Σプロトコルとその応用

尾形わかは

暗号技術を用いてフェアに目的を達成するプロトコルを，暗号プロトコルという．本稿では，暗号プロトコルの中で，「知識の証明」とも呼ばれるΣプロトコルに焦点を絞り，仕組み，具体例，考え方，さまざまな応用などについて解説する．

キーワード：暗号プロトコル，Σプロトコル，零知識証明，認証

1. はじめに

プロトコルとは，幾人かの者が通信しあうことで何らかの目的を達成するやり方であり，インターネットで通信を行う手順もプロトコルと呼ばれる．一般のプロトコルでは，送受信する者がすべて決められた手順どおりに動くことを想定している．しかし，そのような想定は常に成り立つものではなく，いったん手順を無視した者が現れると，意図しない結果が出たり，真面目にやっている者が不利益を被ったりする．そこで暗号技術の出番である．

暗号技術を用いることにより，フェアに目的を達成できるプロトコルを総称して，暗号プロトコルと呼ぶ．目的とは，どんなものを指すのだろうか．単純な例ではチャットでじゃんけんがある．普通に行うと「後出し」ができてしまうが，暗号技術を使うとフェアなじゃんけんが可能になる．大規模な例では，オンライン無記名投票がある．投票を受け付けるサーバの管理者にも各投票者の投票内容がわからないように，また開票結果が正しいことを誰もが確認できるように，暗号技術を使うことができる．

本稿では暗号プロトコルの中でも応用範囲が広い，「Σプロトコル」と呼ばれる種類の暗号プロトコルに焦点を絞ることにする．

Σプロトコルは「知識の証明」とも呼ばれるもので，「秘密を知っていることを相手に証明するが，秘密を教えない」という目的を達成するための暗号プロトコルである．「秘密を知っていることを相手に証明するが，秘密を教えない」というのは，そんなことができるのだろうかと思われるかもしれないが，暗号技術を使うとできてしまう．

では，そんなプロトコルがいったい何に役に立つのかといえば，典型的な応用例は本人認証である．Σプロトコルを利用すれば，パスワードを知っていることを示すことで本人性を確認でき，パスワードは教えないことで検証者でも成りすましができない．このほか，大規模なプロトコルの中でツールとして用いられたり，変換してディジタル署名として利用するなど，様々な応用がある．本稿では，Σプロトコルの具体例やいくつかの応用を紹介する．

2. Σプロトコルの定義

証明者と呼ばれる人と，検証者と呼ばれる人がいて，証明者が「自分はある秘密を知っている」ということを通信を使って検証者に示したいとしよう．これを単純に実行するためには次のようにすれば良いだろう．

(1) 証明者は秘密を検証者へ送る．
(2) 検証者は，秘密が正しいことを何らかの方法で確認し，正しければ証明を受理（accept）し，正しくなければ却下（reject）する．

むろん検証者は秘密を知ることができ，証明者にとって好ましくない場合がある．

Σプロトコルとは，「自分はある秘密を知っている」ということを「秘密は漏らさずに」証明するものである．もう少し厳密には，以下の4つの性質を持つプロトコルである．

＊ **通信モデルの条件**：プロトコルは3回の通信で終わる．つまり，まず証明者が検証者へ情報Rを送り，次いで検証者が証明者へ情報cを送り，最後に証明者が検証者へ情報zを送る．これを3ムーブという．さらに，検証者が送る情報cは，常にランダムビット列であり，チャレンジ

おがた　わかは
東京工業大学 大学院イノベーションマネジメント研究科
〒152-8552 目黒区大岡山2-12-1

と呼ばれる．

* **完全性**：証明者が秘密を知っていて，証明者と検証者が決められたプロトコル通りに振る舞えば，検証者は最後にacceptする．
* **健全性**：証明者が秘密を知らない場合，証明者がどのように振る舞おうとも，検証者が騙される確率，つまり最後にacceptする確率はある一定以下である．厳密には，検証者が送るチャレンジcがtビットであるとき，騙せる確率は$1/2^t$以下である．この確率をsoundness確率と呼ぶ．
* **条件付零知識性**：検証者がプロトコルに従っている限りは，証明者から検証者へ何の情報も漏れない．

最初の性質は，通信が非常にシンプルであるという性質だと思ったらよい．検証者がランダムビット列cのみを送り返すという条件は，検証者の負荷が小さいだけでなく，応用のしやすさの理由となっている．

チャレンジcを100ビット（t＝100）とすれば，soundness確率は$1/2^{100}$となり，無視できるほど小さい．したがって，証明者は検証者を騙せない，と考えて良い．

4つ目の性質が，「秘密を漏らさない」ことを保障している．さらにいえば，秘密そのものだけでなく，秘密から派生する補助的な情報（秘密の最下位ビットなど）ですら漏らさないことを示している．

3. ZKIPとΣプロトコルの生い立ち

Σプロトコルと非常に近い関係にあるものに，零知識証明（Zero-Knowledge Interactive Proof，以後ZKIP）がある．ここでは，ΣプロトコルとZKIPとの違いや，これらの歴史について簡単に述べる．

ZKIPは，1985年に提案された概念であり，「ある命題を，全く知識を漏らさずに通信を使って証明する」ものである[1]．Σプロトコルとの違いは，主に次の4点である．

* 証明すべき事柄が，「証明者の知識」だけでなく，「何らかの命題」に一般化されている．
* 通信モデルの条件がない．通信の回数制限もなく，検証者はランダムビット列以外の情報を送ることができる．
* 検証者がプロトコルに従っていなかったとしても，証明者から検証者へ何の情報も漏れない．
* 証明者は，無限の力を持っており何でも計算できるとする．つまり，何でも計算できたとしても，検証者を騙せない．

ZKIPの初期の研究では，「命題がNP問題[1]であれば，必ずZKIPを構成可能である」ことや，NP問題であれば，通信が2往復，つまり検証者（V）から証明者（P）へ，PからVへ，VからPへ，PからVへ，の4ムーブであるZKIPが構成可能であることが証明された．さらに，3ムーブではNP問題に対するZKIPが構成できないこと，すなわち，4ムーブが最も効率的であることも証明された．

一方で，ZKIPの概念を実用的な用途に利用するための方法も発表された．まず，フィアットとシャミアは，平方根の知識のZKIPを利用した個人認証方式や，ディジタル署名への応用方法を提案した[2]．その後，離散対数の知識を効率的に証明するプロトコルがシュノアによって提案され[3]，これを元に多くのプロトコルが作られていった．これらのプロトコルは，完全性，健全性は持つものの，「検証者がプロトコルに従っていなかったとしても，証明者から検証者へは，何の情報も漏れない」という性質は持たないため，厳密な意味でのZKIPではない．しかし，実用化に有利な共通的な性質を持つため，これらを総称してΣプロトコルと呼ぶことにし，ZKIPとは別の概念として研究対象として確立した．

4. FS法と個人認証方式

Σプロトコルの1つ目の例として，古典的なΣプロトコルであるフィアット・シャミアプロトコル（FS法）を紹介する．FS法は，自然数aとNが与えられたときに，証明者が「Nを法とするaの平方根を知っている」ことを示すΣプロトコルである[2]（つまり，$a=x^2 \bmod N$を満たすxを知っていること）．

まずは，FS法の基本方式を見てみよう．証明者（prover）をP，検証者（verifier）をVと呼ぶことにし，Pは$a=x^2 \bmod N$となるxを知っていると仮定する．PとVは次のように情報を送りあう（図1参照）．

(1) Pはランダムにrを選び$R=r^2 \bmod N$を計算し，

[1] NP問題とは，与えられた入力がある特定の性質をもつかどうかを判断する問題であって，
* 入力からそれを判断するのは指数時間かかる
* ある証拠となる情報があれば，その性質を持つことを判断するのは多項式時間で計算可能

という条件を持つものである．

[2] Nが2つの素数の積である場合には，Nを法とするaの平方根xを求めることは非常に困難であり，平方剰余を求める問題はNP問題であると考えられている．

Vへ R を送信.

(2) V はランダムに1ビット c を選び，P へ c を送信.

(3) P は $z=rx^c \bmod N$ を計算し，V へ z を送信.

(4) V は検証式 $z^2=Ra^c \pmod N$ が成り立てば accept，そうでなければ reject.

このプロトコルでは，チャレンジ c は1ビットであり，平方根 x を知らない P が V をだませる確率（soundness 確率）は 1/2 であり，十分安全とはいえない．Soundness 確率をもっと小さく，例えば $1/2^T$ としたければ，この基本方式を独立に T 回並行して実行すればよい．つまり，(1)では P は T 個の r を選び，T 個の R を計算して V へ送信する，というように実行するのである．T＝50 程度とすれば十分安全であるといえるが，残念ながらあまり効率的とはいえない．

個人認証への応用：FS 法は，個人認証方式として提案された．個人認証方式とは，各ユーザが何らかの情報を検証サーバへ登録しておき，必要に応じてユーザがある特定のユーザ本人であることを証明するシステムである．FS 法を用いた認証システムでは，ユーザが自分の秘密として平方根 x を持ち，認証サーバはユーザの登録情報として（N, a）を保管しておく．ユーザが自分の本人性を証明したい場合は，a の平方根を知っていることを FS 法を用いて証明すればよい．

ただ，先にも書いたように，soundness 確率を十分小さくしようとすると，通信量が大きくなってしまう．そこで，秘密情報を複数化して通信量を削減するなどの効率化がされている．

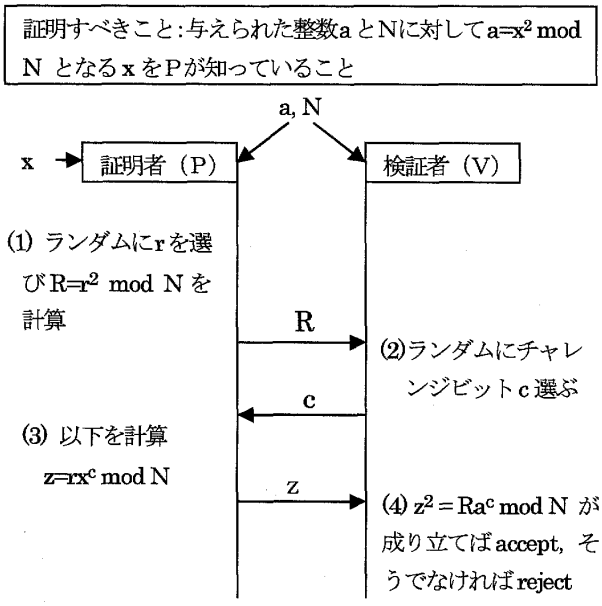

図1　FS 法（基本方式）

コラム

FS 法を例に，完全性や健全性，零知識性がどのように保障されるのかを説明しよう．

完全性：証明者 P が a の平方根を知っており，P と V が共にプロトコルに従うならば，最終的に V が accept することはすぐに確認できる．

健全性：P が平方根を知らずに，V に accept といわせることはできるか？　P がどんな戦略で V をだますにしても，P は(1)で何らかの R を V へ送らなければならない．チャンレンジとして c＝0 が選ばれたときに検証式が成り立つためには，P は(3)で R の平方根 r を送る必要がある．一方で c＝1 が選ばれたときに検証式が成り立つためには，P は(3)で Ra の平方根 rx を送る必要がある．つまり，どちらの c が選ばれても上手く V をだませるためには，P は r と rx を両方とも知っていなければならない．これは x そのものを知っていることに他ならない．裏を返せば，x を知らなければ2通りのチャレンジの一方に対してのみ，(3)で上手く答えることができる．結局，P が V をだませる確率（soundness 確率）は高々 1/2 である．

条件付零知識性：V は FS 法を行うことによって何か知識を得ているか？　P から V へ何らかの知識が漏れているとしたら，その知識は P から V へ送られる情報に乗っているはずである．そこで，PV 間の通信内容と同じような情報を V が自分で計算できるならば，V は何も知識を受取っていない，と考える．FS 法の場合，PV 間の通信内容は2つのパターン：〈ランダムな平方剰余数 R, c＝0, R の平方根 r〉〈ランダムな平方剰余数 R, c＝1, aR の平方根 rx〉がある．1つ目のパターンは，V が自分で計算できることは明確である（r を選び R を計算すれば良い）．2つめのパターンは，V が元々知らない情報である x を含んでいるため，V は知識が増えたように見える．しかし，〈ランダムな R と，aR の平方根〉というのは見方を変えると〈z^2/a と，ランダムな z〉であり，これまた V が自分で作ることのできる情報である．したがって V は何の知識も得ていないことになる．

一般に，〈チャレンジが先にわかれば，検証式を満たす R と z を誰でも計算できる〉ときに条件付零知識性を持つといえる．

5. 離散対数の知識の証明

pが大きな素数であるとき，gとxから$g^x \bmod p$を計算することは容易であるが，逆にgとX=$g^x \bmod p$から離散対数であるxを計算することは非常に困難である．この性質はエルガマル暗号をはじめ，多くの暗号システムで利用されている．

エルガマル暗号では，X(=$g^x \bmod p$)を生成したユーザのみが離散対数xを知っていることを利用している．そこで「離散対数を知っていることを示す知識の証明」を作成すれば，エルガマル暗号と相性の良い個人認証方式ができる．FS法を真似てこれを作ることも可能だが，1989年にシュノアによって効率の面で画期的なΣプロトコルが考案された．

シュノアのプロトコルを図2に示す（わかりやすさのために多少変更している）．証明者はXの離散対数，つまりX=$g^x \bmod p$となるxを知っているものとする．FS法との大きな違いは，チャレンジcのビット長である．FS法（基本方式）ではcが1ビットであったため，検証者がだまされるsoundness確率が1/2であり，そのため何度も並行して実行する必要があった．しかし図2のプロトコルではcを任意の長さのビット列とすることができ，例えばcを100ビットにするだけでsoundness確率は$1/2^{100}$になる．これにより，通信量を画期的に節約できる．

シュノアのプロトコルと同様にして，離散対数に関連した等式について何らかの知識を証明する様々なプロトコルが容易に構成可能である．

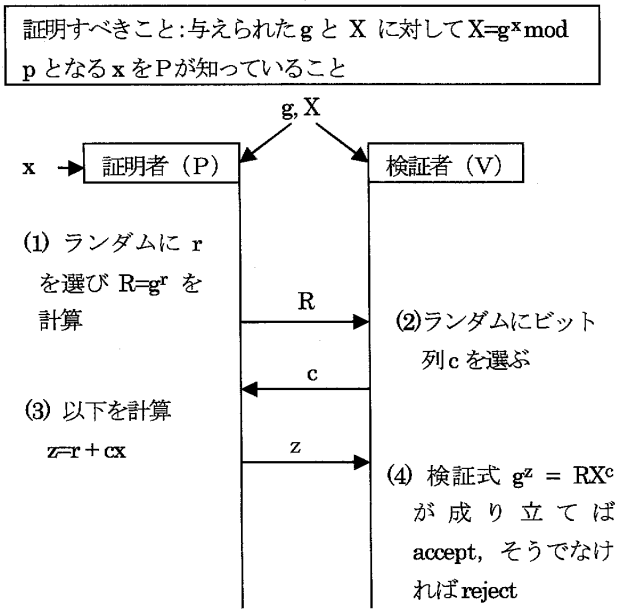

図2　離散対数の知識を示すプロトコル

6. FS変換とディジタル署名への応用

Σプロトコルは，ZKIPに比べて安全性の面では劣っている．つまり，ZKIPは「絶対に知識が漏れない」のに対し，Σプロトコルは「検証者が正しくプロトコルに従っている限りは知識は漏れない」ことしか保障しない．

さて，「検証者が正しくプロトコルに従っている」ということは，「チャレンジcがランダムである」ということである．しかし実際に検証者がcをランダムに選ぶことは，だれも保障できない．では，cを検証者が選ぶのではなく，強制的に何らかの乱数が割り当てられるようにしたらどうだろうか．そうすれば，検証者がcに何か細工をすることで何らかの知識が漏れるという心配が不要になる．また，乱数割り当てを証明者が行えば，3ムーブが1ムーブ（つまり一方的に情報を送るのみ）にすることも可能である．

具体的には，以下のとおりである．

(1) まず証明者が何か計算し，結果Rを得る．
(2) 証明者はRをハッシュ関数へ入力し，ランダムビット列に見えるハッシュ値cを得る．
(3) 証明者はcを元に何らかの計算をし，その結果zを得る．検証者へ（R, c, z）を送る．
(4) 検証者は検証式が成り立ち，かつcが正しいハッシュ値である場合にacceptし，そうでなければrejectする．

Σプロトコルは条件付零知識性をもつので，「ランダムビットcがあらかじめわかっていれば，それに合ったRとzを計算することは容易」である（→コラムを参照）．それならば，知識を持たない証明者が検証者を騙すことも可能ではないか．例えば，証明者は先にcを適当に選び，検証式を満たす（R, z）を計算できる．しかしその場合，cはRのハッシュ値には一致しない．したがってacceptされるためには，3ムーブのΣプロトコルと同様に，特定の知識を持っていなければならない．

このように，ハッシュ関数を利用することによって，Σプロトコルの弱点である検証者の不正を防ぎ，かつ1ムーブで証明が終了するように変換が可能である[3]．この着想は，FS法とともにフィアットとシャミアに

[3] ただし，ハッシュ関数は十分安全なものを用いる必要がある．

よって提案されたため，FS変換と呼ばれる．

ディジタル署名への応用：元々FS変換は，FS法をディジタル署名へ変換するため提案された．ディジタル署名とは，ある文書に対して署名者が「この文書の内容を私は承知していますよ」という事実を証明することである．したがって，cがRだけでなく署名したいメッセージにも依存するように，c＝hash（R, メッセージ）とすれば，(R, c, z) をディジタル署名として利用できるのである．

シュノアのΣプロトコルに対してFS変換を施したものがシュノア署名である．ただし，シュノアは以下のように修正を加えることによって，より署名をコンパクトにしている．

* 検証式は（R, c, z）の3変数の関係を示す等式である．この式を利用すると，cとzからRを求めることができるため，署名は（c, z）のみとする．
* 特殊なgを選択することで，zのビット長を必要最低限に短くする．

これらの工夫により，署名長が320ビットのディジタル署名方式が実現できる（ちなみに，有名なRSA署名で同程度の安全性を確保しようとすると署名長は1024ビット程度必要）．

7. エルガマル暗号関連のプロトコル

シュノアのプロトコル（図2）を元に，2つの離散対数が同じであることを示すプロトコルが容易に構成できる．具体的には，gと$X=g^x \bmod p$に加えてhと$X'=h^x \bmod p$が与えられているとしよう．このとき，「(gを底としたときの) Xの離散対数であり，かつ (hを底としたときの) X′の離散対数であるようなxを知っている」ことを示す知識の証明（Σプロトコル）を作ることができる（図3を参照）．

このプロトコルは，エルガマル暗号と非常に相性が良い．エルガマル暗号では，平文mは公開鍵 (g, y, p) を用いて

$$\text{暗号文}=(G, M)=(g^k \bmod p,\ my^k \bmod p)$$

と暗号化される（kは暗号化のたびに選ばれる乱数）．少し変形すると，$(G, M/m)=(g^k \bmod p,\ y^k \bmod p)$である．そこで，図3のプロトコルを$(g, X=G, h=y, X'=M/m)$に対して適用すると，平文がある固定値mであることを証明することができる[4]．

さらに，エルガマル暗号を用いて，同じ平文を2回暗号化するとどうなるか考えてみよう．1回目の暗号

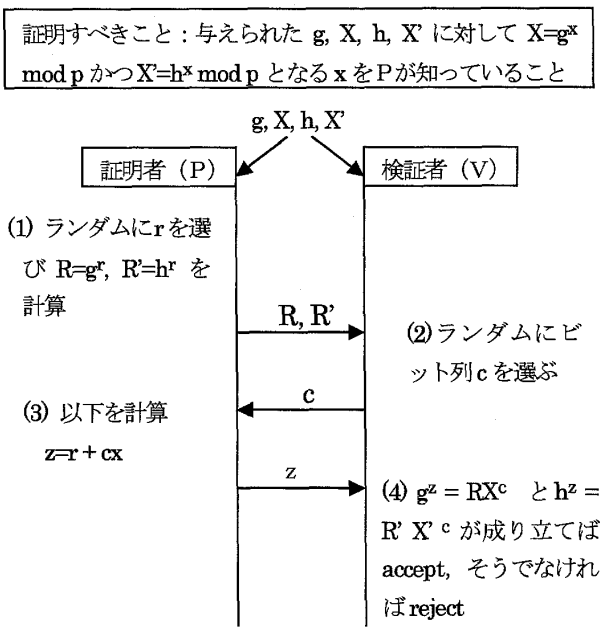

図3　離散対数が同じであることを示すプロトコル

文は乱数k1を用いて

$$(G1, M1)=(g^{k1} \bmod p,\ my^{k1} \bmod p)$$

となり，2回目の暗号文は別の乱数k2を用いて

$$(G2, M2)=(g^{k2} \bmod p,\ my^{k2} \bmod p)$$

となる．同じ平文であっても見た目の異なる暗号文が生成されるのである．図3のプロトコルは，2つの暗号文が〈見た目は違うが，平文は同じである〉ことを示すことにも利用することができる．具体的には，$(g, X=G1/G2, y, X'=M1/M2)$に対して図3のプロトコルを適用すれば良い．

匿名通信路への応用：〈見た目は違うが，平文は同じである〉ことを示すプロトコルは，ミックスネットタイプの匿名通信路に応用される．ミックスネットとは，多数のユーザからの暗号文を受け取りシャフルしてから復号することで，各通信内容が誰から来たものかを秘匿できる通信路である．ミックスサーバと呼ばれる機関が暗号文の束をシャフルするが，暗号文の追跡を不可能にするために，ミックスサーバにおいて各暗号文は〈平文は同じで見た目が違う〉暗号文へ変換される．その際に，ミックスサーバが平文をすり替えないとも限らない．そこで，〈見た目は違うが平文は同じ〉ことをミックスサーバが証明するわけである．実際は，複数の暗号文の順番も入れ替えるため，もっ

[4] 正確には，「既知の平文mを乱数kを使って暗号化すると (G, M) になるような乱数kを知っている」ことを証明することになるが，そのような乱数の存在自体が，「平文がmである」ことを示している．

と複雑なプロトコルが必要となるが，ここでは詳細は割愛する．

8. 部分的な知識を示すΣプロトコル

次に，Σプロトコルの組み合わせのテクニックを紹介する．

例えば「Aを知っている」ことを証明するΣプロトコルP_Aと，「Bを知っている」を証明するΣプロトコルP_Bがあったとしよう．もし，「AもBも知っている」ことを証明したい場合には，2つのプロトコルを順番に，あるいは同時に実行すればよい．では，「AまたはBを知っている」ことを証明したい場合は，どうすれば良いだろうか（実際にどちらを知っているかは秘密にしたままで）．この問題は非常に興味深いテクニックによって解決された[4]．

証明者がAを知っている場合，「Aを知っている」ことを証明するプロトコルP_Aは正常どおり実行でき，Bは知らないのでプロトコルP_Bは実行できない．しかし，チャレンジcの値が事前にわかっていれば，「Bも知っている」と騙すことはできるのである．そこで，「AまたはB」の知識の証明は以下のようになる．

(1) c_Bを適当に決め，それに合うようにR_B，z_Bを計算しておく．
(2) P_Aのプロトコルに従ってR_Aを計算する．検証者へR_AとR_Bを送る．
(3) 検証者はランダムビット列cを選び送る．
(4) 証明者は$c_A=c-c_B$とする．プロトコルP_Aでc_Aを受取ったとみなし，z_Aを計算する．検証者へ(z_A, z_B, c_A, c_B)を送る．
(5) 検証者は(R_A, c_A, z_A)，(R_B, c_B, z_B)がそれぞれP_AとP_Bの検証式を満たし，$c=c_A+c_B$であるならばacceptする．

「Bのみ知っている」という場合は，P_BとP_Aの役割を逆にすれば良い．

この組み合わせ手法を一般化すると，3つ以上の命題に対して「少なくとも1つは知っている」ことを示すΣプロトコルも作ることができ，「知識のORの証明」「ORのプロトコル」と呼ばれている．

信任投票への応用：ORのプロトコルを用いると，安全なオンライン信任投票を実現できる．例えば，「信任」を投票したければ「1」を，「不信任」の場合は「0」をエルガマル暗号で暗号化して，集計センターへ送る．エルガマル暗号は，「いくつかの暗号文の積は，平文の和の暗号文となる」という性質を持つため，すべての投票暗号文をすべて掛け合わせてから復号すれば，個々の投票内容は秘匿したまま，信任の数を得ることができるのである．しかし，もし投票者が「3の暗号文」を送ったとすると，その投票者は3人分の信任票としてカウントされてしまう．そこで，投票者は集計センターに対して，その暗号文は「1の暗号文または0の暗号文である」ことを証明すれば良いのである．

9. ハッシュ関数への応用

Σプロトコルをハッシュ関数へ応用する手法も最近提案された．ハッシュ関数とは，任意のビット列から一定の長さのビット列へ変換するための関数である．実際の計算では，可変長の入力を一定の長さのビット列に区切り，圧縮関数と呼ばれる小規模な関数を繰り返し用いることで，一定の長さのビット列へ変換する．Σプロトコルからは，この圧縮関数を作ることができる．安全性の原理については省略するが，例えば図2のプロトコルからは，$\mathrm{hash}(c, z)=X^c g^z$という関数が得られる．ただし，誰も離散対数xを知らないような固定値Xを公開パラメータとして用いる．

10. おわりに

本稿では，Σプロトコルと呼ばれる「特定の知識の所有を証明するためのプロトコル」について解説した．紙面の都合上，新しい成果についてはほとんど触れることができなかったが，暗号プロトコルにおける基本的な考え方や面白さを感じていただけたらと思う．

参考文献

[1] S. Goldwasser, S. Micali and R. Rivest, "The knowledge complexity of interactive proof-system," STOC '85, pp. 291-304 (1985).

[2] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," LNCS Vol. 263, Crypto '86, pp. 186-194 (1986).

[3] C. P. Schnorr, "Efficient signature generation by smart cards," J. of Cryptology, pp. 161-174 (1991).

[4] R. Cramer, I. Damgard and B. Schoenmakers, "Proofs of partial knowledge and simplied design of witness hiding protocols," LNCS Vol. 839, Crypto'94, pp. 174-187 (1994).