暗号プロトコルとその応用

黒澤　馨，尾形　わかは

1. はじめに

コンピュータネットワークの発達に伴って，今まで対面によってのみ行われてきたショッピングや株の売買などが（対面せずに）ネットワークを介してできるようになってきた。

しかし，対面してできる物事を，情報のやり取りだけでできるのだろうか。暗号プロトコルは，このような一見不可能なようなやり取りを，暗号などの情報セキュリティ技術を用いることで可能にしてしまうマジックプロトコルである。本稿では，電話で行うコイン投げやオンライン・カードゲーム・プロトコルのようなパズル的なものから，高度な認証を可能とするゼロ知識証明などについて，わかりやすく解説する。また，暗号プロトコルに必要なセキュリティツールについても簡単に説明する。

2. 暗号プロトコルでできること

暗号プロトコルとは，通信ではできそうにないこと，さらには，どうやっても（通信以外の手段を用いても）できそうにないことを可能にしてしまうマジックプロトコルである。では，マジックプロトコルによってできてしまう「できそうもないこと」とは一体どんなことだろうか？

まず，コイン投げやカードゲーム，無記名投票など。これらは，モノがあれば容易にできるが，モノを通信あるいは情報に置き換えたとたん，できそうになくなってしまう。このような「できそうにないこと」を暗号等をつかって可能にしたプロトコルは，「電子○○」「電話で○○」「オンライン○○」「ディジタル○○」という名前が付いていたりする。たとえば，すぐ後で説明する「電話でコイン投げ」や本特集の別の記事で解説されている「電子選挙」などである。また印鑑を情報化した「ディジタル署名」もそうである。

もう一つの「できそうにないこと」は，「秘密を他の人には知らせないでおきながら，秘密を知らないとできないようなことをすること」である。たとえば，証明は秘密にしたまま，定理が正しいことのみを証明したり，10人のもつ財産の額を秘密に保ったまま，その総額を求めたりする。これは，一見不可能なように見えるが，暗号プロトコルを使うとこんなことができてしまう。前者のように秘密に関する証明をするプロトコルはゼロ知識証明と呼ばれており，後者のように秘密の値から別の値を求めるようなプロトコルはマルチパーティプロトコルと呼ばれている。

実は暗号プロトコルは「絶対にウソをつかないし，絶対に秘密をもらさない」という信頼の置ける人（trusted party）がいる場合，話は簡単になる。なぜなら，モノが必要な場合にはtrusted partyがモノのかわりになることができ，ゼロ知識証明では，trusted partyに証明を見せ検証してもらえばよい。また，マルチパーティプロトコルでは，trusted partyに秘密を全て教えて必要な値を計算してもらえばよい。しかし，絶対にウソをつかなく秘密も守るという人の存在を仮定するのは現実の世界では無理があるし，たとえそのような人がいたとしても，その人が必ず参加しないと目的が果たせないのでは不便である。そこで，trusted partyがいなくてもうまくやっていけるように作られたのが暗号プロトコルである。

一言で暗号プロトコルと言ってもさまざまなものがある。次の章からは，いくつかの具体的な暗号プロトコルについて見ていく。

3. 電話でコイン投げ

暗号プロトコルの最初の例として，電話で行うコイン投げを挙げよう。コイン投げは，意見の対立する2人の間で，公平に物事を決めるのに役立つ。たとえば，

くろさわ　かおる　東京工業大学大学院理工学研究科集積システム専攻
〒152-8552 目黒区大岡山2-12-1
おがた　わかは　東京工業大学理財工学研究センター
〒152-8552 目黒区大岡山2-12-1

一方向性置換

一方向性置換とは以下の条件を満たす1：1の関数である．

1. 任意のxに対して，$y=f(x)$を求めることが容易である．
2. 任意のyに対して，$y=f(x)$となるxを見つけることは非常に難しい．

たとえば，代表的な公開鍵暗号であるRSA暗号の暗号化関数

$$f(x)=x^e \bmod n$$

は，一方向性置換になっている．

ハードコア属性

ある一方向性置換fに対するハードコア属性とは，以下の条件を満たす2値$\{0,1\}$をとる関数Bである．

1. 任意のxに対して，$y=B(x)$を求めることが容易である．
2. 任意のyに対して，$B(f^{-1}(y))$を予想できる確率はほとんど1/2である．

たとえば，$B(x)=x \bmod 2$は，左で示したRSA暗号化関数fのハードコア属性になっている．つまり暗号文を見ても平文の最下位ビットを当てることはできない．

Aliceは裏が出るか表が出るかを予想し，Bobがコインを振る．Aliceの予想通りならばAliceの勝ち，そうでなければBobの勝ちである．対面して行うコイン投げにはコインという物理的なモノが使われ，それによってランダムな勝敗をとてもシンプルに決めることができる．

では，電話ごしに同じことができるであろうか？Aliceは表か裏を予想し電話でBobに言う．Bobはコインを振って，結果を電話でAliceに知らせる．はたしてAliceはその結果を信じるだろうか？　2人の意見が対立していれば対立しているほど，Aliceは納得しないだろう．Aliceに「証拠を見せてよ」と言われても，Bobが見せられる証拠はない．しかし，便利なセキュリティツールを使うと，公平なコイン投げを電話ですることができるのである．

最初に世に出た電話ごしのコイン投げは，Blum数という特殊な数を使う方法[1]であるが，ここでは，一方向性置換とそのハードコア属性を使った方法について説明する．最も便利なセキュリティツールである**一方向性置換**と，一方向性置換に付随するセキュリティツールである**ハードコア属性**については簡単な定義と例を囲みの中で説明しておく．

まず，AliceとBobはどんな一方向性置換fとハードコア属性Bを使うかを決めておく．次にBobは適当にxを選んで$f(x)$の値を計算し，それを電話でAliceに教える．Aliceはそのハードコア属性$B(x)$が1(表)であるか0(裏)であるか予想して，Bobに伝える．BobはxをAliceにばらす．そして，Aliceの予想があたっていればAliceの勝ち，間違っていればBobの勝ちとする．

一方向性置換とハードコア属性の性質から，xから$f(x)$や$B(x)$を容易に計算できるためBobがウソをつくことはできないし，Aliceは$f(x)$から$B(x)$を当てることもできない．したがって，勝敗は公平に決められる．

4. カードゲーム

もう一つ，モノがあれば簡単なタイプの暗号プロトコルを紹介する．それはオンラインでカードゲームができるプロトコルである．なお，先に述べたとおり，絶対に信頼の置けるtrusted partyは存在しないとする．

実際のカードゲームは，裏にするとどのカードも同じように見え，全てのカードはそれぞれ一枚ずつしか存在しえないというモノの特徴を利用している．カードというモノを情報に置き換えたとたん，このような特徴はなくなってしまうので困ったことになる．

オンラインのカードゲームを行うための基本方針は，まずカードの表を0から52の数値で表し，カードの裏を表の数値の暗号文によって表すことである．そして，カードの裏を示す暗号文は，全員が見ることのできるホームページなどに常に示しておく．たとえばリアルワールドで，裏を向けた53枚のカードがデッキにある場合は，53個の暗号文がホームページに書かれる．このとき，暗号方法としてRSA暗号のように1対1の関数を使うと，53回のしらみつぶしによって裏から表の数値がまる見えになってしまう．そこで，1つの平文の暗号文が数多く存在する**多値確率的暗号**というセキュリティツールを用いる．多値確率的暗号については，囲みに説明を示す．

53枚のカードを使うゲームを行う場合，$l=53$の多

多値確率的暗号

l 値確率的暗号とは，第一引数の定義域が $\{0, 1, \cdots, l-1\}$ で，以下の条件を満たす2変数関数 $E(\cdot, \cdot)$ である。

1. 任意の平文：$m \in \{0, 1, \cdots, l-1\}$ と乱数 r に対して，暗号文：$y=E(m, r)$ を求めることが容易である。
2. 暗号文 y から $y=E(m, r)$ を満たす平文 m は一意に定まり，特定の秘密を知っていれば暗号文から平文を求めるのは容易である。
3. 特定の秘密を知らずに，任意の y に対して $y=E(m, r)$ となる $m \in \{0, 1, \cdots, l-1\}$ を推測できる確率はほとんど $1/l$ に近い。

たとえば，

$$E(m, r)=y^m r^2 \bmod n$$

は，2値確率的暗号になっている。ただし，y はどんな a を用いても $y=a^2 \bmod n$ と書けないような値である。

値確率的暗号系を使うと，カードの裏から表の数値が全くわからない。つまり，普通のカードのようにすべてのカードの裏は全く区別がつかないようにできる。

では，カードゲームをはじめよう。まず始めに53枚のカードを作り出さなければいけない。そのためには，カードの表の情報を0から52の値と仮定する。次に誰かがこれらを裏に向け（暗号化し）て結果（53個の暗号文）をホームページに載せる。次いで誰かが裏のままシャフルして，並べ替えられた暗号文をホームページに載せる。このときモノを使ったカードゲームと違い，シャフルしたプレーヤにはカードの並び替え方がわかってしまうので，全員が順にシャフルをする。

さて，裏を向けたままのシャフルは，暗号文の順番を入れ換えるだけでよいだろうか？　たとえば，シャフル前のカードの裏が

$$E(0, r_0), E(1, r_1), \cdots, E(52, r_{52})$$

で表されていて，シャフル後のカードの裏が

$$E(1, r_1), E(23, r_{23}), \cdots, E(0, r_0)$$

で表されていたとしたら，一番前にあるカードは明らかに1であり，その次は23であり，最後のカードは0である。つまり，多値確率的暗号系により裏から表の数値を区別不可能なようにはできたが，シャフルによってカードがどこへ移動したかという相対的な位置は誰にでも分かってしまう。そこで，シャフルしたカードの裏は

$$E(1, r_1'), E(23, r_{23}'), \cdots, E(0, r_0')$$

のように，各平文に対し暗号文を作りなおして表現する必要がある。

ここまでがカードを裏に向けてシャフルするという基本動作である。さらにゲームを進めていくためには，これ以外に「カードを配る」「カードを捨てる」「カードを見せる」等の基本動作が必要であり，これらの実現方法も提案されているが，ここでは詳しくは述べない[2]。

また，上で示した方法は「誰も不正（イカサマ）をしない」ということを前提にしている。この前提を取り払うためには，自分は不正（たとえば，シャフル中に全てのカードをスペードのエースに変えてしまうなど）を行っていないという証明を，要所要所で行う必要がある。このような証明は，証明専用のセキュリティツールであるゼロ知識証明を用いれば可能である。

ゼロ知識証明の3条件

1. 正しい命題は検証者によって必ず受理される。
2. 正しくない命題は検証者によって圧倒的確率で棄却され，正しくない命題が検証者に受理される確率は限りなく0に近づけることができる。
3. 命題が正しい場合に，検証者がどのようにふるまおうとも，「命題は正しい」という事実以外の知識は得られない。

5. ゼロ知識証明

この章では，暗号プロトコルの中でも特に重要なゼロ知識証明（Zero-knowledge Interactive Proof，以下ZKIPと書く）について，例を挙げて説明する。

5.1 定義と応用例

ZKIPとは，何かを証明したい人とそれを検証したい人で行う2者間のプロトコルである。証明をしたい人と検証したい人はそれぞれ証明者，検証者と呼ばれる。証明者は何か特殊な計算能力を持っていたり特別な秘密の値を知っており，それに対して検証者はごく普通の人（多項式時間チューリングマシン）である。証明したい事柄は，命題と呼ばれる。ZKIPの満たす

べき3条件[3]を囲みに示した．

ZKIPの第1，第2の条件の意味は明白であるが第3の条件はちょっと分かりづらいので，例を挙げて考えてみよう．

たとえばオンライン・カードゲームで証明したい命題とは，「自分のシャフルした後のカードの束は，シャフル前のカードの束の順番を入れ換えただけである」というものであった．手っ取り早くこれを証明するためには，シャフルを行ったプレーヤが用いた乱数等を全て公開すればよい．そうすれば他のプレーヤは正しく計算していることを確認することができる．しかし，このような計算の証拠を見せてしまってはシャフルした意味がないので，「証拠を見せずに」納得してもらう必要がある．これを実現するための条件が第3の条件である．

この例のように，入力がある一定の条件を満たしていることを証明するZKIPは「言語のZKIP」と呼ばれている．(「言語」とは，ある一定の条件を満たしている入力の集まりのことである) 言語のZKIPでは，証明者は無限大の能力を持っていると仮定される．

次に，もっと実用的な例を考えよう．銀行のATMを用いて預金を下ろすときには，暗証番号を打ち込むのが主流である．そして，暗証番号を知っているという事実によって本人認証を行う．このやり方は，ATMに盗聴装置を取りつけられることはなく，暗証番号を受け取って本人認証を行うソフトウェアを作るプログラマーは他人の暗証番号を盗める仕掛けをしないなど，いくつかの仮定の上で安全性が保証されている．しかし最近では，ATMコーナーに比べればセキュリティ度が低いと予想される小売り店において，暗証番号を打ち込むことによって預金口座からお金を引きだして買い物ができるシステムが広がりつつある．この場合，小売り店において暗証番号が盗聴されれば，その暗証番号を用いたなりすましが可能となってしまう．

そこで，ZKIPが活躍する．ZKIPを用いると，「暗証番号を知っている」という事実を暗証番号そのものの値が検証者 (ATM) に漏れることなく，証明できる．したがって，ATMに細工をしてもなりすましをすることはできない．

ここで使われるZKIPは，証明者が何かの情報を知っていることを証明するため，言語のZKIPに対して「知識のZKIP」と呼ばれている．知識のZKIPでは，証明者は無限大の計算能力は持たないが代わりに秘密情報を知っている．

さて，ZKIPがどんなに便利かについて長々と述べてきたが，実際にどのようにしたらそのような証明ができるのか，次節で具体例を示す．

5.2 Fiat-Shamir法

本節では，Fiat-Shamir法[4]と呼ばれるZKIPを紹介し，それが先に示した3条件を満たしていることを簡単に示す．このプロトコルは，平方剰余問題に対する言語のZKIPで，与えられた入力(a, n)に対して

$$a=x^2 \bmod n$$

となるxが存在することをxを示さずに証明する．具体的なプロトコルの手順を囲みに示した．

このプロトコルがZKIPの第1の条件を満たすことは明らかである．

第2の条件を満たすことを示すために，簡単にこのプロトコルの流れを見てみよう．まず証明者は適当なbを作り，検証者に送る．そして，$e=0$の時は$b=r^2 \bmod n$を満たすrが存在することを証拠rを送ることによって示す．一方，$e=1$の時は$ab=z^2 \bmod n$を満たすzが存在することを証拠zを送ることによって示す．したがって，いずれのeに対してもチェックが成り立つためにはrとzが存在して$a=(r/z)^2 \bmod n$が成り立たなければならない．しかし，命題が正しくない場合には$a=x^2 \bmod n$となるようなx

Fiat-Shamir法

共通入力は(a, n)であり，証明者は$a=x^2 \bmod n$となるxを知っている．証明者と検証者は以下のやり取りを$N=\log_2 n$回繰り返す．

1. 証明者は乱数rを選び，$b=r^2 \bmod n$を計算し，bを検証者に送る．
2. 検証者はランダムビットeを選び，これを証明者に送る．
3. 証明者は，受け取ったeが0のときには$z=r$を，eが1のときには$z=rx \bmod n$を検証者に送る．
4. 検証者は，eが0のときには$b=z^2 \bmod n$が成り立つことを，eが1のときには$ab=z^2 \bmod n$が成り立つことをチェックし，成り立たなければプロトコルを中断する．

N回のチェックが全て成り立ったならば，検証者は入力$(a. n)$を受理する．

シミュレータ M

M は入力 (a, n) に対して，まず乱数ビット列 R を選び，検証者に彼の乱数テープとして与える．$Seq=(R)$ とする．以下の作業を N 回繰り返す．

1. ビット e' を予想する．$e'=0$ のとき，乱数 r を選び，$b=r^2 \bmod n$ を計算する．$e'=1$ のとき，乱数 z を選び，$b=z^2/a \bmod n$ を計算する．
2. 検証者に b を送り，e を受け取る．
3. $e=e'$ のときは1.まで検証者をリセットし，M 自身も1.まで戻る．
4. $e=e'=0$ ならば $z=r$ とし，$e=e'=1$ ならば $z=rx \bmod n$ とする．
5. 検証者に z を送る（かならずチェックは通る）．Seq の最後に，(b, e, z) を加える．

N 回繰り返した後，Seq を出力する．

は存在しない．つまり，上記のような r と z のうち少なくとも一方は存在しない．よってランダムなビット e に対して証明者がチェックを通過できる確率は高々1/2である．繰り返し回数が N の時に，全てのチェックを通過する確率は高々 $(1/2)^N$ であり，これは N を増やすことによって指数関数的に減らすことができる．

最後に，第3の条件を満たすことを示す．正しい命題に対し，検証者が見ることのできる情報全てを $View$ で表すことにすると

$$View=(R, b_1, e_1, z_1, b_2, e_2, z_2, \cdots, b_l, e_l, z_l)$$

となる．ただし，R は検証者の乱数テープである．検証者がどのようにふるまおうとも $View$ には証明者しか知り得ないような情報が入っていない，ということを示したい．

ここで，検証者と対話をし，$View$ とそっくりな情報列を出力しようとする**シミュレータ M** を考える．M は x を知らない多項式チューリングマシンである．ただし，M は検証者をリセットすることができるとする．M の動作を囲みに示す．

このとき，M の出力と，検証者が証明者とFiat-Shamir法を行った際に見ることのできる $View$ は，確率変数として全く等しくなる．M は検証者と同じ計算能力しか持たないので，これは結局，検証者自身が $View$ を計算できることを意味する．したがって，検証者はFiat-Shamir法を行うことによって新しい知識を得ることはない，ということがいえる．

これで，Fiat-Shamir法はZKIPの3条件を満たしていることが示せた．

以上ではFiat-Shamir法を言語のZKIPとして扱ったが，Fiat-Shamir法は知識のZKIPでもある．言語のZKIPにおける証明者は無限大の能力を持っているのに対し，知識のZKIPにおける証明者は多項式時間の能力しか持たない．しかし，Fiat-Shamir法において多項式以上の能力を必要とするのは a の平方根である x を求めるときのみであるから，その x を知っていさえすれば多項式の能力しか持たない証明者でもプロトコルを行うことができる（第1の条件）．次に，証明者が多項式の計算能力しか持たず，また x を知らないならば，2通りのランダムビット e に対して正しく z を作ることができない．したがって，検証者が受理するならば証明者はほぼ確実に x を知っているはずである（第2の条件）．

6. マルチパーティプロトコル

最後に，マルチパーティプロトコルとよばれるプロトコルを紹介しよう．マルチパーティプロトコルでは，複数のプレーヤ $P_1, P_2, \cdots$ がそれぞれ秘密 $s_1, s_2, \cdots$ を持っており，秘密の値は漏らさないである関数値 $f(s_1, s_2, \cdots)$ を全員で計算しようというものである．ここでは，プレーヤの数を n とする．

「秘密に対する関数値を秘密をばらさないで計算する」という，矛盾するようなことを可能にするには，ZKIPと共に，秘密分散共有法というセキュリティツールを使うとよい．**秘密分散共有法**とは，秘密 s を全体で共有する方法であり，一定以上のプレーヤの集合は秘密を復元できる．どのプレーヤ集合が秘密を復元できるかによって多くの方式が存在するが，ここでは簡単のため，**満場一致法**を考える．具体的な方法については囲みに示した．ただし，秘密 s はある素数 q に対して集合 $Z_q=\{0, 1, \cdots, q-1\}$ に含まれているとする．

では，満場一致法を応用して

$$f(s_1, s_2, \cdots, s_n)=s_1+s_2+\cdots+s_n \bmod q$$

を求めるプロトコルを考えよう．

まず，各 P_i はディーラーとなり，自分の秘密 s_i から満場一致法の分散手順に従って $v_{i1}, v_{i2}, \cdots, v_{in}$ を作り，各プレーヤ P_j に v_{ij} を秘密に渡す．つまり

$$s_i=v_{i1}+v_{i2}+\cdots+v_{in} \bmod q$$

である．プレーヤ P_j は $v_{1j}, v_{2j}, \cdots, v_{nj}$ を得る．次に

簡単な秘密分散共有法

n 人のプレーヤ全てが集まったときのみに秘密が復元できる方法.

秘密の分散手順 ディーラーは集合 Z_q から $v_1, v_2, \cdots, v_{n-1}$ を独立に一様ランダムに選び,

$$v_n = s - v_1 - v_2 - \cdots - v_{n-1} \bmod q$$

とする. ディーラーは, 各プレーヤ P_i に v_i を秘密に渡す.

秘密の復元手順 各プレーヤは各 v_i を公開する. すると,

$$s = v_1 + v_2 + \cdots + v_n \bmod q$$

として s が求まる.

各 P_j は

$$v_j = v_{1j} + v_{2j} + \cdots + v_{nj} \bmod q$$

を計算し, これを公開する. 求めたい和: $s = s_1 + s_2 + \cdots + s_n$ は, 全ての v_j から

$$s = v_1 + v_2 + \cdots + v_n \bmod q$$

で求めることができることは, 簡単にわかるだろう.

ここで, マルチパーティプロトコルで守るべき情報について, 少しだけ議論する. マルチパーティプロトコルでは, 各プレーヤの持つ秘密が守られなければ安全とは言えないように感じる. しかし, そうでない場合もある. たとえば $n=2$ の場合, どんなプロトコルでも $f(s_1, s_2) = s_1 + s_2$ が得られるのと同時に, P_1 は s_2 を, P_2 は s_1 を知ることができてしまうのは当然であり, 秘密を隠し通すことは不可能である. そこで, マルチパーティプロトコルを行った後にプレーヤが知ることのできる全ての情報が, 理想的な方法で $f(s_1, s_2, \cdots)$ を求めたときに得られるならば, このプロトコルは安全であるという. ここで, 理想的な方法とは, trusted party を仮定して全ての計算を trusted party にしてもらうという方法である.

実際にマルチパーティプロトコルを応用する場合は, プレーヤの停止や不正の可能性を考える必要がある. たとえば, 満場一致法を用いた上記の例では, 途中でプレーヤが1人でも停止すると $f(s_1, s_2, \cdots)$ は得られない. これを解決するには, 満場一致法の代わりに, n 人中 k 人のプレーヤによって秘密が復元できる (k, n) しきい値法[5], [6]と呼ばれる秘密分散共有法を用いればよい. そのほか, プレーヤの不正を検出するためには, 別のセキュリティツールや ZKIP を用いたりする必要がある.

7. おわりに

本稿では, 数ある暗号プロトコルのごく一部を紹介した. 不可能なように見えることが暗号プロトコルによって可能になることが理解していただければ幸いである. 興味のある方は, 参考文献[7], [8]を参照されたい.

参考文献

[1] M. Blum: "Coin flipping by telephone", Proc. of IEEE, COMPCON, pp. 133-137, (1982).

[2] K. Kurosawa, Y. Katayama and W. Ogata: "Reshufflable and Laziness Tolerant Mental Card Game Protocol", IEICE Transaction, E 80-A, pp. 72-78, (1997).

[3] S. Goldwasser, S. Micali and C. Rackoff: "The knowledge complexity of interactive proof systems", SIAM Journal on Computing, Vol. 18, pp. 186-208, (1989).

[4] A. Fiat and A. Shamir: "How to prove youself: Practical Solutions to identification and signature problems", LNCS 263, Advanced in Cryptology-Proc. CRYPTO '86, pp. 186-194, (1986).

[5] A. Shamir: "How to Share a Secret", Communications of the ACM, Vol. 22, No. 11, pp. 612-613, (1979).

[6] G. R. Blakely: "Safeguarding cryptographic keys", Proc. of the AFIPS 1979 National Computer Conference, vol. 48, pp. 313-317, (1979).

[7] 太田和夫, 黒沢馨, 渡辺治: "情報セキュリティの科学—マジック・プロトコルへの招待—", 講談社, ブルーバックス, (1995).

[8] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone: "HANDBOOK of APPLIED CRYPTOGRAPHY", CRC Press, (1997).