、例えば、図 2 1 に示した通りである。つまり、鍵生成アルゴリズム G_{en} 、証明者アルゴリズム P 、検証者アルゴリズム V は、ROM 904、RAM 906、記憶部 920、リムーバブル記録媒体 928 などに記録されたプログラムに基づいて CPU 902 により実行される。

[0042] (鍵生成アルゴリズム G_{en})

鍵生成アルゴリズム G_{en} は、証明者により利用される。そして、鍵生成アルゴリズム G_{en} は、証明者に固有の秘密鍵 s_k と公開鍵 p_k の組を生成するアルゴリズムである。鍵生成アルゴリズム G_{en} により生成された公開

鍵 p_k は公開される。そして、公開された公開鍵 p_k は、検証者により利用される。一方、鍵生成アルゴリズム Gen により生成された秘密鍵 s_k は、証明者が秘密に管理する。そして、秘密に管理される秘密鍵 s_k は、検証者に対して公開鍵 p_k に対応する秘密鍵 s_k を保有していることを証明するために利用される。形式的に、鍵生成アルゴリズム Gen は、セキュリティパラメータ 1^λ (λ は 0 以上の整数) を入力とし、秘密鍵 s_k と公開鍵 p_k を出力するアルゴリズムとして、下記の式 (1) のように表現される。

[0043] [数1]

$$(sk, pk) \leftarrow Gen(1^\lambda) \\ \dots (1)$$

[0044] (証明者アルゴリズム P)

証明者アルゴリズム P は、証明者により利用される。そして、証明者アルゴリズム P は、公開鍵 p_k に対応する秘密鍵 s_k を保有していることを証明するアルゴリズムである。証明者アルゴリズム P は、証明者の秘密鍵 s_k と公開鍵 p_k を入力とし、検証者との対話プロトコルを実行するアルゴリズムとして定義される。

[0045] (検証者アルゴリズム V)

検証者アルゴリズム V は、検証者により利用される。そして、検証者アルゴリズム V は、対話プロトコルの中で、公開鍵 p_k に対応する秘密鍵 s_k を証明者が保有しているか否かを検証するアルゴリズムである。検証者アルゴリズム V は、証明者の公開鍵 p_k を入力とし、証明者との間で対話プロトコルを実行した後、0 又は 1 (1 b i t) を出力するアルゴリズムとして定義される。なお、出力 0 の場合には証明者が不正なものであるとし、出力 1 の場合には証明者が正当なものであるとする。

[0046] (補足)

上記の通り、公開鍵認証方式は、安全性を確保するため、健全性と零知識

性という2つの条件を満たすことが求められる。しかし、証明者が秘密鍵 s_k を保有していることを証明者に証明させるためには、証明者が秘密鍵 s_k に依存した手続きを実行し、その結果を検証者に通知して、その通知内容に基づく検証を検証者に実行させる必要がある。秘密鍵 s_k に依存した手続きを実行するのは、健全性を担保するために必要である。一方で、この手続きの結果を検証者に通知しても、秘密鍵 s_k の情報が一切検証者に漏れないようにする必要がある。そのため、これらの要件を満たすように、上記の鍵生成アルゴリズム G_{en} 、証明者アルゴリズム P 、検証者アルゴリズム V が設計される。

[0047] 以上、公開鍵認証方式のアルゴリズム構成について説明した。

[0048] [1 - 2 : n パスの公開鍵認証方式について]

次に、図2を参照しながら、 n パスの公開鍵認証方式について説明する。

図2は、 n パスの公開鍵認証方式について説明するための説明図である。

[0049] 上記の通り、公開鍵認証方式は、対話プロトコルの中で、証明者が公開鍵 p_k に対応する秘密鍵 s_k を保有していることを検証者に証明する認証方式である。また、公開鍵認証方式の安全性を担保するため、健全性と零知識性という2つの条件を満たす必要がある。そのため、対話プロトコルの中では、図2に示すように、証明者と検証者の双方がそれぞれ処理を実行しながら、証明者と検証者との間で n 回の情報交換が行われる。

[0050] n パスの公開鍵認証方式の場合、証明者アルゴリズム P を用いて証明者により処理 (Step. 1) が実行され、情報 T_1 が検証者に送信される。次いで、検証者アルゴリズム V を用いて検証者により処理 (Step. 2) が実行され、情報 T_2 が証明者に送信される。同様にして処理 (Step. 3) ~ (Step. n) が実行され、情報 $T_3, \dots, T_n$ が送信され、処理 (Step. $n+1$) が実行される。このように、情報が n 回送受信される対話プロトコルに基づく公開鍵認証方式のことを「 n パス」の公開鍵認証方式と呼ぶ。

[0051] 以上、 n パスの公開鍵認証方式について説明した。

[0052] [1-3: SSH10a 公開鍵認証方式の対話プロトコル]

次に、図3を参照しながら、SSH10a 公開鍵認証方式の対話プロトコルについて説明する。図3は、SSH10a 公開鍵認証方式の対話プロトコルについて説明するための説明図である。なお、SSH10a 公開鍵認証方式とは、本件発明者（作本、白井、樋渡）により考案された多次多変数連立方程式の求解問題に基づく公開鍵認証方式の一つである。また、このSSH10a 公開鍵認証方式は、3パスの公開鍵認証方式の一例である。

[0053] なお、多次多変数連立方程式の求解問題とは、環 K 上の m 本の n 変数多次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、ベクトル $y \in K^m$ が与えられたとき、 $(f_1(s_1, \dots, s_n), \dots, f_m(s_1, \dots, s_n)) = y$ となるベクトル $(s_1, \dots, s_n) \in K^n$ を求める問題である。2次以上の多変数連立方程式の求解問題は、NP困難問題と呼ばれ、解くことが非常に困難な問題のクラスに属する。

[0054] さて、SSH10a 公開鍵認証方式の対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0055] (鍵生成アルゴリズム Gen)

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数2次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、ベクトル $s = (s_1, \dots, s_n) \in K^n$ を生成する。次に、鍵生成アルゴリズム Gen は、 $y = (y_1, \dots, y_m) \leftarrow (f_1(s), \dots, f_m(s))$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(f_1, \dots, f_m, y)$ を公開鍵 pk に設定し、 s を秘密鍵に設定する。なお、以下では、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数2次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。

[0056] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図3を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム

Vの構成について説明する。SSH10a公開鍵認証方式の対話プロトコルは、「証明者が $y = F(s)$ を満たす s を知っていること」を s の情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム G_{en} により生成された公開鍵 p_k は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム G_{en} により生成された秘密鍵 s_k は、証明者により秘密に管理されているものとする。

[0057]

ここで、2次多項式の性質について言及しておきたい。

[0058]

m 本の n 変数2次多項式の組 $(f_1(x), \dots, f_m(x))$ は、下記の式

(2) のように表現することができる。但し、 $x = (x_1, \dots, x_n)$ は、 n

個の変数を表すベクトルである。また、 $A_1, \dots, A_m$ は、 $n \times n$ 行列である

。さらに、 $b_1, \dots, b_m$ は $n \times 1$ ベクトルである。そして、 c は、 $m \times 1$ ベ

クトルである。

[0059] [数2]

$$F(x) = \begin{pmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{pmatrix} = \begin{pmatrix} x^T A_1 x + b_1^T x \\ \vdots \\ x^T A_m x + b_m^T x \end{pmatrix} + c \quad \dots (2)$$

[0060] この表現を用いると、多項式の組 F は、下記の式(3)及び式(4)のよ

うに表現することができる。この表現が成り立つことは、下記の式(5)か

ら容易に確認することができる。

[0061] [数3]

$$F(x_1 + x_2) = F(x_1) + F(x_2) + F_b(x_1, x_2) - c \quad \dots (3)$$

$$F_b(x_1, x_2) = \begin{pmatrix} x_2^T (A_1^T + A_1) x_1 \\ \vdots \\ x_2^T (A_m^T + A_m) x_1 \end{pmatrix}$$

… (4)

$$\begin{aligned} f_l(x_1 + x_2) &= (x_1 + x_2)^T A_l (x_1 + x_2) + b_l^T (x_1 + x_2) + c_l \\ &= x_1^T A_l x_1 + x_1^T A_l x_2 + x_2^T A_l x_1 + x_2^T A_l x_2 + b_l^T x_1 + b_l^T x_2 + c_l \\ &= f_l(x_1) + f_l(x_2) - c_l + x_1^T A_l x_2 + x_2^T A_l x_1 \\ &= f_l(x_1) + f_l(x_2) - c_l + x_1^T (A_l^T)^T x_2 + x_2^T A_l x_1 \\ &= f_l(x_1) + f_l(x_2) - c_l + (A_l^T x_1)^T x_2 + x_2^T A_l x_1 \\ &= f_l(x_1) + f_l(x_2) - c_l + x_2^T (A_l^T x_1) + x_2^T A_l x_1 \\ &= f_l(x_1) + x_2^T (A_l^T + A_l) x_1 + f_l(x_2) - c_l \end{aligned}$$

… (5)

[0062] このように $F(x_1 + x_2)$ を x_1 に依存する部分と、 x_2 に依存する部分と、 x_1 、 x_2 の両方に依存する部分の3つに分けたとき、 x_1 と x_2 の両方に依存する部分 $F_b(x_1, x_2)$ は、 x_1 、 x_2 について双線形写像になる。なお、以下で説明するSSH10a公開鍵認証方式は、上記のような2次多項式の性質を利用したものである。

[0063] 再びSSH10a公開鍵認証方式の対話プロトコル(図3を参照)における証明者アルゴリズムP、検証者アルゴリズムVの説明に戻る。

[0064] Step. 1 :

まず、証明者アルゴリズムPは、任意に数 w を選択する。次いで、証明者アルゴリズムPは、擬似乱数生成器 G_1 に数 w を適用してベクトル $r \in K^n$ と数 w' を生成する。つまり、証明者アルゴリズムPは、 $(r, w') \leftarrow G_1(w)$

w) を計算する。次いで、証明者アルゴリズム P は、擬似乱数生成器 G_2 に数 w' を適用して 2 つのベクトル $t \in K^n$ 、 $e \in K^m$ を生成する。つまり、証明者アルゴリズム P は、 $(t, e) \leftarrow G_2(w')$ を計算する。次いで、証明者アルゴリズム P は、 $z \leftarrow s - r$ を計算する。この計算は、秘密鍵 s をベクトル r によりマスクする操作に相当する。さらに、証明者アルゴリズム P は、 $t' \leftarrow r + t$ を計算する。次いで、証明者アルゴリズム P は、 $e' \leftarrow F(r) - c + e$ を計算する。

[0065] 次いで、証明者アルゴリズム P は、上記の式 (3) 及び式 (4) に示した関数 F_b の定義に基づいて $F_b(z, t)$ を算出し、 $F_b(z, t) + e$ と z のハッシュ値 c_1 を生成する。つまり、証明者アルゴリズム P は、 $c_1 \leftarrow H_1(F_b(z, t) + e, z)$ を計算する。また、証明者アルゴリズム P は、数 w' のハッシュ値 c_2 を生成する。つまり、証明者アルゴリズム P は、 $c_2 \leftarrow H_2(w')$ を計算する。さらに、証明者アルゴリズム P は、2 つのベクトル t' と e' のハッシュ値 c_3 を生成する。つまり、証明者アルゴリズム P は、 $c_3 \leftarrow H_3(t', e')$ を計算する。

[0066] 次いで、証明者アルゴリズム P は、 $St \leftarrow (F, y, s, r, t, e, z, t', e')$ 及び $Cmt \leftarrow (c_1, c_2, c_3)$ を設定する。そして、Step. 1 で生成された Cmt は、検証者（検証者アルゴリズム V）に送られる。なお、上記の $H_1(\dots)$ 、 $H_2(\dots)$ 、 $H_3(\dots)$ は、ハッシュ関数である。また、Step. 1 における上記の操作を $(Cmt; St) \leftarrow P_{a,1}(F, y, s; r, t, e)$ と表現することにする。

[0067] Step. 2 :

Cmt を受け取った検証者アルゴリズム V は、3 つの検証パターンのうち、どの検証パターンを利用するかを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジ $Ch \in_R \{0, 1, 2\}$ を証明者（証明者アルゴリズム P）に送る。

[0068] Step. 3 :

Ch を受け取った証明者アルゴリズム P は、検証者アルゴリズム V から受

け取ったチャレンジ C_h に応じて検証者アルゴリズム V へと送り返す回答 R_{sp} を生成する。もし $C_h = 0$ の場合、証明者アルゴリズム P は、回答 $R_{sp} \leftarrow (r, t, e)$ を生成する。また、 $C_h = 1$ の場合、証明者アルゴリズム P は、回答 $R_{sp} = (z, t, e)$ を生成する。そして、 $C_h = 2$ の場合、証明者アルゴリズム P は、回答 $R_{sp} = (z, t', e')$ を生成する。なお、Step. 3 における上記の操作を $R_{sp} \leftarrow P_{a,2}(C_h; St)$ と表現することにする。また、Step. 3 で生成された R_{sp} は、検証者（検証者アルゴリズム V ）に送られる。

[0069] Step. 4 :

R_{sp} を受け取った検証者アルゴリズム V は、受け取った R_{sp} に対して次のいずれかの検証を実行する。

[0070] $C_h = 0$ の場合、検証者アルゴリズム V は、 $(r'', t'', e'') \leftarrow R_{sp}$ を実行する。次いで、検証者アルゴリズム V は、 $c_2 = H_2(t'', e'')$ 及び $c_3 = H_3(r'' + t'', F(r'') - c + e'')$ が成り立つか否かを検証する。

$C_h = 1$ の場合、検証者アルゴリズム V は、 $(z'', t'', e'') \leftarrow R_{sp}$ を実行する。次いで、検証者アルゴリズム V は、 $c_1 = H_1(F_b(z'', t'') + e'', z'')$ 及び $c_2 = H_2(t'', e'')$ が成り立つか否かを検証する。

$C_h = 2$ の場合、検証者アルゴリズム V は、 $(z'', t''', e''') \leftarrow R_{sp}$ を実行する。次いで、検証者アルゴリズム V は、 $c_1 = H_1(F(z'') + F_b(z'', t''') + e''' - y, z'')$ 及び $c_3 = H_3(t''', e''')$ が成り立つか否かを検証する。

[0071] なお、Step. 4 における検証の操作を $0/1 \leftarrow Dec_a(F, y; Cmt, C_h, R_{sp})$ と表現することにする。この操作において、出力 1 は検証の成功を示し、出力 0 は検証の失敗を示すものとする。

[0072] 以上、SSH10a 公開鍵認証方式における証明者アルゴリズム P 及び検証者アルゴリズム V の処理内容について説明した。なお、上記の方式ではハ

ハッシュ関数 H_1, H_2, H_3 を用いて c_1, c_2, c_3 を計算しているが、ハッシュ関数 H_1, H_2, H_3 の代わりにコミットメント関数 COM を用いてもよい。また、本稿を通して、ハッシュ関数を利用している箇所をコミットメント関数 COM に置き換えてもよい。

[0073] コミットメント関数 COM は、文字列 S と乱数 p の 2 つを引数にとる関数である。コミットメント関数の例としては、 $Sha1$ 、 $Ha1ev1$ と $S1$ 、 $lv10$ 、 $Micali1$ によって国際会議 $Crypt01996$ で発表された方式などがある。コミットメント関数を用いる場合、 c_1, c_2, c_3 を計算する前に乱数 p_1, p_2, p_3 を用意し、ハッシュ関数 $H_1(\cdot), H_2(\cdot), H_3(\cdot)$ を適用する代わりにコミットメント関数 $COM(\cdot, p_1), CO$
 $M(\cdot, p_2), COM(\cdot, p_3)$ を適用して c_1, c_2, c_3 を生成する。また、 p_i は回答に含めて証明者 A から検証者 V に送られる。

[0074] [1-4: $SSH10b$ 公開鍵認証方式の対話プロトコル]

次に、図 4 を参照しながら、 $SSH10b$ 公開鍵認証方式の対話プロトコルについて説明する。図 4 は、 $SSH10b$ 公開鍵認証方式の対話プロトコルについて説明するための説明図である。なお、 $SSH10b$ 公開鍵認証方式とは、本件発明者（作本、白井、極渡）により考案された多次多変数連立方程式の求解問題に基づく公開鍵認証方式の一つである。また、この $SSH10b$ 公開鍵認証方式は、5 パスの公開鍵認証方式の一例である。

[0075] $SSH10b$ 公開鍵認証方式の対話プロトコルは、 $SSH10a$ 公開鍵認証方式の対話プロトコルと同様に、鍵生成アルゴリズム G_{en} 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0076] (鍵生成アルゴリズム G_{en})

まず、鍵生成アルゴリズム G_{en} の構成について説明する。鍵生成アルゴリズム G_{en} は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n)$ は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n) \in K$

n を生成する。次に、鍵生成アルゴリズム Gen は、 $y = (y_1, \dots, y_m) \leftarrow (f_1(s), \dots, f_m(s))$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(f_1, \dots, f_m, y)$ を公開鍵 pk に設定し、 s を秘密鍵に設定する。なお、以下では、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。

[0077] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図 4 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。SSH10b 公開鍵認証方式の対話プロトコルは、「証明者が $y = F(s)$ を満たす s を知っていること」を s の情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0078] SSH10b 公開鍵認証方式の対話プロトコルは、図 4 に示す Step. 1 ~ Step. 6 の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0079] Step. 1 :

まず、証明者アルゴリズム P は、任意に数 w を選択する。次いで、証明者アルゴリズム P は、擬似乱数生成器 G に数 w を適用してベクトル $r \in K^n$ 、 $t \in K^n$ 、 $e \in K^m$ を生成する。つまり、証明者アルゴリズム P は、 $(r, t, e) \leftarrow G(w)$ を計算する。次いで、証明者アルゴリズム P は、 $z \leftarrow s - r$ を計算する。この計算は、秘密鍵 s をベクトル r によりマスクする操作に相当する。

[0080] 次いで、証明者アルゴリズム P は、 $F_b(z, t) + e$ と z のハッシュ値 c_1 を生成する。つまり、証明者アルゴリズム P は、 $c_1 \leftarrow H_1(F_b(z, t) + e, z)$ を計算する。また、証明者アルゴリズム P は、ベクトル r 、 t 、 e のハッシュ値 c_2 を生成する。つまり、証明者アルゴリズム P は、 $c_2 \leftarrow H_2$

(r, t, e) を計算する。なお、上記の $H_1(\dots)$ 、 $H_2(\dots)$ は、ハッシュ関数である。

[0081] 次いで、証明者アルゴリズム P は、 $St_A \leftarrow (F, y, s, r, t, e, z)$ 及び $Cmt_A \leftarrow (c_1, c_2)$ を設定する。そして、Step. 1 で生成された Cmt_A は、検証者（検証者アルゴリズム V ）に送られる。なお、上記の $H_1(\dots)$ 、 $H_2(\dots)$ は、ハッシュ関数である。また、Step. 1 における上記の操作を $(Cmt_A; St_A) \leftarrow P_{b,1}(F, y, s; r, t, e)$ と表現することにする。

[0082] Step. 2 :

Cmt_A を受け取った検証者アルゴリズム V は、 q 通り存在する環 K の元から 1 つの乱数 α を選択する。そして、検証者アルゴリズム V は、チャレンジ $Ch_A = \alpha$ を証明者（証明者アルゴリズム P ）に送る。

[0083] Step. 3 :

Ch_A を受け取った証明者アルゴリズム P は、 $t' \leftarrow \alpha r + t$ を計算する。さらに、証明者アルゴリズム P は、 $e' \leftarrow \alpha (F(r) - c) + e$ を計算する。そして、証明者アルゴリズム P は、 $St_B \leftarrow (St_A, Ch_A, t', e')$ 及び $Cmt_B \leftarrow (t', e')$ を設定する。また、証明者アルゴリズム P は、 Cmt_B を検証者（検証者アルゴリズム V ）に送る。なお、Step. 3 における上記の操作を $(Cmt_B; St_B) \leftarrow P_{b,2}(Ch_A; St_A)$ と表現することにする。

[0084] Step. 4 :

Cmt_B を受け取った検証者アルゴリズム V は、2 つの検証パターンのうち、どちらの検証パターンを利用するかを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジ $Ch_B \in_R \{0, 1\}$ を証明者（証明者アルゴリズム P ）に送る。

[0085] Step. 5 :

Ch_B を受け取った証明者アルゴリズム P は、検証者アルゴリズム V から受けたチャレンジ Ch_B に応じて検証者（検証者アルゴリズム V ）に送り返す回

答 Rsp を次のようにして生成する。 $Ch_B = 0$ の場合、証明者アルゴリズム P は、 $Rsp \leftarrow r$ に設定する。 $Ch_B = 1$ の場合、証明者アルゴリズム P は、 $Rsp \leftarrow z$ に設定する。そして、証明者アルゴリズム P は、回答 Rsp を検証者（検証者アルゴリズム V ）に送る。なお、Step. 5 における上記の操作を $Rsp \leftarrow P_{b,3}(Ch_B; St_B)$ と表現することにする。

[0086] Step. 6 :

。 回答 Rsp を受け取った検証者アルゴリズム V は、証明者（証明者アルゴリズム P ）から受け取った回答 Rsp を利用して以下の検証処理を実行する。

[0087] $Ch_B = 0$ の場合、検証者アルゴリズム V は、 $r \rightarrow Rsp$ として、 $c_2 = H_2(r, t' - \alpha r, e' - \alpha(c) - c)$ が成り立つか否かを検証する。

$Ch_B = 1$ の場合、検証者アルゴリズム V は、 $z \rightarrow Rsp$ として、 $c_1 = H_1(\alpha(F(z)) - y) + F_b(z, t') + e', z$ が成り立つか否かを検証する。

[0088] なお、Step. 6 における検証の操作を $0/1 \rightarrow Dec_b(F, y; Cmt_A, Ch_A, Cmt_B, Ch_B, Rsp)$ と表現することにする。この操作において、出力 1 は検証の成功を示し、出力 0 は検証の失敗を示すものとする。

。 以上、SSH10b 公開鍵認証方式における証明者アルゴリズム P 及び検証者アルゴリズム V の処理内容について説明した。なお、上記の方式ではハッシュ関数 H_1 、 H_2 を用いて c_1 、 c_2 を計算しているが、ハッシュ関数 H_1 、 H_2 の代わりにコミットメント関数 COM を用いてもよい。

[0090] [1-5 : 対話プロトコルの繰り返し構成]

さて、上記のSSH10a 公開鍵認証方式の対話プロトコルを適用すれば、偽証が成功する確率を $2/3$ 以下に抑制することができる。従って、この対話プロトコルを 2 回実行すれば、偽証が成功する確率を $(2/3)^2$ 以下に抑制することができる。同様に、この対話プロトコルを N 回実行すると、偽

証が成功する確率は $(2/3)^N$ となり、 N を十分に大きい数（例えば、 $N = 140$ ）にすれば、偽証が成功する確率は無視できる程度に小さくなる。

[0091] 同様に、上記の SSH 10 b 公開鍵認証方式の対話プロトコルを適用すれば、偽証が成功する確率を $(1/2 + 1/2^q)$ 以下に抑制することができる。従って、この対話プロトコルを 2 回実行すれば、偽証が成功する確率を $(1/2 + 1/2^q)^2$ 以下に抑制することができる。同様に、この対話プロトコルを N 回実行すると、偽証が成功する確率は $(1/2 + 1/2^q)^N$ となり、 N を十分に大きい数（例えば、 $N = 80$ ）にすれば、偽証が成功する確率は無視できる程度に小さくなる。

[0092] なお、SSH 10 a 公開鍵認証方式、SSH 10 b 公開鍵認証方式に限らず、上記の非特許文献 1～3 などに開示されている公開鍵認証方式においても同様であるが、対話プロトコルを 1 回実行した場合の偽証成功確率が十分に小さい場合、対話プロトコルを繰り返し実行して偽証成功確率を下げる必要がある。対話プロトコルを繰り返し実行する方法としては、図 5 に示した直列的繰り返し構成と、図 6 に示した並列的繰り返し構成がある。

[0093] 直列的繰り返し構成は、図 5 に示すように、対話プロトコルを 1 回実行し、1 回目が終了した後に 2 回目を、2 回目が終了した後に 3 回目を実行するといった形で、逐次的に対話プロトコルを実行する方法である。一方、並列的繰り返し構成は、図 6 に示すように、 N 回分の 1 パス目のメッセージ $m_{1,1}, \dots, m_{N,1}$ を同時に送信し、次に N 回分の 2 パス目のメッセージ $m_{1,2}, \dots, m_{N,2}$ を同時に送信し、次に N 回分の 3 パス目のメッセージ $m_{1,3}, \dots, m_{N,3}$ を同時に送信するといった形で、各パスにおいて N 回分のメッセージを同時送信する方法である。

[0094] さて、上記の非特許文献 1～3 に開示されている公開鍵認証方式、SSH 10 a 公開鍵認証方式、及び SSH 10 b 公開鍵認証方式は、受動的攻撃に対する安全性レベルを保証している。しかし、これらの公開鍵認証方式は、並列的繰り返し構成を行なった場合、能動的攻撃に対する安全性レベルを確実に保証しているか否かが知られていない。これらの公開鍵認証方式は、1

組の鍵ペア（公開鍵 y 、秘密鍵 s ）を用いて「 y について $y = F(s)$ となる s を知っていること」を証明者が検証者に対して証明する方式であった。

[0095] そのため、検証で受理される対話を行った場合、「対話を行った証明者が s を用いた」という情報を検証者に知られてしまうことが避けられなかった。また、これらの方式で用いる F は衝突困難性が保証されていない。その結果、これらの公開鍵認証方式は、並列的繰り返し構成を行なった場合、能動的攻撃に対する安全性レベルを確実に保証しているか否かが知られていなかったのである。特に、これらの公開鍵認証方式を上記の並列的繰り返し構成により実施する場合には能動的攻撃に対する安全性レベルを保証するための手立てが知られていなかった。

[0096] そこで、本件発明者は、これらの公開鍵認証方式を上記の並列的繰り返し構成により実施する際に能動的攻撃に対する安全性レベルが保証されるようにする仕組みを考案した。この仕組み（本手法 # 1、# 2）については、後段において、SSH 10 a 公開鍵認証方式、SSH 10 b 公開鍵認証方式を例に挙げて詳細に説明する。

[0097] [1-6: SSH 10 a 公開鍵認証方式に対する偽証アルゴリズム]

ここで、図 7 を参照しながら、SSH 10 a 公開鍵認証方式（1 回の対話プロトコル）において $2/3$ の確率で成功する偽証アルゴリズムについて考えてみたい。図 7 は、SSH 10 a 公開鍵認証方式（1 回の対話プロトコル）において $2/3$ の確率で成功する偽証アルゴリズムについて説明するための説明図である。この偽証アルゴリズムは、偽証者が検証者に対して「 $y = F(s)$ を満たす s を知っているふり」をするアルゴリズムである。但し、この偽証アルゴリズムを適用しても、 $1/3$ の確率で偽証が失敗する。

[0098] Step. 1:

まず、偽証者（偽証アルゴリズム）は、ベクトル $s, r, t \in K^n$ 、 $e \in K^m$ を生成する。次いで、偽証アルゴリズムは、 $Ch^* \in \{0, 1, 2\}$ を任意に選択する。この Ch^* は、偽証アルゴリズムが回答できない検証パターンに対応する。なお、ここで用いる s は正当な秘密鍵でない。しかし、証明者（

証明者アルゴリズムP)は、 Ch^* 以外の検証パターンならば正当な秘密鍵を利用せずに正しく回答することができる。

[0099] 次に、偽証アルゴリズムは、 $z \leftarrow s - r$ 、 $t' \leftarrow r + t$ を計算する。また、偽証アルゴリズムは、 $Ch^* = 0$ の場合、 $e' \leftarrow y - F(s) + F(r) - c + e$ を計算する。一方、 $Ch^* = 1, 2$ の場合、偽証アルゴリズムは、 $e' \leftarrow F(r) - c + e$ を計算する。次いで、偽証アルゴリズムは、 $Ch^* = 0, 2$ の場合、 $c_1 \leftarrow H_1(F_b(z, t) + e, z)$ を計算する。一方、 $Ch^* = 1$ の場合、偽証アルゴリズムは、 $c_1 \leftarrow H_1(F(z) + F_b(z, t') + e' - y, z)$ を計算する。

[0100] 次いで、偽証アルゴリズムは、 $c_2 \leftarrow H_2(t, e)$ 、 $c_3 \leftarrow H_3(t', e')$ を計算する。そして、偽証アルゴリズムは、 $St \leftarrow (Ch^*, F, y, s, r, t, e, z, t', e')$ 及び $Cmt \leftarrow (c_1, c_2, c_3)$ を設定する。次いで、偽証アルゴリズムは、 Cmt を検証者(検証者アルゴリズムV)に送る。なお、Step. 1における上記の操作を $(Cmt; St) \leftarrow M_{a, 1}(F, y; Ch^*, s, r, t, e)$ と表現することにする。

[0101] Step. 2 :

Cmt を受け取った検証者アルゴリズムVは、3つの検証パターンのうち、どの検証パターンを利用するかを選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジ $Ch \in_R \{0, 1, 2\}$ を偽証者(偽証アルゴリズム)に送る。

[0102] Step. 3 :

Ch を受け取った偽証アルゴリズムは、検証者アルゴリズムVから受け取ったチャレンジ Ch に応じて検証者アルゴリズムVへと送り返す回答 Rsp を生成する。なお、 $Ch = Ch^*$ の場合、偽証アルゴリズムは、エラーを出力して対話プロトコルを終了する。 $Ch \neq Ch^*$ であって $Ch = 0$ の場合、偽証アルゴリズムは、回答 $Rsp \leftarrow (r, t, e)$ を生成する。また、 $Ch = 1$ の場合、偽証アルゴリズムは、回答 $Rsp = (z, t, e)$ を生成する。そして、 $Ch = 2$ の場合、偽証アルゴリズムは、回答 $Rsp = (z, t', e$

') を生成する。

[0103] なお、Step. 3における上記の操作を $Rsp \leftarrow M_{a,2}(Ch; St)$ と表現することにする。また、Step. 3で生成された Rsp は、検証者（検証者アルゴリズム V ）に送られる。

[0104] Step. 4 :

$Ch \neq Ch^*$ の場合、回答 Rsp が検証者アルゴリズム V に送られるため、検証者アルゴリズム V によって検証処理 $0/1 \leftarrow Dec_a(F, y; Cmt, Ch, Rsp)$ が実施される。

[0105] 以上、SSH10a 公開鍵認証方式に対する偽証アルゴリズムについて説明した。このように、 s を任意に選択したとしても、 $2/3$ の確率（ $Ch \neq Ch^*$ となる確率）で検証を通過する Rsp を検証者に送り返すことができる。従って、1回の対話プロトコルにおいて $2/3$ の確率で偽証が成功してしまう。そこで、先に説明した繰り返し構成が適用される。

[0106] [1-7 : SSH10b 公開鍵認証方式に対する偽証アルゴリズム]

次に、図8を参照しながら、SSH10b 公開鍵認証方式（1回の対話プロトコル）において $1/2$ の確率で成功する偽証アルゴリズムについて考えてみたい。図8は、SSH10b 公開鍵認証方式（1回の対話プロトコル）において $1/2$ の確率で成功する偽証アルゴリズムについて説明するための説明図である。この偽証アルゴリズムは、偽証者が検証者に対して「 $y = F(s)$ を満たす s を知っているふり」をするアルゴリズムである。但し、この偽証アルゴリズムを適用しても、 $1/2$ の確率で偽証が失敗する。

[0107] Step. 1 :

まず、偽証アルゴリズムは、ベクトル $s, r, t \in K^n$ 、 $e \in K^m$ を生成する。次いで、偽証アルゴリズムは、 $Ch^*_B \in \{0, 1\}$ を選択する。この Ch^*_B は、 i_0 の系列について偽証者（偽証アルゴリズム）が回答できない検証パターンに対応する。なお、ここで用いる s_{i_0} は正当な秘密鍵ではないが、偽証アルゴリズムは、 Ch^*_B 以外の検証パターンに対しては正当な秘密鍵を利用せずに正しく回答できる。

[0108] 次いで、偽証アルゴリズムは、 $z \leftarrow s - r$ を計算する。次いで、偽証アルゴリズムは、 $c_1 \leftarrow H_1(F_b(z, t) + e, z)$ 及び $c_2 \leftarrow H_2(r, t, e)$ を計算する。次いで、偽証アルゴリズムは、 $St_A \leftarrow (F, y, Ch^*_B, s, r, t, e, z)$ 及び $Cmt_A \leftarrow (c_1, c_2)$ を設定する。そして、Step. 1で生成された Cmt_A は、検証者（検証者アルゴリズムV）に送られる。なお、上記の $H_1(\dots)$ 、 $H_2(\dots)$ は、ハッシュ関数である。また、Step. 1における上記の操作を $(Cmt_A; St_A) \leftarrow M_{b,1}(F, y; Ch^*_B, s, r, t, e)$ と表現することにする。

[0109] Step. 2 :

Cmt_A を受け取った検証者アルゴリズムVは、 q 通り存在する環Kの元から1つの乱数 α を選択する。そして、検証者アルゴリズムVは、チャレンジ $Ch_A = \alpha$ を偽証者（偽証アルゴリズム）に送る。

[0110] Step. 3 :

Ch_A を受け取った偽証アルゴリズムは、 $t' \leftarrow \alpha r + t$ を計算する。さらに、偽証アルゴリズムは、 $Ch^*_B = 1$ の場合、 $e' \leftarrow \alpha(F(r) - c) + e$ を計算する。一方、 $Ch^*_B = 0$ の場合、偽証アルゴリズムは、 $e' \leftarrow \alpha(y - F(s) + F(r) - c) + e$ を計算する。そして、偽証アルゴリズムは、 $St_B \leftarrow (St_A, Ch_A, t', e')$ 及び $Cmt_B \leftarrow (t', e')$ を設定する。また、偽証アルゴリズムは、 Cmt_B を検証者（検証者アルゴリズムV）に送る。なお、Step. 3における上記の操作を $(Cmt_B; St_B) \leftarrow M_{b,2}(Ch_A; St_A)$ と表現することにする。

[0111] Step. 4 :

Cmt_B を受け取った検証者アルゴリズムVは、2つの検証パターンのうち、どちらの検証パターンを利用するかを選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジ $Ch_B \in_R \{0, 1\}$ を偽証者（偽証アルゴリズム）に送る。

[0112] Step. 5 :

Ch_B を受け取った偽証アルゴリズムは、 $Ch_B = Ch^*_B$ の場合、エラーを

出力して対話プロトコルを終了する。 $Ch_B \neq Ch_B^*$ の場合、偽証アルゴリズムは、検証者アルゴリズムVから受けたチャレンジ Ch_B に応じて検証者（検証者アルゴリズムV）に送り返す回答 Rsp を次のようにして生成する。

[0113] $Ch_B = 0$ の場合、偽証アルゴリズムは、 $Rsp \leftarrow r$ に設定する。 $Ch_B = 1$ の場合、偽証アルゴリズムは、 $Rsp \leftarrow z$ に設定する。そして、偽証アルゴリズムは、回答 Rsp を検証者（検証者アルゴリズムV）に送る。なお、Step. 5における上記の操作を $Rsp \leftarrow M_{b,3}(Ch_B; St_B)$ と表現することにする。

[0114] Step. 6 :

$Ch_B \neq Ch_B^*$ の場合、回答 Rsp が検証者アルゴリズムVに送られるため、検証者アルゴリズムVによって検証処理 $0/1 \leftarrow Dec_b(F, y; Cmt_A, Ch_A, Cmt_B, Ch_B, Rsp)$ が実施される。

[0115] 以上、SSH10b公開鍵認証方式に対する偽証アルゴリズムについて説明した。このように、 s を任意に選択したとしても、 $1/2$ の確率（ $Ch_B \neq Ch_B^*$ となる確率）で検証を通過する Rsp を検証者に送り返すことができる。従って、1回の対話プロトコルにおいて $1/2$ の確率で偽証が成功してしまう。そこで、先に説明した繰り返し構成が適用される。

[0116] （補足）

ここまで、SSH10a公開鍵認証方式、SSH10b公開鍵認証方式、その偽証アルゴリズム、及び対話プロトコルの繰り返し構成について説明してきた。これらの公開鍵認証方式は、1組の鍵ペア（公開鍵 y 、秘密鍵 s ）を用いて「 y について $y = F(s)$ となる s を知っていること」を証明者が検証者に対して証明する方式である。そのため、検証で受理される対話を行った場合、「検証を行った証明者が s を用いた」という情報を検証者に知られてしまうことが避けられなかった。また、これらの方式で用いる F は衝突困難性が保証されていない。その結果、これらの公開鍵認証方式は、並列的繰り返し構成を行なった場合、能動的攻撃に対する安全性レベルを確実に保証しているか否かが知られていなかった。

[0117] そこで、本件発明者は、これらの公開鍵認証方式を上記の並列的繰り返し構成により実施する際に能動的攻撃に対する安全性レベルが保証されるようにする仕組みを考案した。以下、この仕組みについて、具体例を挙げて詳細に説明する。

[0118] < 2 : 第 1 実施形態 (本手法 # 1) >

まず、本開示の第 1 実施形態 (以下、本手法 # 1) について説明する。

[0119] [2 - 1 : 概要]

本手法 # 1 は、SSH 1 0 a 公開鍵認証方式、SSH 1 0 b 公開鍵認証方式に対し、並列的繰り返し構成を行なった場合にも能動的攻撃に対する安全性レベルが保証されるようにする仕組みを適用したものである。繰り返しになるが、能動的攻撃に対する安全性レベルが保証されているか否かが知られていなかった理由は、利用する関数 F の衝突困難性が保証されない上、「検証を行った証明者が s を用いた」という情報を検証者に知られてしまうことが避けられなかったからである。従って、対話プロトコルの中で「検証を行った証明者が s を用いた」という情報を検証者に知られないようにすれば、能動的攻撃に対する安全性レベルを保証することができる。

[0120] そこで、本件発明者は、秘密鍵 s 、公開鍵 y を多重化する方式を考案した。この方式は、 L 個 ($L \geq 2$) の $s_1, \dots, s_L \in K^n$ を秘密鍵とし、 m 本の n 変数多次多項式 $F(x) = (f_1(x), \dots, f_m(x))$ に対して $(y_1, \dots, y_L) = (F(s_1), \dots, F(s_L))$ を満たす $y_1, \dots, y_L \in K^m$ を公開鍵とする。さらに、この方式は、「 $i = 1, \dots, L$ のうち、 $L - 1$ 個の i について $y_i = F(s_i)$ となる s_i を知っていること」を、どの s_i を対話プロトコルの中で使用したかが分からないように証明する方式である。この方式を適用すると、対話プロトコルの中で「どの s_i を利用したか」に関する情報が漏れないため、能動的攻撃に対する安全性レベルが保証されるのである。

[0121] 本手法 # 1 の方式は、検証者が $i = 1, \dots, L$ についてチャレンジ $Ch_1, \dots, Ch_L$ (検証パターン) を証明者に対して送信し、チャレンジ $Ch_1, \dots, Ch_L$ を受け取った証明者が $L - 1$ 個のチャレンジ Ch_i を選択して回答す

るというものである。この方式を用いると、 $s_1, \dots, s_L$ を知っている証明者は L 個全てのチャレンジ $C h_i$ について回答できるが、 $s_1, \dots, s_L$ を知らない偽証者は、ある程度の確率で偽証が失敗する。また、証明者は $L-1$ 個のチャレンジ $C h_i$ に対して回答すればよいから、ある1つのチャレンジ $C h_i$ については s_i を使用せずに認証を成立させることができる。つまり、対話プロトコルの中で、ある s_i を使用したか否かが検証者には分からないのである。

[0122] [2-2: SSH10a 公開鍵認証方式への適用]

まず、図9を参照しながら、本手法#1の方式をSSH10a 公開鍵認証方式に適用した場合の対話プロトコルについて説明する。図9は、本手法#1の方式をSSH10a 公開鍵認証方式に適用した場合の対話プロトコルについて説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0123] (鍵生成アルゴリズム Gen)

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数2次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数2次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 pk に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0124] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図9を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。本手法#1の対話プロトコルは、「証明者が $L-1$ 個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させる

ものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0125] 本手法 # 1 の対話プロトコルは、図 9 に示す $Step. 1 \sim Step. 4$ の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0126] $Step. 1$:

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $(Cmt_i; St_i) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $Cmt_1, \dots, Cmt_L$ を検証者（検証者アルゴリズム V ）に送る。

[0127] $Step. 2$:

$Cmt_1, \dots, Cmt_L$ を受け取った検証者アルゴリズム V は、3 つの検証パターンのうち、どの検証パターンを利用するかを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジ $Ch_1, \dots, Ch_L \in_R \{0, 1, 2\}$ を証明者（証明者アルゴリズム P ）に送る。

[0128] $Step. 3$:

$Ch_1, \dots, Ch_L$ を受け取った証明者アルゴリズム P は、 $i = 1, \dots, L$ から 1 個だけ回答しないチャレンジ Ch_i のインデックス i （以下、 i^* ）をランダムに選択する。次いで、証明者アルゴリズム P は、 $i \in \{1, \dots, L\} \setminus \{i^*\}$ について $Rsp_i \leftarrow P_{a,2}(Ch_i; St_i)$ を計算する。そして、証明者アルゴリズム P は、 $(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を検証者（検証者アルゴリズム V ）に送る。

[0129] $Step. 4$:

$(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を受け取った検証者アルゴリズム V は、 $i \in \{1, \dots, L\} \setminus \{i^*\}$ について $0/1$

$\leftarrow \text{Dec}_a(F, y_i; \text{Cmt}_i, \text{Ch}_i, \text{Resp}_i)$ を実行する。そして、検証者アルゴリズム V は、 $i \in \{1, \dots, L\} \setminus \{i^*\}$ について全て受理（出力 1）されれば検証が成立したものとみなす。

[0130] 以上、本手法 # 1 の方式を SSH 1 0 a 公開鍵認証方式に適用した場合の対話プロトコルについて説明した。この対話プロトコルは、SSH 1 0 a 公開鍵認証方式の安全性により、 s_i を持たない偽証者が $i = 1, \dots, L$ について、それぞれ $2/3$ 以下の確率でしか、検証者から送られたチャレンジ Ch_i に正しく回答できないことが保証されている。また、本手法 # 1 を適用したことにより、 $L - 1$ 個以上の i について検証者から送られたチャレンジ Ch_i に正しく回答する必要があるため、偽証が成功する確率は、 $(2/3)^{L+L} (1/3) (2/3)^{L-1} = (2+L) 2^{L-1} / 3^L$ である。

[0131] また、上記の対話プロトコルにおいては $s_1, \dots, s_L$ の全てを使用しているが、 $L \leq 3$ の場合であれば、 $i = 1, \dots, L$ の 1 つについて s_i を使用しなくても、どの s_i を使用しなかったかを検証者に一切知られずに全く証明者と同様の振る舞いをすることが可能である。そこで、1 つの s_i を使用せずに、上記の対話プロトコルと同じ認証が実現可能な対話プロトコル（変形例 1 ~ 3）について説明する。

[0132] [2 - 3 : SSH 1 0 a 公開鍵認証方式への適用（変形例 1）]

まず、図 1 0 を参照しながら、本手法 # 1 の方式を SSH 1 0 a 公開鍵認証方式に適用した場合の対話プロトコル（変形例 1）について説明する。図 1 0 は、本手法 # 1 の方式を SSH 1 0 a 公開鍵認証方式に適用した場合の対話プロトコル（変形例 1）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。但し、検証者アルゴリズム V において、先に説明した偽証アルゴリズムの構成が利用される。また、 $L = 3$ の場合について説明する。さらに、使用しない s_i のインデックスを i_0 とする。以下、各アルゴリズムの内容について説明する。

[0133] （鍵生成アルゴリズム Gen ）

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。但し、 s_{i_0} は、適当に選択したベクトルを用いる。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。但し、 $y_{i_0} = F(s_{i_0})$ になることは保証されていなくてもよい。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 pk に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0134] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図 10 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0135] 本手法 # 1 の対話プロトコルは、図 10 に示す Step. 1 ~ Step. 4 の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0136] Step. 1 :

まず、証明者アルゴリズム P は、 $i = 1, 2, 3$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $Ch^* \in \{0, 1, 2\}$ を 1 つ選択する。この Ch^* は、 i_0 の系列について証明者が回答できない検証パターンに対応する。次いで、証明者アルゴリズム P は、 $i \neq i_0$ の場合、 $(Cmt_i; St_i) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。一方、 $i = i_0$ の場合、証明者アルゴリズム P は、 $(Cmt_i; St_i) \leftarrow M_{a,1}(F, y_i; Ch^*, s_i, r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 Cmt_1, Cmt_2, Cmt_3 を

検証者（検証者アルゴリズムV）に送る。

[0137] Step. 2 :

Cmt_1, Cmt_2, Cmt_3 を受け取った検証者アルゴリズムVは、3つの検証パターンのうち、どの検証パターンを利用するかを選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジ $Ch_1, Ch_2, Ch_3 \in_R \{0, 1, 2\}$ を証明者（証明者アルゴリズムP）に送る。

[0138] Step. 3 :

Ch_1, Ch_2, Ch_3 を受け取った証明者アルゴリズムPは、 $i = 1, \dots, L$ から1個だけ回答しないチャレンジ Ch_i のインデックス i （以下、 i^* ）を次のようにして選択する。 $Ch_{i_0} = Ch^*$ の場合、検証者アルゴリズムVは、 $i^* \leftarrow i_0$ を設定する。一方、 $Ch_{i_0} \neq Ch^*$ の場合、検証者アルゴリズムVは、 $i^* \in \{1, 2, 3\} \setminus \{i_0\}$ からランダムに i^* を選択する。この i^* の設定方法を用いると、検証者がどのような検証パターンを要求してきても i^* は、それぞれ $1/3$ の確率で1, 2, 3の値をとる。つまり、 i_0 が1, 2, 3のどの値であったかという情報が完全に秘匿されているのである。

[0139] 次いで、証明者アルゴリズムPは、 $i \in \{1, 2, 3\} \setminus \{i^*\}$ について次のように Rsp_i を計算する。 $i \neq i_0$ の場合、証明者アルゴリズムPは、 $Rsp_i \leftarrow P_{a,2}(Ch_i; St_i)$ を計算する。 $i = i_0$ の場合、証明者アルゴリズムPは、 $Rsp_i \leftarrow M_{a,2}(Ch_i; St_i)$ を計算する。そして、証明者アルゴリズムPは、 $(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を検証者（検証者アルゴリズムV）に送る。

[0140] Step. 4 :

$(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を受け取った検証者アルゴリズムVは、 $i \in \{1, 2, 3\} \setminus \{i^*\}$ について $0/1 \leftarrow Dec_a(F, y_i; Cmt_i, Ch_i, Rsp_i)$ を実行する。そして、検証者アルゴリズムVは、 $i \in \{1, 2, 3\} \setminus \{i^*\}$ について全て受理（出力1）されれば検証が成立したものとみなす。

[0141] 以上、本手法#1の方式をSSH10a公開鍵認証方式に適用した場合の

対話プロトコル（変形例）について説明した。ここでは $L = 3$ の場合について説明したが、 $L = 2$ の場合についても同様である。例えば、次のようにして $i = 1, 2$ から 1 個だけ利用しない i (i^*) を設定する。 $Ch_{i_0} = Ch^*$ ならば $i^* = i_0$ に設定する。一方、 $Ch_{i_0} \neq Ch^*$ ならば $1/4$ の確率で $i^* = i_0$ に設定し、 $3/4$ の確率で $i^* \neq i_0$ に設定する。このように設定すると、検証者がどのような検証パターンを要求してきても、 i^* はそれぞれ $1/2$ の確率で $1, 2$ の値をとる。そのため、 i_0 が $1, 2$ のどの値であったかという情報が完全に秘匿される。

[0142] [2-4 : SSH10a 公開鍵認証方式への適用（変形例 2）]

次に、図 11 を参照しながら、本手法 # 1 の方式を SSH10a 公開鍵認証方式に適用した場合の対話プロトコル（変形例 2）について説明する。なお、変形例 2 は、図 9 に示した対話プロトコルの一変形例である。図 11 は、本手法 # 1 の方式を SSH10a 公開鍵認証方式に適用した場合の対話プロトコル（変形例 2）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0143] （鍵生成アルゴリズム Gen ）

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、2 個のベクトル $s_1, s_2 \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1)$ 、 $y_2 = F(s_2)$ を計算する。そして、鍵生成アルゴリズム Gen は、 (F, y_1, y_2) を公開鍵 pk に設定し、 (s_1, s_2) を秘密鍵に設定する。

[0144] （証明者アルゴリズム P 、検証者アルゴリズム V ）

次に、図 11 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム

ムVの構成について説明する。本手法#1の対話プロトコルは、「証明者が1個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズムGenにより生成された公開鍵pkは、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズムGenにより生成された秘密鍵skは、証明者により秘密に管理されているものとする。

[0145] 本手法#1の対話プロトコルは、図11に示すStep. 1～Step. 4の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0146] Step. 1 :

まず、証明者アルゴリズムPは、ベクトル $r_1, t_1 \in K^n$ と、ベクトル $e_1 \in K^m$ を生成する。次いで、証明者アルゴリズムPは、 $(Cmt_1; St_1) \leftarrow P_{a,1}(F, y_1, s_1; r_1, t_1, e_1)$ 、 $(Cmt_2; St_2) \leftarrow P_{a,1}(F, y_2, s_2; r_1, t_1, e_1)$ を計算する。但し、 $Cmt_1 = (c_{11}, c_{12}, c_{13})$ 、 $Cmt_2 = (c_{21}, c_{12}, c_{13})$ となる。次いで、証明者アルゴリズムPは、 $c_{11}, c_{12}, c_{13}, c_{21}$ を検証者（検証者アルゴリズムV）に送る。

[0147] Step. 2 :

$c_{11}, c_{12}, c_{13}, c_{21}$ を受け取った検証者アルゴリズムVは、どの検証パターンを利用するかを選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジの組 $(Ch_1, Ch_2) \in \{0, 1, 2\} \times \{0, 1, 2\}$ を証明者（証明者アルゴリズムP）に送る。

[0148] Step. 3 :

Ch_1, Ch_2 を受け取った証明者アルゴリズムPは、回答するチャレンジ Ch_i のインデックスiをランダムに選択する。次いで、証明者アルゴリズムPは、選択したiについて、 $Rsp_i \leftarrow P_{a,2}(Ch_i; St_i, r_1, t_1, e_1)$ を計算する。そして、証明者アルゴリズムPは、 (Rsp_i, i) を検証

者（検証者アルゴリズムV）に送る。

[0149] Step. 4 :

Rsp_i , i を受け取った検証者アルゴリズムVは、 $0/1 \leftarrow Dec_a(F, y_i; (c_{i1}, c_{i2}, c_{i3}), Ch_i, Rsp_i)$ を実行する。そして、検証者アルゴリズムVは、出力が1の場合（受理された場合）に検証が成立したものとみなす。

[0150] 以上、本手法#1の方式をSSH10a公開鍵認証方式に適用した場合の対話プロトコル（変形例2）について説明した。本変形例は、 Cmt_1 、 St_1 、 Cmt_2 、 St_2 を生成する際に用いる乱数の組 (r_1, t_1, e_1) と (r_2, t_2, e_2) を共通化する点に特徴がある。これらの乱数を共通化することにより、 $Cmt_1 = (c_{11}, c_{12}, c_{13})$ 、 $Cmt_2 = (c_{21}, c_{22}, c_{23})$ を構成する要素のうち、 (c_{12}, c_{13}) と (c_{22}, c_{23}) が共通の値となる。そのため、証明者から検証者に Cmt_1 と Cmt_2 を送る際に、4個の値 $(c_{11}, c_{12}, c_{13}, c_{21})$ だけを送れば済むようになり、通信量を低減させることが可能になる。なお、本変形例の場合、3パス目において一方の系列に関する情報しか公開しないため、上記のように乱数の一部を共通化しても零知識性は損なわれない。

[0151] [2-5: SSH10a公開鍵認証方式への適用（変形例3）]

次に、図12を参照しながら、本手法#1の方式をSSH10a公開鍵認証方式に適用した場合の対話プロトコル（変形例3）について説明する。なお、変形例3は、図11に示した対話プロトコル（変形例2）の一変形例である。図12は、本手法#1の方式をSSH10a公開鍵認証方式に適用した場合の対話プロトコル（変形例3）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズムGen、証明者アルゴリズムP、検証者アルゴリズムVにより構成される。以下、各アルゴリズムの内容について説明する。

[0152] （鍵生成アルゴリズムGen）

まず、鍵生成アルゴリズムGenの構成について説明する。鍵生成アルゴ

リズム Gen は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、2 個のベクトル $s_1, s_2 \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1)$, $y_2 = F(s_2)$ を計算する。そして、鍵生成アルゴリズム Gen は、 (F, y_1, y_2) を公開鍵 pk に設定し、 (s_1, s_2) を秘密鍵に設定する。

[0153] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図 12 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。本手法 # 1 の対話プロトコルは、「証明者が 1 個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0154] 本手法 # 1 の対話プロトコルは、図 12 に示す Step. 1 ~ Step. 4 の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0155] Step. 1 :

まず、証明者アルゴリズム P は、ベクトル $r_1, t_1 \in K^n$ と、ベクトル $e_1 \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $(Cmt_1; St_1) \leftarrow P_{a,1}(F, y_1, s_1; r_1, t_1, e_1)$ 、 $(Cmt_2; St_2) \leftarrow P_{a,1}(F, y_2, s_2; r_1, t_1, e_1)$ を計算する。但し、 $Cmt_1 = (c_{11}, c_{12}, c_{13})$ 、 $Cmt_2 = (c_{21}, c_{12}, c_{13})$ となる。次いで、証明者アルゴリズム P は、 $c_{11}, c_{12}, c_{13}, c_{21}$ を検証者（検証者アルゴリズム V ）に送る。

[0156] Step. 2 :

c_{11} , c_{12} , c_{13} , c_{21} を受け取った検証者アルゴリズムVは、どの検証パターンを利用するかを選択する。このとき、検証者アルゴリズムVは、検証パターン（チャレンジの組 (Ch_1, Ch_2) ）を $(Ch_1, Ch_2) \in \{(0, 0), (1, 1), (1, 2), (2, 1), (2, 2)\}$ の中から選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジの組 (Ch_1, Ch_2) を証明者（証明者アルゴリズムP）に送る。

[0157] Step. 3 :

Ch_1 , Ch_2 を受け取った証明者アルゴリズムPは、回答するチャレンジ Ch_i のインデックス i をランダムに選択する。次いで、証明者アルゴリズムPは、選択した i について、 $Rsp_i \leftarrow P_{a,2}(Ch_i; St_i, r_1, t_1, e_1)$ を計算する。そして、証明者アルゴリズムPは、 (Rsp_i, i) を検証者（検証者アルゴリズムV）に送る。

[0158] Step. 4 :

(Rsp_i, i) を受け取った検証者アルゴリズムVは、 $0/1 \leftarrow Dec_a(F, y_i; (c_{11}, c_{12}, c_{13}), Ch_i, Rsp_i)$ を実行する。そして、検証者アルゴリズムVは、出力が1の場合（受理された場合）に検証が成立したものとみなす。

[0159] 以上、本手法#1の方式をSSH10a公開鍵認証方式に適用した場合の対話プロトコル（変形例3）について説明した。本変形例は、利用可能な検証パターンを $(Ch_1, Ch_2) \in \{(0, 0), (1, 1), (1, 2), (2, 1), (2, 2)\}$ の5パターンに限定する点に特徴がある。

[0160] 上記のように、 Cmt_1 の要素 (c_{12}, c_{13}) と Cmt_2 の要素 (c_{22}, c_{23}) が共通化されている場合、 $Ch_1=0$ に対する回答 (c_{12}, c_{13}) に対する回答）と $Ch_2=0$ に対する回答 (c_{22}, c_{23}) に対する回答）は共通になる。そのため、「 $(Ch_1, Ch_2) = (0, 0)$ に回答できること」は、「 $Ch_1=0$ と $Ch_2=0$ の両方に回答できること」を意味する。つまり、 $(Ch_1, Ch_2) = (0, 0)$ に回答できる証明者は、 $(Ch_1, Ch_2) = (0, 1), (0, 2), (1, 0), (2, 0)$ のいずれにも回答できることにな

る。従って、検証者は、 $(Ch_1, Ch_2) = (0, 0), (1, 1), (1, 2), (2, 1), (2, 2)$ の5パターンについて検証を行えば十分である。

[0161] このような理由から、秘密鍵 s_k を知らない限り、上記の5パターンの全てに回答することはできないことが保証される。つまり、秘密鍵 s_k を知らない者は最大で4パターンにしか回答できないことになるため、偽証確率は最大で $4/5$ になる。図9に示した対話プロトコルの場合、 $L=2$ における偽証確率は $8/9$ であった。従って、変形例3の構成を適用することにより、偽証確率が低減する。

[0162] [2-6: SSH10b 公開鍵認証方式への適用]

次に、図13を参照しながら、本手法#1の方式をSSH10b公開鍵認証方式に適用した場合の対話プロトコルについて説明する。図13は、本手法#1の方式をSSH10b公開鍵認証方式に適用した場合の対話プロトコルについて説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0163] (鍵生成アルゴリズム Gen)

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数2次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数2次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 p_k に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0164] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図13を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。本手法#1の対話プロトコルは、「証明者が

$L-1$ 個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0165] 本手法 # 1 の対話プロトコルは、図 13 に示す $Step. 1 \sim Step. 6$ の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0166] $Step. 1$:

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $(Cmt_{A,i}; St_{A,i}) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $(Cmt_{A,1}, \dots, Cmt_{A,L})$ を検証者（検証者アルゴリズム V ）に送る。

[0167] $Step. 2$:

$(Cmt_{A,1}, \dots, Cmt_{A,L})$ を受け取った検証者アルゴリズム V は、 q 通り存在する環 K の元から L 個の乱数の組 $(\alpha_1, \dots, \alpha_L)$ を選択する。そして、検証者アルゴリズム V は、チャレンジ $(Ch_{A,1}, \dots, Ch_{A,L}) = (\alpha_1, \dots, \alpha_L)$ を証明者（証明者アルゴリズム P ）に送る。

[0168] $Step. 3$:

$(Ch_{A,1}, \dots, Ch_{A,L})$ を受け取った証明者アルゴリズム P は、 $i = 1, \dots, L$ について $(Cmt_{B,i}; St_{B,i}) \leftarrow P_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。そして、証明者アルゴリズム P は、 $(Cmt_{B,1}, \dots, Cmt_{B,L})$ を検証者（検証者アルゴリズム V ）に送る。

[0169] $Step. 4$:

$(Cmt_{B,1}, \dots, Cmt_{B,L})$ を受け取った検証者アルゴリズム V は、2 つの検証パターンのうち、どちらの検証パターンを利用するかを $i = 1, \dots$

、 L のそれぞれについて選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジ $Ch_{B,1}, \dots, Ch_{B,L} \in_R \{0, 1\}$ を証明者（証明者アルゴリズム P ）に送る。

[0170] Step. 5 :

$Ch_{B,1}, \dots, Ch_{B,L}$ を受け取った証明者アルゴリズム P は、 $i = 1, \dots, L$ から1個だけ回答しない i （以下、 i^* ）をランダムに選択する。そして、証明者アルゴリズム P は、 $Rsp_i \leftarrow P_{b,3}(Ch_{B,i}; St_{B,i})$ を計算する。そして、証明者アルゴリズム P は、 $(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を検証者（検証者アルゴリズム V ）に送る。

[0171] Step. 6 :

$(Rsp_1, \dots, Rsp_{i^*-1}, Rsp_{i^*+1}, \dots, Rsp_L, i^*)$ を受け取った検証者アルゴリズム V は、 $i \in \{1, \dots, L\} \setminus \{i^*\}$ について $0/1 \leftarrow Dec_b(F, y_i; Cmt_{A,i}, Ch_{A,i}, Cmt_{B,i}, Ch_{B,i}, Rsp_i)$ を実施する。そして、検証者アルゴリズム V は、 $i \in \{1, \dots, L\} \setminus \{i^*\}$ について全て受理（出力1）されれば検証が成立したものとみなす。

[0172] 以上、本手法#1の方式をSSH10b公開鍵認証方式に適用した場合の対話プロトコルについて説明した。この対話プロトコルは、SSH10b公開鍵認証方式の安全性により、 s_i を持たない偽証者が $i = 1, \dots, L$ について、それぞれ $1/2 + 1/2q$ 以下の確率でしか、検証者から送られたチャレンジ Ch_i に正しく回答できないことが保証されている。また、本手法#1を適用したことにより、 $L-1$ 個以上の i について検証者から送られたチャレンジ Ch_i に正しく回答する必要があるため、偽証が成功する確率は、 $(1/2 + 1/2q)^L + L(1/2 - 1/2q)(1/2 + 1/2q)^{L-1}$ である。

[0173] また、上記の対話プロトコルにおいては $s_1, \dots, s_L$ の全てを使用しているが、 $L=2$ の場合であれば、 $i = 1, \dots, L$ の1つについて s_i を使用しなくても、どの s_i を使用しなかったかを検証者に一切知られずに全く証明者と同様の振る舞いをすることが可能である。そこで、1つの s_i を使用せずに、

上記の対話プロトコルと同じ認証が実現可能な対話プロトコル（変形例）について説明する。

[0174] [2-7: SSH 10b 公開鍵認証方式への適用（変形例）]

以下、図14を参照しながら、本手法#1の方式をSSH 10b 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明する。図14は、本手法#1の方式をSSH 10b 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズムGen、証明者アルゴリズムP、検証者アルゴリズムVにより構成される。但し、検証者アルゴリズムVにおいて、先に説明した偽証アルゴリズムの構成が利用される。また、 $L=2$ の場合について説明する。さらに、使用しない s_i のインデックスを i_0 とする。以下、各アルゴリズムの内容について説明する。

[0175] （鍵生成アルゴリズムGen）

まず、鍵生成アルゴリズムGenの構成について説明する。鍵生成アルゴリズムGenは、環K上で定義されるm本のn変数2次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、L個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。但し、 s_{i_0} は、適当に選択したベクトルを用いる。なお、n変数のベクトル $(x_1, \dots, x_n)$ をxと表記し、m本のn変数2次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズムGenは、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。但し、 $y_{i_0} = F(s_{i_0})$ になることは保証されていなくてもよい。そして、鍵生成アルゴリズムGenは、 $(F, y_1, \dots, y_L)$ を公開鍵pkに設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0176] （証明者アルゴリズムP、検証者アルゴリズムV）

次に、図14を参照しながら、証明者アルゴリズムP、検証者アルゴリズムVの構成について説明する。なお、鍵生成アルゴリズムGenにより生成された公開鍵pkは、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズムGenにより生成された秘密鍵skは、証明者によ

り秘密に管理されているものとする。

[0177] 本手法#1の対話プロトコルは、図14に示すStep. 1～Step. 6の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0178] Step. 1 :

まず、証明者アルゴリズムPは、 $i = 1, 2$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズムPは、 $Ch_B^* \in \{0, 1\}$ を1つ選択する。この Ch_B^* は、 i_0 の系列について証明者が回答できない検証パターンに対応する。次いで、証明者アルゴリズムPは、 $i \neq i_0$ の場合について、 $(Cmt_{A,i}; St_{A,i}) \leftarrow P_{b,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。また、 $i = i_0$ の場合について、証明者アルゴリズムPは、 $(Cmt_{A,i}; St_{A,i}) \leftarrow M_{b,1}(F, y_i; Ch_B^*, s_i, r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズムPは、 $(Cmt_{A,1}, Cmt_{A,2})$ を検証者（検証者アルゴリズムV）に送る。

[0179] Step. 2 :

$(Cmt_{A,1}, Cmt_{A,2})$ を受け取った検証者アルゴリズムVは、 q 通り存在する環Kの元から2つの乱数の組 (α_1, α_2) を選択する。そして、検証者アルゴリズムVは、チャレンジ $(Ch_{A,1}, Ch_{A,2}) = (\alpha_1, \alpha_2)$ を証明者（証明者アルゴリズムP）に送る。

[0180] Step. 3 :

$(Ch_{A,1}, Ch_{A,2})$ を受け取った証明者アルゴリズムPは、 $i = 1, 2$ について次のように $Cmt_{B,i}$ を計算する。 $i \neq i_0$ の場合、証明者アルゴリズムPは、 $(Cmt_{B,i}; St_{B,i}) \leftarrow P_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。一方、 $i = i_0$ の場合、証明者アルゴリズムPは、 $(Cmt_{B,i}; St_{B,i}) \leftarrow M_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。そして、証明者アルゴリズムPは、 $(Cmt_{B,1}, Cmt_{B,2})$ を検証者（検証者アルゴリズムV）に送る。

[0181] Step. 4 :

($Cmt_{B,1}, Cmt_{B,2}$)を受け取った検証者アルゴリズムVは、2つの検証パターンのうち、どちらの検証パターンを利用するかを選択する。そして、検証者アルゴリズムVは、選択した検証パターンを表すチャレンジ($Ch_{B,1}, Ch_{B,2}$) $\in_R \{0, 1\}$ を証明者(証明者アルゴリズムP)に送る。

[0182] Step. 5 :

($Ch_{B,1}, Ch_{B,2}$)を受け取った証明者アルゴリズムPは、 $i = 1, 2$ から1個だけ回答しない i (以下、 i^*)を次のようにして選択する。 $Ch_{i_0} = Ch^*$ の場合、証明者アルゴリズムPは、 $Ch_{i_0} \neq Ch^*$ に設定する。一方、 $i \neq i_0$ の場合、証明者アルゴリズムPは、 $i^* \neq i_0$ となるようにランダムに設定する。この i^* の設定方法を用いると、検証者がどのような検証パターンを要求してきても i^* は、それぞれ $1/2$ の確率で1, 2の値をとる。つまり、 i_0 が1, 2のどの値であったかという情報が完全に秘匿されているのである。

[0183] 次いで、証明者アルゴリズムPは、 $i \in \{1, 2\} \setminus \{i^*\}$ について次のように Rsp_i を計算する。 $i \neq i_0$ の場合、証明者アルゴリズムPは、 $Rsp_i \leftarrow P_{b,3}(Ch_i; St_i)$ を計算する。 $i = i_0$ の場合、証明者アルゴリズムPは、 $Rsp_i \leftarrow M_{b,3}(Ch_i; St_i)$ を計算する。そして、証明者アルゴリズムPは、(Rsp_1, Rsp_2, i^*)を検証者(検証者アルゴリズムV)に送る。

[0184] Step. 6 :

(Rsp_1, Rsp_2, i^*)を受け取った検証者アルゴリズムVは、 $i \in \{1, 2\} \setminus \{i^*\}$ について $0/1 \leftarrow Dec_b(F, y_i; Cmt_{A,i}, Ch_{A,i}, Cmt_{B,i}, Ch_{B,i}, Rsp_i)$ を実行する。そして、検証者アルゴリズムVは、 $i \in \{1, 2\} \setminus \{i^*\}$ について全て受理(出力1)されれば検証が成立したものとみなす。

[0185] 以上、本手法#1の方式をSSH10b公開鍵認証方式に適用した場合の対話プロトコル(変形例)について説明した。ここでは $L = 2$ の場合につい

て説明したが、 $L \geq 3$ の場合についても同じ仕組みを実現することは可能である。

[0186] 以上、本開示の第1実施形態について説明した。上記説明においては、SSH10a公開鍵認証方式、SSH10b公開鍵認証方式を例に挙げて述べたが、本手法#1の適用範囲はこれに限定されない。SSH10a公開鍵認証方式やSSH10b公開鍵認証方式の変形例、及びその他の公開鍵認証方式に対しても適用が可能である。例えば、上記の例では公開鍵を (F, y) に設定しているが、 F が秘密鍵によらないパラメータであるため、この F を証明者毎に設定するのではなく、システム全体に共通するパラメータとしてもよい。この場合、個別に公開すべき公開鍵は y だけとなり、公開鍵のサイズが小さくなる。また、上記の対話プロトコルにおいては、 $i = 1, \dots, L$ について個別に乱数 $\alpha_1, \dots, \alpha_L$ を選択しているが、これらの乱数を1つに共通化してもよい。この場合、チャレンジ $Ch_{A, i}$ を送る際の通信コストを削減することが可能になる。

[0187] <3：第2実施形態（本手法#2）>

次に、本開示の第2実施形態（本手法#2）について説明する。

[0188] [3-1：概要]

本手法#2は、SSH10a公開鍵認証方式、SSH10b公開鍵認証方式に対し、能動的攻撃に対する安全性レベルが保証されるようにする仕組みを適用したものである。繰り返しになるが、能動的攻撃に対する安全性レベルが保証されているか否かが知られていなかった理由は、利用する関数 F の衝突困難性が保証されない上、「検証を行った証明者が s を用いた」という情報を検証者に知られてしまうことが避けられなかったからである。従って、対話プロトコルの中で「検証を行った証明者が s を用いた」という情報を検証者に知られないようにすれば、能動的攻撃に対する安全性レベルを保証することができる。

[0189] そこで、本件発明者は、秘密鍵 s 、公開鍵 y を多重化する方式を考案した。この方式は、 L 個（ $L \geq 2$ ）の $s_1, \dots, s_L \in K^n$ を秘密鍵とし、 m 本の n

変数多次多項式 $F(x) = (f_1(x), \dots, f_m(x))$ に対して $(y_1, \dots, y_L) = (F(s_1), \dots, F(s_L))$ を満たす $y_1, \dots, y_L \in K^m$ を公開鍵とする。さらに、この方式は、どの s_i を対話プロトコルの中で使用したかが分からないように証明する方式である。この方式を適用すると、対話プロトコルの中で「どの s_i を利用したか」に関する情報が漏れないため、能動的攻撃に対する安全性レベルが保証されるのである。

[0190] 本手法#2の方式は、検証者が $i = 1, \dots, L$ についてチャレンジの組 $(Ch_1^{(0)}, \dots, Ch_L^{(0)})$ 及び $(Ch_1^{(1)}, \dots, Ch_L^{(1)})$ を証明者に送り、証明者が片方のチャレンジの組を選択して回答するというものである。この手法は、一般に Q ($Q \geq 2$) 組のチャレンジの組を証明者に送り、証明者が1つのチャレンジの組を選択して回答するものであるが、ここでは $Q = 2$ の場合について述べることにする。 $s_1, \dots, s_L$ を知っている証明者は両方のチャレンジの組に対して回答できるが、 $s_1, \dots, s_L$ を知らない偽証者は、ある程度の確率でいずれのチャレンジに対しても回答することができずに偽証が失敗する。なお、証明者は、1つの i については s_i を利用しなくても片方のチャレンジの組に対して回答することができるため、検証が受理される対話プロトコルを実行したとしても、この対話プロトコルの中で、どの s_i を利用したかは検証者に知られることがないのである。

[0191] [3-2: SSH10a 公開鍵認証方式への適用]

まず、図15を参照しながら、本手法#2の方式をSSH10a 公開鍵認証方式に適用した場合の対話プロトコルについて説明する。図15は、本手法#2の方式をSSH10a 公開鍵認証方式に適用した場合の対話プロトコルについて説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズムGen、証明者アルゴリズムP、検証者アルゴリズムVにより構成される。以下、各アルゴリズムの内容について説明する。

[0192] (鍵生成アルゴリズムGen)

まず、鍵生成アルゴリズムGenの構成について説明する。鍵生成アルゴリズムGenは、環K上で定義されるm本のn変数2次多項式 $f_1(x_1, \dots$

, x_n), ..., $f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 pk に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0193] 次に、図 15 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。本手法 # 1 の対話プロトコルは、「証明者が $L-1$ 個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0194] 本手法 # 2 の対話プロトコルは、図 15 に示す Step. 1 ~ Step. 4 の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0195] Step. 1 :

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $(Cmt_i; St_i) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $Cmt_1, \dots, Cmt_L$ を検証者（検証者アルゴリズム V ）に送る。

[0196] Step. 2 :

$Cmt_1, \dots, Cmt_L$ を受け取った検証者アルゴリズム V は、3 つの検証パターンのうち、どの検証パターンを利用するかを選択する。このとき、検証者アルゴリズム V は、2 組の検証パターンの組み合わせを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジの組

$(Ch_1^{(0)}, \dots, Ch_L^{(0)}), (Ch_1^{(1)}, \dots, Ch_L^{(1)}) \in_R \{0, 1, 2\}$ を証明者（証明者アルゴリズム P）に送る。但し、 $Ch_i^{(0)} \neq Ch_i^{(1)}$ である。

[0197] Step. 3 :

$(Ch_1^{(0)}, \dots, Ch_L^{(0)}), (Ch_1^{(1)}, \dots, Ch_L^{(1)})$ を受け取った証明者アルゴリズム P は、いずれのチャレンジの組に回答するかをランダムに選択する。なお、その選択結果を $d \in_R \{0, 1\}$ と表現する。そして、証明者アルゴリズム P は、 $i \in \{1, \dots, L\}$ について $Rsp_i \leftarrow P_{a, 2}(Ch^{(d)}_i; St_i)$ を計算する。その後、証明者アルゴリズム P は、 $(Rsp_1, \dots, Rsp_L, d)$ を検証者（検証者アルゴリズム V）に送る。

[0198] Step. 4 :

$(Rsp_1, \dots, Rsp_L, d)$ を受け取った検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について $0/1 \leftarrow Dec_a(F, y_i; Cmt_i, Ch^{(d)}_i, Rsp_i)$ を実行する。そして、検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について全て受理（出力 1）されれば検証が成立したものとみなす。

[0199] 以上、本手法 # 2 の方式を SSH10a 公開鍵認証方式に適用した場合の対話プロトコルについて説明した。この対話プロトコルは、SSH10a 公開鍵認証方式の安全性により、 s_i を持たない偽証者が $i = 1, \dots, L$ について、それぞれ $2/3$ 以下の確率でしか、検証者から送られたチャレンジ Ch_i に正しく回答できないことが保証されている。また、本手法 # 2 を適用したことにより、2 つのチャレンジの組の一方については L 個の i について検証者から送られたチャレンジ Ch_i に正しく回答する必要があるため、偽証が成功する確率は、 $(2/3)^L + (2/3)^L - (1/3)^L = (2^{L+1} - 1) / 3^L$ である。

[0200] また、上記の対話プロトコルにおいては $s_1, \dots, s_L$ の全てを使用しているが、 $L \leq 3$ の場合であれば、 $i = 1, \dots, L$ の 1 つについて s_i を使用しなくても、どの s_i を使用しなかったかを検証者に一切知られずに全く証明者と同様の振る舞いをすることが可能である。そこで、1 つの s_i を使用せずに、

上記の対話プロトコルと同じ認証が実現可能な対話プロトコル（変形例）について説明する。

[0201] [3-3: SSH 10a 公開鍵認証方式への適用（変形例）]

以下、図16を参照しながら、本手法#2の方式をSSH 10a 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明する。図16は、本手法#2の方式をSSH 10a 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズムGen、証明者アルゴリズムP、検証者アルゴリズムVにより構成される。但し、検証者アルゴリズムVにおいて、先に説明した偽証アルゴリズムの構成が利用される。また、使用しない s_i のインデックスを i_0 とする。以下、各アルゴリズムの内容について説明する。

[0202] （鍵生成アルゴリズムGen）

まず、鍵生成アルゴリズムGenの構成について説明する。鍵生成アルゴリズムGenは、環K上で定義されるm本のn変数2次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、L個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。但し、 s_{i_0} は、適当に選択したベクトルを用いる。なお、n変数のベクトル $(x_1, \dots, x_n)$ をxと表記し、m本のn変数2次多項式 $(f_1(x), \dots, f_m(x))$ をF(x)と表記する。次に、鍵生成アルゴリズムGenは、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。但し、 $y_{i_0} = F(s_{i_0})$ になることは保証されていなくてもよい。そして、鍵生成アルゴリズムGenは、 $(F, y_1, \dots, y_L)$ を公開鍵pkに設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0203] （証明者アルゴリズムP、検証者アルゴリズムV）

次に、図16を参照しながら、証明者アルゴリズムP、検証者アルゴリズムVの構成について説明する。なお、鍵生成アルゴリズムGenにより生成された公開鍵pkは、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズムGenにより生成された秘密鍵skは、証明者により秘密に管理されているものとする。

[0204] 本手法#2の対話プロトコルは、図16に示すStep. 1~Step.

4の処理ステップにより構成される。以下、各ステップの処理について説明

する。

[0205] Step. 1 :

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $Ch^* \in \{0, 1, 2\}$ を1つ選択する。この Ch^* は、 i_0 の系列について証明者が回答できない検証パターンに対応する。次いで、証明者アルゴリズム P は、 $i \neq i_0$ の場合、 $(Cmt_i; St_i) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。一方、 $i = i_0$ の場合、証明者アルゴリズム P は、 $(Cmt_i; St_i) \leftarrow M_{a,1}(F, y_i; Ch^*, s_i, r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $Cmt_1, \dots, Cmt_L$ を検証者(検証者アルゴリズム V)に送る。

[0206] Step. 2 :

$Cmt_1, \dots, Cmt_L$ を受け取った検証者アルゴリズム V は、3つの検証パターンのうち、どの検証パターンを利用するかを選択する。このとき、検証者アルゴリズム V は、2組の検証パターンの組み合わせを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジの組 $(Ch_1^{(0)}, \dots, Ch_L^{(0)})$ 、 $(Ch_1^{(1)}, \dots, Ch_L^{(1)}) \in_R \{0, 1\}$ 、2}を証明者(証明者アルゴリズム P)に送る。但し、 $Ch_i^{(0)} \neq Ch_i^{(1)}$ である。

[0207] Step. 3 :

$(Ch_1^{(0)}, \dots, Ch_L^{(0)})$ 、 $(Ch_1^{(1)}, \dots, Ch_L^{(1)})$ を受け取った証明者アルゴリズム P は、いずれのチャレンジの組に回答するかを次のようにして選択する。なお、その選択結果を $d \in_R \{0, 1\}$ と表現する。 $Ch_{i_0}^{(0)} = Ch^*$ の場合(条件1)、証明者アルゴリズム P は、 $d \leftarrow 1$ に設定する。一方、 $Ch_{i_0}^{(1)} = Ch^*$ の場合(条件2)、証明者アルゴリズム P は、 $d \leftarrow 0$ に設定する。条件1、2以外の場合、証明者アルゴリズム P は、ラン

ダムに $d \in \{0, 1\}$ を設定する。このような設定方法によると、検証者がどのような検証パターンを要求してきても、 d は $1/2$ の確率で $0, 1$ の値をとるため、 i_0 が $1, \dots, L$ のいずれであったかという情報は完全に秘匿される。

[0208] d を設定した後、証明者アルゴリズム P は、 $i \in \{1, \dots, L\}$ について次のように Rsp_i を計算する。 $i \neq i_0$ の場合、証明者アルゴリズム P は、 $Rsp_i \leftarrow P_{a,2}(Ch^{(d)}_i; St_i)$ を計算する。一方、 $i = i_0$ の場合、証明者アルゴリズム P は、 $Rsp_i \leftarrow M_{a,2}(Ch^{(d)}_i; St_i)$ を計算する。その後、証明者アルゴリズム P は、 $(Rsp_1, \dots, Rsp_L, d)$ を検証者（検証者アルゴリズム V ）に送る。

[0209] Step. 4 :

$(Rsp_1, \dots, Rsp_L, d)$ を受け取った検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について $0/1 \leftarrow Dec_a(F, y_i; Cmt_i, Ch^{(d)}_i, Rsp_i)$ を実行する。そして、検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について全て受理（出力 1）されれば検証が成立したものとみなす。

[0210] 以上、本手法 # 2 の方式を SSH 1 0 a 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明した。なお、上記の説明においては、2 つのチャレンジの組を検証者から証明者に送っているが、例えば、2 つ目のチャレンジが、 $Ch_i^{(1)} = Ch_i^{(0)} + 1 \pmod{3}$ であると予め決めてしまってもよい。この場合、片方のチャレンジの組を送らなくて済むため、通信量を削減することが可能になる。

[0211] [3-4 : SSH 1 0 b 公開鍵認証方式への適用]

次に、図 1 7 を参照しながら、本手法 # 2 の方式を SSH 1 0 b 公開鍵認証方式に適用した場合の対話プロトコルについて説明する。図 1 7 は、本手法 # 2 の方式を SSH 1 0 b 公開鍵認証方式に適用した場合の対話プロトコルについて説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。以下、各アルゴリズムの内容について説明する。

[0212] (鍵生成アルゴリズム Gen)

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴリズム Gen は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 pk に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0213] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図 135 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。本手法 # 2 の対話プロトコルは、「証明者が $L-1$ 個の $y_i = F(s_i)$ を満たす s_i を知っていること」を、「どの s_i を使用したか否か」という情報を検証者に一切漏らさずに、検証者に証明させるものである。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0214] 本手法 # 2 の対話プロトコルは、図 17 に示す $Step. 1 \sim Step. 6$ の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0215] $Step. 1$:

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $(Cmt_{A,i}; St_{A,i}) \leftarrow P_{a,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $(Cmt_{A,1}, \dots, Cmt_{A,L})$ を検証者 (検証者アルゴリズム V) に送る。

[0216] $Step. 2$:

($Cmt_{A,1}, \dots, Cmt_{A,L}$) を受け取った検証者アルゴリズム V は、 q 通り存在する環 K の元から L 個の乱数の組 ($\alpha_1, \dots, \alpha_L$) を選択する。そして、検証者アルゴリズム V は、チャレンジ ($Ch_{A,1}, \dots, Ch_{A,L}$) = ($\alpha_1, \dots, \alpha_L$) を証明者 (証明者アルゴリズム P) に送る。

[0217] Step. 3 :

($Ch_{A,1}, \dots, Ch_{A,L}$) を受け取った証明者アルゴリズム P は、 $i = 1, \dots, L$ について ($Cmt_{B,i}; St_{B,i}$) $\leftarrow P_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。そして、証明者アルゴリズム P は、($Cmt_{B,1}, \dots, Cmt_{B,L}$) を検証者 (検証者アルゴリズム V) に送る。

[0218] Step. 4 :

($Cmt_{B,1}, \dots, Cmt_{B,L}$) を受け取った検証者アルゴリズム V は、3 つの検証パターンのうち、どの検証パターンを利用するかを選択する。このとき、検証者アルゴリズム V は、2 組の検証パターンの組み合わせを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジの組 ($Ch_{B,1}^{(0)}, \dots, Ch_{B,L}^{(0)}$), ($Ch_{B,1}^{(1)}, \dots, Ch_{B,L}^{(1)}$) $\in_R \{0, 1\}^L$ を証明者 (証明者アルゴリズム P) に送る。但し、 $Ch_{B,i}^{(0)} \neq Ch_{B,i}^{(1)}$ である。

[0219] Step. 5 :

($Ch_{B,1}^{(0)}, \dots, Ch_{B,L}^{(0)}$), ($Ch_{B,1}^{(1)}, \dots, Ch_{B,L}^{(1)}$) を受け取った証明者アルゴリズム P は、いずれのチャレンジの組に回答するかをランダムに選択する。なお、その選択結果を $d \in_R \{0, 1\}$ と表現する。そして、証明者アルゴリズム P は、 $i \in \{1, \dots, L\}$ について $Rsp_i \leftarrow P_{b,2}(Ch_{B,i}^{(d)}; St_{B,i})$ を計算する。その後、証明者アルゴリズム P は、($Rsp_1, \dots, Rsp_L, d$) を検証者 (検証者アルゴリズム V) に送る。

[0220] Step. 6 :

($Rsp_1, \dots, Rsp_L, d$) を受け取った検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について $0/1 \leftarrow Dec_b(F, y_i; Cmt_{A,i}, Ch_{A,i})$

, $Cmt_{B,i}$, $Ch_{B,i}^{(d)}$, Rsp_i) を実行する。そして、検証者アルゴリズム V は、 $i \in \{1, \dots, L\}$ について全て受理（出力 1）されれば検証が成立したものとみなす。

[0221] 以上、本手法 # 2 の方式を SSH 1 0 b 公開鍵認証方式に適用した場合の対話プロトコルについて説明した。この対話プロトコルは、SSH 1 0 b 公開鍵認証方式の安全性により、 s_i を持たない偽証者が $i = 1, \dots, L$ について、それぞれ $1/2 + 1/2q$ 以下の確率でしか、検証者から送られたチャレンジに正しく回答できないことが保証されている。また、本手法 # 2 を適用したことにより、2 つのチャレンジの組の一方に対し、 L 個の i について検証者から送られたチャレンジに正しく回答する必要があるため、偽証が成功する確率は、 $(1/2 + 1/2q)^L + (1/2 + 1/2q)^L = 2(1/2 + 1/2q)^L$ である。

[0222] また、上記の対話プロトコルにおいては $s_1, \dots, s_L$ の全てを使用しているが、 $i = 1, \dots, L$ の 1 つについて s_i を使用しなくても、どの s_i を使用しなかったかを検証者に一切知られずに全く証明者と同様の振る舞いをする事が可能である。そこで、1 つの s_i を使用せずに、上記の対話プロトコルと同じ認証が実現可能な対話プロトコル（変形例）について説明する。

[0223] [3 - 5 : SSH 1 0 b 公開鍵認証方式への適用（変形例）]

以下、図 1 8 を参照しながら、本手法 # 2 の方式を SSH 1 0 b 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明する。図 1 8 は、本手法 # 2 の方式を SSH 1 0 b 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明するための説明図である。この対話プロトコルは、鍵生成アルゴリズム Gen 、証明者アルゴリズム P 、検証者アルゴリズム V により構成される。但し、検証者アルゴリズム V において、先に説明した偽証アルゴリズムの構成が利用される。また、使用しない s_i のインデックスを i_0 とする。以下、各アルゴリズムの内容について説明する。

[0224] （鍵生成アルゴリズム Gen ）

まず、鍵生成アルゴリズム Gen の構成について説明する。鍵生成アルゴ

リズム Gen は、環 K 上で定義される m 本の n 変数 2 次多項式 $f_1(x_1, \dots, x_n), \dots, f_m(x_1, \dots, x_n)$ と、 L 個のベクトル $s_1, \dots, s_L \in K^n$ を生成する。但し、 s_{i_0} は、適当に選択したベクトルを用いる。なお、 n 変数のベクトル $(x_1, \dots, x_n)$ を x と表記し、 m 本の n 変数 2 次多項式 $(f_1(x), \dots, f_m(x))$ を $F(x)$ と表記する。次に、鍵生成アルゴリズム Gen は、 $y_1 = F(s_1), \dots, y_L = F(s_L)$ を計算する。但し、 $y_{i_0} = F(s_{i_0})$ になることは保証されていなくてもよい。そして、鍵生成アルゴリズム Gen は、 $(F, y_1, \dots, y_L)$ を公開鍵 pk に設定し、 $(s_1, \dots, s_L)$ を秘密鍵に設定する。

[0225] (証明者アルゴリズム P 、検証者アルゴリズム V)

次に、図 18 を参照しながら、証明者アルゴリズム P 、検証者アルゴリズム V の構成について説明する。なお、鍵生成アルゴリズム Gen により生成された公開鍵 pk は、証明者と検証者の間で共有されているものとする。また、鍵生成アルゴリズム Gen により生成された秘密鍵 sk は、証明者により秘密に管理されているものとする。

[0226] 本手法 #2 の対話プロトコルは、図 18 に示す Step. 1 ~ Step. 6 の処理ステップにより構成される。以下、各ステップの処理について説明する。

[0227] Step. 1 :

まず、証明者アルゴリズム P は、 $i = 1, \dots, L$ について、ベクトル $r_i, t_i \in K^n$ と、ベクトル $e_i \in K^m$ を生成する。次いで、証明者アルゴリズム P は、 $Ch_B^* \in \{0, 1\}$ を 1 つ選択する。この Ch^* は、 i_0 の系列について証明者が回答できない検証パターンに対応する。次いで、証明者アルゴリズム P は、 $i \neq i_0$ の場合について、 $(Cmt_{A,i}; St_{A,i}) \leftarrow P_{b,1}(F, y_i, s_i; r_i, t_i, e_i)$ を計算する。また、 $i = i_0$ の場合について、証明者アルゴリズム P は、 $(Cmt_{A,i}; St_{A,i}) \leftarrow M_{b,1}(F, y_i; Ch_B^*, s_i, r_i, t_i, e_i)$ を計算する。そして、証明者アルゴリズム P は、 $(Cmt_{A,1}, \dots, Cmt_{A,L})$ を検証者（検証者アルゴリズム V ）に送る。

[0228] Step. 2 :

$(Cmt_{A,1}, \dots, Cmt_{A,L})$ を受け取った検証者アルゴリズム V は、 q 通り存在する環 K の元から L 個の乱数の組 $(\alpha_1, \dots, \alpha_L)$ を選択する。そして、検証者アルゴリズム V は、チャレンジ $(Ch_{A,1}, \dots, Ch_{A,L}) = (\alpha_1, \dots, \alpha_L)$ を証明者（証明者アルゴリズム P ）に送る。

[0229] Step. 3 :

$(Ch_{A,1}, \dots, Ch_{A,L})$ を受け取った証明者アルゴリズム P は、 $i = 1, \dots, L$ について次のように $Cmt_{B,i}$ を計算する。まず、 $i \neq i_0$ の場合について、証明者アルゴリズム P は、 $(Cmt_{B,i}; St_{B,i}) \leftarrow P_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。一方、 $i = i_0$ の場合について、証明者アルゴリズム P は、 $(Cmt_{B,i}; St_{B,i}) \leftarrow M_{b,2}(Ch_{A,i}; St_{A,i})$ を計算する。そして、証明者アルゴリズム P は、 $(Cmt_{B,1}, \dots, Cmt_{B,L})$ を検証者（検証者アルゴリズム V ）に送る。

[0230] Step. 4 :

$(Cmt_{B,1}, \dots, Cmt_{B,L})$ を受け取った検証者アルゴリズム V は、3 つの検証パターンのうち、どの検証パターンを利用するかを選択する。このとき、検証者アルゴリズム V は、2 組の検証パターンの組み合わせを選択する。そして、検証者アルゴリズム V は、選択した検証パターンを表すチャレンジの組 $(Ch_{B,1}^{(0)}, \dots, Ch_{B,L}^{(0)}), (Ch_{B,1}^{(1)}, \dots, Ch_{B,L}^{(1)}) \in_R \{0, 1\}^L$ を証明者（証明者アルゴリズム P ）に送る。但し、 $Ch_{B,i}^{(0)} \neq Ch_{B,i}^{(1)}$ である。

[0231] Step. 5 :

$(Ch_{B,1}^{(0)}, \dots, Ch_{B,L}^{(0)}), (Ch_{B,1}^{(1)}, \dots, Ch_{B,L}^{(1)})$ を受け取った証明者アルゴリズム P は、いずれのチャレンジの組に回答するかを次のようにして選択する。なお、その選択結果を $d \in_R \{0, 1\}$ と表現する。 $Ch_{B,i_0}^{(0)} = Ch_{B,*}$ の場合、証明者アルゴリズム P は、 $d \leftarrow 1$ に設定する。一方、 $Ch_{B,i_0}^{(1)} = Ch_{B,*}$ の場合、証明者アルゴリズム P は、 $d \leftarrow 0$ に設定する。この設定方法を用いると、検証者がどのような検証パター

ンを要求してきても d は、それぞれ $1/2$ の確率で $0, 1$ の値をとる。つまり、 d が $0, 1$ のどの値であったかという情報が完全に秘匿されているのである。

[0232] 次いで、証明者アルゴリズム P は、 $i = 1, \dots, L$ について次のように Rsp_i を計算する。 $i \neq i_0$ の場合について、証明者アルゴリズム P は、 $Rsp_i \leftarrow P_{b,3}(Ch_{B,i}^{(d)}; St_{B,i})$ を計算する。また、 $i = i_0$ の場合について、証明者アルゴリズム P は、 $Rsp_i \leftarrow M_{b,3}(Ch_{B,i}^{(d)}; St_{B,i})$ を計算する。その後、証明者アルゴリズム P は、 $(Rsp_1, \dots, Rsp_L, d)$ を検証者（検証者アルゴリズム V ）に送る。

[0233] Step. 6 :

$(Rsp_1, \dots, Rsp_L, d)$ を受け取った検証者アルゴリズム V は、 $i = 1, \dots, L$ について $0/1 \leftarrow Dec_b(F, y_i; Cmt_{A,i}, Ch_{A,i}, Cmt_{B,i}, Ch_{B,i}^{(d)}, Rsp_i)$ を実行する。そして、検証者アルゴリズム V は、 $i = 1, \dots, L$ について全て受理（出力 1 ）されれば検証が成立したものとみなす。

[0234] 以上、本手法 # 2 の方式を SSH10b 公開鍵認証方式に適用した場合の対話プロトコル（変形例）について説明した。なお、上記の説明においては、2つのチャレンジの組を検証者から証明者に送っているが、例えば、2つ目のチャレンジが、 $Ch_i^{(1)} = Ch_i^{(0)} + 1 \pmod{2}$ であると予め決めてしまってもよい。この場合、片方のチャレンジの組を送らなくて済むため、通信量を削減することが可能になる。

[0235] 以上、本開示の第2実施形態について説明した。上記説明においては、SSH10a 公開鍵認証方式、SSH10b 公開鍵認証方式を例に挙げて述べたが、本手法 # 2 の適用範囲はこれに限定されない。SSH10a 公開鍵認証方式や SSH10b 公開鍵認証方式の変形例、及びその他の公開鍵認証方式に対しても適用が可能である。例えば、上記の例では公開鍵を (F, y) に設定しているが、 F が秘密鍵によらないパラメータであるため、この F を証明者毎に設定するのではなく、システム全体に共通するパラメータとして

もよい。この場合、個別に公開すべき公開鍵は y だけとなり、公開鍵のサイズが小さくなる。また、上記の対話プロトコルにおいては、 $i = 1, \dots, L$ について個別に乱数 $\alpha_1, \dots, \alpha_L$ を選択しているが、これらの乱数を1つの共通化してもよい。この場合、チャレンジ $Ch_{A, i}$ を送る際の通信コストを削減することが可能になる。

[0236] <4 : 補足>

ここで、本手法 # 1、# 2 について説明を補足する。

[0237] [4 - 1 : 方式の拡張について]

これまでは多次多変数連立方程式の求解問題に基づく公開鍵認証方式について説明してきた。しかし、本手法 # 1、# 2 の考え方は、その他の公開鍵認証方式に対しても同様に適用することが可能である。そこで、一例として、Syndrome Decoding 問題に基づく公開鍵認証方式への適用方法、Constrained Linear Equations 問題に基づく公開鍵認証方式への適用方法、Permuted Kernel 問題に基づく公開鍵認証方式への適用方法、代数曲面上の求セクション問題に基づく公開鍵認証方式への適用方法、Permuted Perceptrons 問題に基づく公開鍵認証方式への適用方法について紹介する。

[0238] (Syndrome Decoding 問題への適用)

Syndrome Decoding 問題に基づく公開鍵認証方式は、例えば、「Jacques Stern, A New Identification Scheme Based on Syndrome Decoding, CRYPTO 1993, p13-21.」や「Jacques Stern, A New Paradigm for Public Key Identification, IEEE Transactions on Information Theory, 1996, p13-21.」に記載されている。これによると、この公開鍵認証方式には、3パスの方式と5パスの方式がある。

[0239] 3パスの方式は、SSH10a 公開鍵認証方式と同様に、検証パターンが

3パターンあり、そのうちの2パターンまでは対応可能な偽証アルゴリズムが存在する。また、5パスの方式も、SSH10b公開鍵認証方式と同様に、検証パターンが2パターンあり、そのうちの1パターンまでには対応可能な偽証アルゴリズムが存在する。そのため、3パスの方式にはSSH10a公開鍵認証方式について先に説明した方法が、5パスの方式にはSSH10b公開鍵認証方式について先に説明した方法が適用可能である。

[0240] (Constrained Linear Equations問題への適用)

Constrained Linear Equations問題に基づく公開鍵認証方式は、例えば、「Jacques Stern, Designing Identification Schemes with Keys of Short Size, CRYPTO 1994, p 164-173.」に記載されている。これによると、この公開鍵認証方式にも3パスの方式と5パスの方式がある。

[0241] 3パスの方式は、SSH10a公開鍵認証方式と同様に、検証パターンが3パターンあり、そのうちの2パターンまでは対応可能な偽証アルゴリズムが存在する。また、5パスの方式も、SSH10b公開鍵認証方式と同様に、検証パターンが2パターンあり、そのうちの1パターンまでには対応可能な偽証アルゴリズムが存在する。そのため、3パスの方式にはSSH10a公開鍵認証方式について先に説明した方法が、5パスの方式にはSSH10b公開鍵認証方式について先に説明した方法が適用可能である。

[0242] (Permuted Kernel問題への適用)

Permuted Kernel問題に基づく公開鍵認証方式は、例えば、「Adi Shamir, An Efficient Identification Scheme Based on Permuted Kernels (Extended Abstract), CRYPTO 1989, p 606-609.」に記載されている。これによると、この公開鍵認証方式には、検証パターンが2パターンあり、そのうち1パターン

までには対応可能な偽証アルゴリズムが存在する。そのため、この公開鍵認証方式にはSSH10b公開鍵認証方式について先に説明した方法が適用可能である。

[0243] (代数曲面上の求セクション問題への適用)

代数曲面上の求セクション問題に基づく公開鍵認証方式には、3パスと5パスの方式がある（例えば、特願2010-125026号などを参照）。3パスの方式は、SSH10a公開鍵認証方式と同様に、検証パターンが3パターンあり、そのうちの2パターンまでは対応可能な偽証アルゴリズムが存在する。また、5パスの方式も、SSH10b公開鍵認証方式と同様に、検証パターンが2パターンあり、そのうちの1パターンまでには対応可能な偽証アルゴリズムが存在する。そのため、3パスの方式にはSSH10a公開鍵認証方式について先に説明した方法が、5パスの方式にはSSH10b公開鍵認証方式について先に説明した方法が適用可能である。

[0244] (Permuted Perceptrons問題への適用)

Permuted Perceptrons問題に基づく公開鍵認証方式は、例えば、「David Pointcheval, A New Identification Scheme Based on the Perceptrons Problem, EUROCRYPT 1995, p319-328.」や「David Pointcheval and Guillaume Poupard, A New NP-Complete Problem and Public-Key Identification, Des. Codes Cryptography, 2003, p5-31.」に記載されている。これによると、この公開鍵認証方式には、3パスの方式と5パスの方式がある。

[0245] 3パスの方式は検証パターンが4パターンあり、そのうち3パターンまでには対応可能な偽証アルゴリズムが存在する。一方、5パスの方式は検証パターンが3パターンあり、そのうち2パターンまでには対応可能な偽証アルゴリズムが存在する。そのため、SSH10a公開鍵認証方式やSSH10

b 公開鍵認証方式と同様に、この公開鍵認証方式に対しても本手法# 1、# 2の技術を適用することが可能である。但し、この公開鍵認証方式に対して本手法# 1を適用する際には以下の変更が必要になる。

- [0246] 検証パターンが k パターンある方式に対し、秘密鍵を $L = k$ 個に多重化して本手法# 1を適用する場合、 $i = 1, \dots, L$ から1個だけ回答しない i （以下、 i^* ）を次のように選択することが必要になる。つまり、 i^* の選択方法は、 $Ch_{i_0} = Ch^*$ ならば $i^* \leftarrow i_0$ が選択され、 $Ch_{i_0} \neq Ch^*$ ならば $i^* \in \{1, \dots, L\} \setminus \{i_0\}$ からランダムに i^* が選択されるというものになる。
- [0247] また、検証パターンが k パターンある方式に対し、秘密鍵を L （ $L < k$ ）個に多重化して本手法# 1を適用する場合、 $i = 1, \dots, L$ から1個だけ回答しない i （以下、 i^* ）を次のように選択することが必要になる。つまり、 i^* の選択方法は、 $Ch_{i_0} = Ch^*$ ならば $i^* \leftarrow i_0$ が選択され、 $Ch_{i_0} \neq Ch^*$ ならば確率 $(1 - 1/k)^{-1} (1/L - 1/k)$ で $i^* \leftarrow i_0$ が選択され、かつ、確率 $1 - (1 - 1/k)^{-1} (1/L - 1/k)$ で $i^* \in \{1, \dots, L\} \setminus \{i_0\}$ からランダムに i^* が選択されるというものになる。
- [0248] 上記の方法によれば検証者がどのような検証パターンを要求してきても i^* が $1/L$ の確率で $1, \dots, L$ の値をとることになり、 i_0 が $1, \dots, L$ のいずれであったかという情報が完全に秘匿される。
- [0249] 以上説明したように、本手法# 1、# 2は、SSH10a公開鍵認証方式、SSH10b公開鍵認証方式に限らず、様々な公開鍵認証方式に適用可能である。
- [0250] [4-2：非対話型の公開鍵認証方式について]

上記の本手法# 1、# 2を適用した公開鍵認証方式は、検証者が乱数しか送信しない方式であるため、1パス（非対話型）の公開鍵認証方式に変形することが可能である。例えば、 N 回の並列的繰り返し構成において検証者との間でやりとりした内容にハッシュ関数を適用したものを、検証者から送信される乱数の代わりに用いればよい。また、ハッシュ関数の引数に、証明者

が選んだ乱数を加えてもよい。このように、検証者が乱数を選択する代わりに証明者自身がハッシュ関数を利用することで、検証者が乱数を選択した場合と同じ振る舞いが実現される。但し、理想的なハッシュ関数を利用することが望ましい。また、偽証を不可能にするために繰り返し回数 N を十分に大きくとることが望ましい。このような工夫により、非対話型の公開鍵認証方式に変形することが可能になる。

[0251] [4-3: 通信量の削減方法について]

ここで、図19、図20を参照しながら、対話プロトコルにおける通信量の削減方法について簡単に説明を補足する。

[0252] 既に説明したように、SSH10a 公開鍵認証方式の1パス目において、証明者から検証者にメッセージの組 (c_1, c_2, c_3) が送られる。但し、上記説明においては Cmt という表現を用いていた。 N 回の並列的繰り返し構成の場合、このメッセージの組は、図19に示すように、 $(c_{1,1}, c_{1,2}, c_{1,3}, \dots, c_{N,1}, c_{N,2}, c_{N,3})$ となり、通信量が非常に大きくなってしまう。そこで、本件発明者は、これらのメッセージを1つのハッシュ値にまとめて送る構成を考案した。

[0253] 図19に示すように、この構成を適用すると、1パス目で送るメッセージが1つのハッシュ値だけとなり、大幅に通信量を削減することが可能になる。但し、このハッシュ値、及び証明者から送られるチャレンジに対する回答から復元できないハッシュ値については、回答と併せて証明者から送ってもらうようにする。この構成によると、 N 回の並列的繰り返し構成の場合、送るべき情報の数を $2N-1$ 個削減することが可能になる。図20に示すように、SSH10b 公開鍵認証方式に対しても同様の構成を適用することが可能である。この場合、送るべき情報の数を $N-1$ 個削減することが可能になる。

[0254] 以上、本手法#1、#2について説明を補足した。

[0255] <5: ハードウェア構成>

上記の各アルゴリズムは、例えば、図21に示す情報処理装置のハードウ

エア構成を用いて実行することが可能である。つまり、当該各アルゴリズムの処理は、コンピュータプログラムを用いて図21に示すハードウェアを制御することにより実現される。なお、このハードウェアの形態は任意であり、例えば、パーソナルコンピュータ、携帯電話、PHS、PDA等の携帯情報端末、ゲーム機、接触式又は非接触式のICチップ、接触式又は非接触式のICカード、又は種々の情報家電がこれに含まれる。但し、上記のPHSは、Personal Handy-phone Systemの略である。また、上記のPDAは、Personal Digital Assistantの略である。

[0256] 図21に示すように、このハードウェアは、主に、CPU902と、ROM904と、RAM906と、ホストバス908と、ブリッジ910と、を有する。さらに、このハードウェアは、外部バス912と、インターフェース914と、入力部916と、出力部918と、記憶部920と、ドライブ922と、接続ポート924と、通信部926と、を有する。但し、上記のCPUは、Central Processing Unitの略である。また、上記のROMは、Read Only Memoryの略である。そして、上記のRAMは、Random Access Memoryの略である。

[0257] CPU902は、例えば、演算処理装置又は制御装置として機能し、ROM904、RAM906、記憶部920、又はリムーバブル記録媒体928に記録された各種プログラムに基づいて各構成要素の動作全般又はその一部を制御する。ROM904は、CPU902に読み込まれるプログラムや演算に用いるデータ等を格納する手段である。RAM906には、例えば、CPU902に読み込まれるプログラムや、そのプログラムを実行する際に適宜変化する各種パラメータ等が一時的又は永続的に格納される。

[0258] これらの構成要素は、例えば、高速なデータ伝送が可能なホストバス908を介して相互に接続される。一方、ホストバス908は、例えば、ブリッジ910を介して比較的データ伝送速度が低速な外部バス912に接続され

る。また、入力部 916 としては、例えば、マウス、キーボード、タッチパネル、ボタン、スイッチ、及びレバー等が用いられる。さらに、入力部 916 としては、赤外線やその他の電波を利用して制御信号を送信することが可能なリモートコントローラが用いられることもある。

[0259] 出力部 918 としては、例えば、CRT、LCD、PDP、又はELD等のディスプレイ装置、スピーカ、ヘッドホン等のオーディオ出力装置、プリンタ、携帯電話、又はファクシミリ等、取得した情報を利用者に対して視覚的又は聴覚的に通知することが可能な装置である。但し、上記のCRTは、Cathode Ray Tubeの略である。また、上記のLCDは、Liquid Crystal Displayの略である。そして、上記のPDPは、Plasma Display Panelの略である。さらに、上記のELDは、Electro-Luminescence Displayの略である。

[0260] 記憶部 920 は、各種のデータを格納するための装置である。記憶部 920 としては、例えば、ハードディスクドライブ（HDD）等の磁気記憶デバイス、半導体記憶デバイス、光記憶デバイス、又は光磁気記憶デバイス等が用いられる。但し、上記のHDDは、Hard Disk Driveの略である。

[0261] ドライブ 922 は、例えば、磁気ディスク、光ディスク、光磁気ディスク、又は半導体メモリ等のリムーバブル記録媒体 928 に記録された情報を読み出し、又はリムーバブル記録媒体 928 に情報を書き込む装置である。リムーバブル記録媒体 928 は、例えば、DVDメディア、Blu-rayメディア、HD DVDメディア、各種の半導体記憶メディア等である。もちろん、リムーバブル記録媒体 928 は、例えば、非接触型ICチップを搭載したICカード、又は電子機器等であってもよい。但し、上記のICは、Integrated Circuitの略である。

[0262] 接続ポート 924 は、例えば、USBポート、IEEE1394ポート、SCSI、RS-232Cポート、又は光オーディオ端子等のような外部接

続機器 930 を接続するためのポートである。外部接続機器 930 は、例えば、プリンタ、携帯音楽プレーヤ、デジタルカメラ、デジタルビデオカメラ、又は IC レコーダ等である。但し、上記の USB は、Universal Serial Bus の略である。また、上記の SCSI は、Small Computer System Interface の略である。

[0263] 通信部 926 は、ネットワーク 932 に接続するための通信デバイスであり、例えば、有線又は無線 LAN、Bluetooth（登録商標）、又は WUSB 用の通信カード、光通信用のルータ、ADSL 用のルータ、又は接触又は非接触通信用のデバイス等である。また、通信部 926 に接続されるネットワーク 932 は、有線又は無線により接続されたネットワークにより構成され、例えば、インターネット、家庭内 LAN、赤外線通信、可視光通信、放送、又は衛星通信等である。但し、上記の LAN は、Local Area Network の略である。また、上記の WUSB は、Wireless USB の略である。そして、上記の ADSL は、Asymmetric Digital Subscriber Line の略である。

[0264] <6：まとめ>

最後に、本開示の実施形態に係る技術内容について簡単に纏める。ここで述べる技術内容は、例えば、PC、携帯電話、携帯ゲーム機、携帯情報端末、情報家電、カーナビゲーションシステム等、種々の情報処理装置に対して適用することができる。

[0265] 上記の情報処理装置の機能構成は次のように表現することができる。当該情報処理装置は、次のような鍵保持部と、対話プロトコル実行部とを有する。当該鍵保持部は、 L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持するものである。また、上記の対話プロトコル実行部は、 $(L - 1)$ 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行するものである。また、前記対話プロトコル実行部は、前記検証者との間で前記対話プロトコルを実行する際に、

どの秘密鍵 s_i を使用したかを当該検証者に知られないようにする。

[0266] 上記のように、秘密鍵 s_i を多重化し、対話プロトコルを実行する際に、そのうちの一部の秘密鍵を利用し、さらに、どの秘密鍵 s_i が利用されたのかを対話プロトコルの中で知られないようにすることにより、並列的繰り返し構成の場合にも、能動的攻撃に対する安全性レベルを保証することが可能になる。つまり、偽証者が検証者に成りすまして認証に利用される秘密鍵 s_i の情報を得ようとしても、対話プロトコルの中で、どの秘密鍵 s_i が利用されたかが検証者にも分からない。つまり、任意の回数だけ対話プロトコルを実行できる状況にあっても、秘密鍵 s_i の情報が証明者から漏れないため、能動的攻撃に対する安全性レベルを保証することができるのである。

[0267] (備考)

上記の情報処理装置は、証明者側及び検証者側の認証装置の一例である。また、上記のROM 904、RAM 906、記憶部 920、リムーバブル記録媒体 928 は、鍵保持部の一例である。なお、この鍵保持部に保持される秘密鍵 s_i 、公開鍵 y_i は、鍵生成アルゴリズム Gen により生成される。また、証明者アルゴリズム P、検証者アルゴリズム V は、対話プロトコル実行部の一例である。なお、実際には、証明者アルゴリズム P、検証者アルゴリズム V が CPU 902 の機能により実行されることにより対話プロトコル実行部の機能が実現される。また、証明者アルゴリズム P によって実現される機能により、チャレンジ受信部、チャレンジ選択部、回答生成部、回答送信部、メッセージ送信部の機能が実現される。なお、情報の送受信には、通信部 926 の機能が利用される。

[0268] 以上、添付図面を参照しながら本開示の好適な実施形態について説明したが、本開示は係る例に限定されないことは言うまでもない。当業者であれば、特許請求の範囲に記載された範疇内において、各種の変更例または修正例に想到し得ることは明らかであり、それらについても当然に本開示の技術的範囲に属するものと了解される。

符号の説明

[0269] G e n 鍵生成アルゴリズム
 P 証明者アルゴリズム
 V 検証者アルゴリズム

請求の範囲

[請求項1]

L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持部と、

$(L - 1)$ 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行する対話プロトコル実行部と、

を備え、

前記対話プロトコル実行部は、

前記検証者から L 個のチャレンジ Ch_i を受信するチャレンジ受信部と、

前記チャレンジ受信部により受信された L 個のチャレンジ Ch_i の中から $(L - 1)$ 個のチャレンジ Ch_i を任意に選択するチャレンジ選択部と、

前記秘密鍵 s_i を用いて、前記チャレンジ選択部により選択された $(L - 1)$ 個のチャレンジ Ch_i のそれぞれに対する $(L - 1)$ 個の回答 Rsp_i を生成する回答生成部と、

前記回答生成部により生成された $(L - 1)$ 個の回答 Rsp_i を前記検証者に送信する回答送信部と、

を含む、

認証装置。

[請求項2]

前記対話プロトコル実行部は、

前記検証者に対し、 L 個の前記秘密鍵 s_i のそれぞれに対応するメッセージ Cmt_i を送信するメッセージ送信部をさらに含み、

前記チャレンジ受信部は、前記メッセージ送信部により送信された各メッセージ Cmt_i に応じて、前記検証者により k 通り ($k \geq 2$) の検証パターンの中から選択された検証パターンを示すチャレンジ Ch_i を受信する、

請求項 1 に記載の認証装置。

[請求項3] 前記メッセージ $C m t_i$ が $C m t_i = (c_{i,1}, \dots, c_{i,N})$ である場合、

前記メッセージ送信部は、一方向性関数 H を用いて新たなメッセージ $C m t' = H(C m t_1, \dots, C m t_L)$ を算出して当該メッセージ $C m t'$ を前記検証者に送信し、

前記回答送信部は、前記回答 $R s p_i$ と共に、当該回答 $R s p_i$ を利用しても前記検証者が復元できない前記メッセージ $C m t_i$ の要素を送信する、

請求項 2 に記載の認証装置。

[請求項4] 前記鍵保持部は、前記 L 個の秘密鍵 s_i のうち、1つの秘密鍵 s_{i_0} ($1 \leq i_0 \leq L$) を保持しないようにし、

前記対話プロトコル実行部は、前記対話プロトコルの中で実行される前記秘密鍵 s_{i_0} に関する処理を偽証アルゴリズムに基づいて実行する、

請求項 2 に記載の認証装置。

[請求項5] L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持部と、

検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $C h_i^{(j)}$ ($j = 1 \sim Q$) を受信するチャレンジ受信部と、

前記チャレンジ受信部により受信された Q 組の L 個のチャレンジ $C h_i^{(j)}$ の中から 1 組の L 個のチャレンジ $C h_i^{(j)}$ を任意に選択するチャレンジ選択部と、

前記秘密鍵 s_i を用いて、前記チャレンジ選択部により選択された L 個のチャレンジ $C h_i^{(j)}$ のそれぞれに対する L 個の回答 $R s p_i$ を生成する回答生成部と、

前記回答生成部により生成された L 個の回答 $R s p_i$ を前記検証者

に送信する回答送信部と、

を備える、

認証装置。

[請求項6]

前記対話プロトコル実行部は、

前記検証者に対し、 L 個の前記秘密鍵 s_i のそれぞれに対応するメッセージ $C m t_i$ を送信するメッセージ送信部をさらに含み、

前記チャレンジ受信部は、前記メッセージ送信部により送信された各メッセージ $C m t_i$ に応じて、前記検証者により k 通り ($k \geq 2$) の検証パターンの中から選択された検証パターンを示すチャレンジ $C h_i^{(j)}$ を受信する、

請求項5に記載の認証装置。

[請求項7]

前記メッセージ $C m t_i$ が $C m t_i = (c_{i,1}, \dots, c_{i,N})$ である場合、

前記メッセージ送信部は、一方向性関数 H を用いて新たなメッセージ $C m t' = H(C m t_1, \dots, C m t_L)$ を算出して当該メッセージ $C m t'$ を前記検証者に送信し、

前記回答送信部は、前記回答 $R s p_i$ と共に、当該回答 $R s p_i$ を利用しても前記検証者が復元できない前記メッセージ $C m t_i$ の要素を送信する、

請求項6に記載の認証装置。

[請求項8]

L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を生成することと、

$(L - 1)$ 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行することと、
を含み、

前記実行することは、

前記検証者から L 個のチャレンジ $C h_i$ を受信することと、
受信された L 個のチャレンジ $C h_i$ の中から $(L - 1)$ 個のチャレンジ $C h_i$ を任意に選択することと、
前記秘密鍵 s_i を用いて、選択された $(L - 1)$ 個のチャレンジ $C h_i$ のそれぞれに対する $(L - 1)$ 個の回答 $R s p_i$ を生成することと、
生成された $(L - 1)$ 個の回答 $R s p_i$ を前記検証者に送信することと、
を含む、
認証方法。

[請求項9]

L 個 ($L \geq 2$) の秘密鍵 s_i ($i = 1 \sim L$)、及び n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持機能と、

$(L - 1)$ 個の $y_i = F(s_i)$ を満たす秘密鍵 s_i を知っていることを証明するための対話プロトコルを検証者との間で実行する対話プロトコル実行機能と、

をコンピュータに実現させるためのプログラムであり、

前記対話プロトコル実行機能は、

前記検証者から L 個のチャレンジ $C h_i$ を受信するチャレンジ受信機能と、

前記チャレンジ受信機能により受信された L 個のチャレンジ $C h_i$ の中から $(L - 1)$ 個のチャレンジ $C h_i$ を任意に選択するチャレンジ選択機能と、

前記秘密鍵 s_i を用いて、前記チャレンジ選択機能により選択された $(L - 1)$ 個のチャレンジ $C h_i$ のそれぞれに対する $(L - 1)$ 個の回答 $R s p_i$ を生成する回答生成機能と、

前記回答生成機能により生成された $(L - 1)$ 個の回答 $R s p_i$ を前記検証者に送信する回答送信機能と、

を含む、

プログラム。

[請求項10] L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式の組 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を生成することと、

検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $Ch_i^{(j)}$ ($j = 1 \sim Q$) を受信することと、

受信された Q 組の L 個のチャレンジ $Ch_i^{(j)}$ の中から 1 組の L 個のチャレンジ $Ch_i^{(j)}$ を任意に選択することと、

前記秘密鍵 s_i を用いて、選択された L 個のチャレンジ $Ch_i^{(j)}$ のそれぞれに対する L 個の回答 Rsp_i を生成することと、

生成された L 個の回答 Rsp_i を前記検証者に送信することと、
を含む、

認証方法。

[請求項11] L 個の秘密鍵 s_i ($i = 1 \sim L$)、 n 次多変数多項式 F ($n \geq 2$) について $y_i = F(s_i)$ を満たす L 個の公開鍵 y_i を保持する鍵保持機能と、

検証者から Q 組 ($Q \geq 2$) の L 個のチャレンジ $Ch_i^{(j)}$ ($j = 1 \sim Q$) を受信するチャレンジ受信機能と、

前記チャレンジ受信機能により受信された Q 組の L 個のチャレンジ $Ch_i^{(j)}$ の中から 1 組の L 個のチャレンジ $Ch_i^{(j)}$ を任意に選択するチャレンジ選択機能と、

前記秘密鍵 s_i を用いて、前記チャレンジ選択機能により選択された L 個のチャレンジ $Ch_i^{(j)}$ のそれぞれに対する L 個の回答 Rsp_i を生成する回答生成機能と、

前記回答生成機能により生成された L 個の回答 Rsp_i を前記検証者に送信する回答送信機能と、

をコンピュータに実現させるためのプログラム。