中央大学数学教室講究録　No.2

暗号理論における安全性概念

岡本 龍明
原稿作成：森山　大輔

中央大学理工学部数学教室
2008 年 3 月

中央大学数学教室講究録　2

暗号理論における安全性概念

岡本 龍明
(NTT 研究所)
原稿作成：森山　大輔
(中央大学大学院理工学研究科博士課程前期課程情報工学専攻)

中央大学理工学部数学教室
2008 年 3 月

まえがき

本書は，2008 年 1 月 28 日から 2 月 1 日にかけて中央大学大学院 理工学研究科 数学専攻 情報数学特別講義第四 において行われた講義の講義ノートである．

この講義は，代数学の応用の一例として情報セキュリティの基盤技術の 1 つである公開鍵暗号に焦点を当てたものであり，NTT 研究所に所属しておられる岡本龍明 氏を講師としてお招きし，公開鍵暗号とその安全性についての様々な解説を行っていただいた．講義は 5 日間に渡って行われ，公開鍵暗号の安全性やその証明，暗号において基礎として用いられている擬似乱数やゼロ知識，さらに具体的な鍵共有プロトコルとそれに対しての攻撃など，多岐に渡る暗号の分野に関しての説明が行なわれた．

岡本龍明氏は忙しい中を、日々発展している暗号の安全性に関する生々しい概念を丁寧に講義をして下さり、その内容をこの様な形で記録し学生・研究者への便宜を図って下さったことに、本講座の取り纏め役として心より感謝を申し上げる次第である。

また、本原稿は中央大学大学院理工学研究科博士課程前期課程情報工学専攻　森山大輔氏により取り纏められたものである。修士論文作成時期にも拘わらず、多大の労を割いてこの原稿を作成して戴いたことに、ここに深く謝意を表したい。

2008 年 3 月 7 日

中央大学数学教室　關口　力

目 次

1　イントロダクション

暗号は基本機能として

基本機能
○ 秘匿（暗号、鍵配送）
　○ 認証（署名）

を持ち、こうした基本機能の上に応用機能

複合（応用）機能
○ 電子投票
○ 電子決済
○ 電子契約
○ 電子ゲーム
○ マルチパーティプロトコル

を持っており，これらをまとめて暗号プロトコルと呼んでいる．これらに共通することは,「自身の秘密はそのまま隠しておいた上で，何かを行ないたい」という動機に基づいたものであり，近年ではそれらの証明の体系について議論が活発である．

2　暗号の安全性の定式化

暗号の安全性は主に証明によって行なわれているが，その手法として近年で用いたれているものは大きく 2 つに分けることができる．

- 攻撃ベース定式化 - 攻撃者とチャレンジャー間のゲームとして定式化するもので，攻撃者の能力に基づいて証明を追っていくもの．近年ではゲームを除除に変換して作るゲーム列を解析することで，安全性を証明する手法が発展しつつある．—公開鍵，デジタル署名 …
- シミュレーションベース定式化 - 現実のモデルと理想的なモデルとのギャップがほとんどないことを示すことで証明を行なうもの．暗号における全ての対象に対する統一的定式化を可能とする．—汎用的結合可能性 (Universal Composability).

また，現在のほとんどの暗号理論においては計算量的な安全性に基づいて証明が行なわれており，何らかの計算量的な仮定が安全であるならばその暗号プロトコルが安全である，ということを示すことがほとんどである．さらに言うと，その対偶を取ってその暗号プロトコルが敗れるとするならば計算量的な仮定が破ることができる，という理論から安全性を示すことが多い．例えば，公開鍵暗号の安全性に関する攻撃ベー

ス定式化の典型例が RSA 暗号である．

公開鍵暗号の具体例: RSA **暗号**

RSA 暗号は公開鍵 $n = pq$ (p, q : 素数) となるような値が与えられたとき，p, q を求めることが困難であるという素因数分解の安全性に基づいている．

3 公開鍵暗号

3.1 公開鍵暗号の定義

公開鍵暗号は 3 つのアルゴリズム (G, E, D) からなっており，

G : 鍵生成 - セキュリティパラメータ 1^n（n ビット列を意味する）を入力とし，公開鍵と秘密鍵のペア (e, d) を出力するアルゴリズム．

$$1^n \rightarrow \boxed{G} \rightarrow (e, d)$$

E : 暗号化 - 平文 M と公開鍵 e を入力とし，暗号文 C を出力するようなアルゴリズム．

$$(M, e) \rightarrow \boxed{E} \rightarrow C$$

D : 復号 - 暗号文 C と秘密鍵 d を入力とし，平文 M を出力するアルゴリズム．

$$(C, d) \rightarrow \boxed{D} \rightarrow M$$

である．n というのはセキュリティパラメータと呼ばれ，公開鍵や秘密鍵のサイズを決定するものである．例えば $n = 1024$ であれば，1024 bit の公開鍵が生成されるということである．このとき，鍵生成アルゴリズム G は確率的なアルゴリズムであり，同じセキュリティパラメータが入力されてもそのアルゴリズム内の乱数が作用することで毎回異なるような値を出力するようなアルゴリズムである．また，復号に関してはほとんどの場合において確定的なアルゴリズムである．確定的アルゴリズムとは，同じ暗号文と秘密鍵が入力されたときは毎回同じような出力を行なうようなアルゴリズムである．

3.2 公開鍵暗号の安全性

3.2.1 安全性の達成度

公開鍵暗号における安全性の達成度は以下のように分けることができる．

- 秘匿性: 暗号文の部分情報について得られるかどうかを捉えたもの．暗号では秘匿性に関して 2 つに分けて考える．

  一方向性 (OW:One Wayness) - 暗号文 C から平文 M が得られないこと．

  強秘匿性 (IND:Indistinguishability) - 暗号文 C から平文のいかなる部分情報も得られないこと．

  - 頑強性 (NM:Non Malleability) - ある平文 M に対する暗号文 $C = E(m)$ が与えられたとき，関係 R に関して $R(M, M')$ となるような暗号文 $C' = E(M')$ を出力することができないこと．すなわち，C, C' の情報から平文 M, M' の関係 R を決められない．

  .

頑強性に関しての具体例:電子入札

例えば，A 社と B 社が電子入札を利用して 1 つのある物を競り落とす場合と考える．そして，その入札金額は暗号化の処理が施されているとする．A 社は 1000 万で入札しようと思い平文 $m = 1000$ 万 から暗号文 $c = E(m)$ として入札したとする．このとき，B 社は A 社の暗号文 c を傍受しても，その暗号が強秘匿性を満たしていれば 1000 万円で入札したという情報はまったく分からない．しかし，もし頑強性を満たしていなかった場合，暗号文 c から $m' = 1001$ 万 となるような暗号文 $c' = E(m')$ を作ることで，A 社の入札金額に 1 万足した値の暗号文を作ることができる．このような m と m' に関係性があるような暗号文を作ることができないというのが頑強性である．

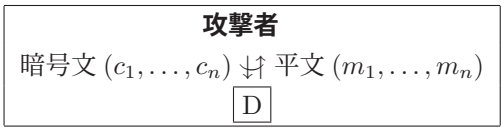
3.2.2 攻撃法

公開鍵暗号における攻撃者の能力は以下のように分けることができる．

- 受動的…選択平文攻撃 (CPA:Chosen Plaintext Attack)
  暗号文 C を受け取ったとき，攻撃者はいくつかの平文に対する暗号文を得た後に，暗号文 C に対する平文 M を出力するような攻撃である．公開鍵暗号では相手の公開鍵を利用することで誰もが暗号文を作ることができるので，これは受動的な攻撃である．

- 能動的…選択暗号文攻撃 (CCA:Chosen Ciphertext Attack)

  暗号文 C を受け取ったとき，攻撃者は受動的な動作のほかに，暗号文を送ることで平文を返してくる復号オラクルを利用することができる．また，この復号オラクルには何度もアクセスできる．ただし C を受け取った後に明示的に C に対して復号オラクルを用いることはないとする (つまり $C \neq \{c_1, \ldots, c_n\}$).



  選択暗号文攻撃では，暗号文 C を受け取る前においてのみ CCA を行なうことができるものを CCA1 と呼び，暗号文 C を受け取る前と後との両方で CCA を行なうことができる CCA2 という 2 つの攻撃方法がある．

注意：CCA の起こりえる場合として，例えば SSL 等においては，システムは文書が意味をなさない場合などに，コメントを返すプロトコルになっており，これからある程度の復号化情報が得られる．

3.2.3 公開鍵暗号の安全性の関係と歴史

また，上記で述べた 3 つ安全性の達成度と 3 つの攻撃法に関して，現在の研究では以下のような関係にあることが示されている．

	OW	IND	NM
CPA	OW-CPA	IND-CPA	NM-CPA
CCA1	OW-CCA1	IND-CCA1	NM-CCA1
CCA2	OW-CCA2	IND-CCA2	NM-CCA2

表 1: 公開鍵暗号の安全性の関係

この表の中にある 9 つの安全性のうち，OW-CPA が一番弱い安全性であり，NM-CCA2 が一番強い安全性である．基本的には，OW<IND<NM の順に安全性が強くなり，CPA<CCA1<CCA2 の順に安全性が強くなる．ただし，IND-CCA2 と NM-CCA2 に関しては等価であることが示されており，IND-CCA2 で安全であるような公開鍵暗号が一番安全性が高いということができる．また，公開鍵暗号および公開鍵暗号の安全性に関する定式化の歴史は以下の通りである．

1976 年:Diffie,Hellman (Diffie-Hellman 鍵配送) [11]

1978 年:Rivest,Shamir,Adleman (RSA 暗号) [27]

1979 年:Rabin (OW-CPA) [25]

1982 年:Goldwasser,Micali (IND-CPA) [15]

1990 年:Naor,Yung (IND-CCA1) [24]

1990 年:Rackoff,Simons (IND-CCA2) [26]

1992 年:Dolev,Dwork,Naor (NM-CCA) [10]

1998 年:Bellare,Desai,Pointcheval,Rogaway (IND-CCA2=NM-CCA2 の等価性の証明) [1]

3.3 安全性の厳密な定義

3.3.1 強秘匿性の定義１ (Semantic Security の定義)

いかなる攻撃者 $\mathcal{A}$ に対しても効率的なアルゴリズム $\mathcal{B}$ が存在し，どのような関数 f, h と全ての確率変数族 $\{X_n\}_{n\in\mathbb{N}}$ に対して次が成立することを強秘匿 (Semantic Security) という．

$$\Pr[\mathcal{A}(pk, E_{pk}(X_n), h(X_n)) = f(X_n)] < \Pr[\mathcal{B}(pk, h(X_n)) = f(X_n)] + \epsilon(n)$$

ここで，pk は公開鍵, $E_{pk}(X_n)$ は X_n を公開鍵 pk で暗号化したもの, $h(X_n), f(X_n)$ は X_n の部分情報, n はセキュリティパラメータ, ϵ は無視できる (negligible) 関数 $(^{\forall}c, ^{\exists}N, ^{\forall}n > N, \epsilon(n) < \frac{1}{n^c})$ をそれぞれ意味するものとする．$\mathcal{B}$ の計算量は $\mathcal{A}$ の計算量で定まる．つまり，この式はある平文に対しての暗号文が得られている状態から部分情報を推測する確率と，暗号文なしの状態から部分情報を推測する確率がほとんどないということを言いたいわけである．また，関数 f, h において f を最下位 1bit を出力する関数, h を最下位 1bit 以外のすべての bit を出力する関数とした場合，攻撃者は暗号文から 1bit の情報も得られていない，ということを意味する．

3.3.2 強秘匿性の定義２ (Indistinguishability の定義)

ある攻撃者に対して公開鍵を入力して 2 つの平文 m_0, m_1 を出力させる．そして，2 つの平文のうちどちらか一方を uniform に選び暗号化してその暗号文を攻撃者に返す．そして，攻撃者がそのどちらが暗号化されたかを推測し，その推測が当たっているのならば攻撃者の勝ちとするものが Indistinguishability の定義である．

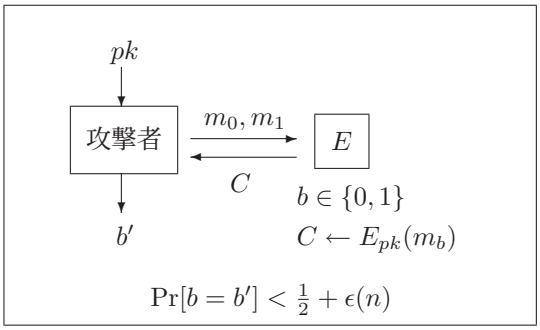


図 1: Indistinguishability の定義

3.3.3 強秘匿性の定義の比較

-定義 1 (semantically secure)

如何なる部分情報も漏らさないという意味をそのまま反映した定義，すなわち，暗号文と $h(X_n)$ から得られる如何なる平文に関する情報 $f(X_n)$ は，暗号文を用いない $h(X_n)$ から求められる．つまり，暗号文は平文の化なる部分情報 $f(X_n)$ を求めることに貢献しない．

-定義 2 (indistinguishability)

本来の意味は反映していないが，証明には利用しやすい．技術的な意味で有意義な定義である．

-等価性

定義 1 (Semantic Security) と定義 2 (Indistinguishability) は等価であることが示されている．

なお，公開鍵と平文から暗号文が一意に定まるような確定的暗号の場合，Indistinguishability は満たすことができない．なぜならば，確定的に暗号文が定まるのであれば 2 つの平文を暗号化してやればどちらの暗号文が来たのかがそのまま分かるからである．よって，Indistinguishability を満たすためには，暗号処理が確率的でなければならない．

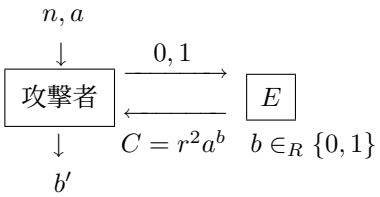
強秘匿暗号の例：GM(Goldwasser-Micali) **暗号** [15]

公開鍵: $n = pq, a \in \mathbb{Z}_n^*$，秘密鍵: p, q は素数とする．また，$a \in QNR_n$: 平方非剰余 であるとする．平方非剰余とは，a に対して $b^2 \equiv a \pmod{n}$ となるような b が存在しないということである．また，Jacobi 記号の値 $\left(\frac{a}{n}\right) = 1$ を仮定する．

暗号化は 1bit の平文 $m \in \{0,1\}$ に対して行う．$\mathbb{Z}_n$ の中から uniform にランダムな値 r を選び ($r \xleftarrow{U} \mathbb{Z}_n$ と書く)，$c = r^2 \cdot a^m$ を暗号文とする．このとき，$m = 0$ ならば $c = r^2$ であり，$m = 1$ ならば $c = r^2 \cdot a$ となる．

復号では，受け取った暗号文 c から秘密鍵 p, q を用いて $c_p := c \bmod p$ や $c_q := c \bmod q$ を計算する．そこから，$\left(\frac{c_p}{p}\right) = 1$ ならば $m = 0$，$\left(\frac{c_p}{p}\right) = -1$ ならば $m = 1$ として復号する．

GM 暗号は，素因数分解が困難なときに $\mathbb{Z}_n^*$ 上の平方剰余判定が困難である，という**平方剰余仮定**の下で IND-CPA を満たすことを示すことができる．



先ほどの Indistinguishability の図を今回の GM 暗号に適応させたものが上図である．攻撃者による bit の推測はそのまま平文に対しての推測と同じであるため，もし攻撃者が正しく bit を推測することができるならばそこから平方剰余を求めることができるので，それは平方剰余仮定に矛盾する．よって，GM 暗号は平方剰余仮定の下で IND-CPA であることが示すことができたことになる．

なお，GM 暗号において受け取った暗号文 $c = r^2 \cdot a^b$ に対して，$c \cdot a$ とすると反転させた bit に対しての暗号文を作ることができるため，NM は満たされない．
注意：公開鍵暗号で確定的暗号（何らかの乱数を導入していないもの）では，強秘匿性の定義 2 を満たすことはない．何故ならば，敵が公開鍵で計算できる彼である．従って，強秘匿であるためには，確率的でなければならない．

3.3.4 頑強性の定義１ (Non Malleability の定義１)

Non Malleability(NM) の定義は 2 通りあり，1 つは Dolev,Dwork,Naor が 1992 年に提案したものである [?].

これは，暗号文 c_1 から関係 R に関して $R(D(c_1), D(c_2))$ を満たすような暗号文 c_2 を出力するということを形式的に述べたもので，以下のような式によって表される．

$$\Pr[\mathcal{A}(c_1) \to (R, c_2) \mid R(D(c_1), D(c_2))] - \Pr[\mathcal{A}(c_1) \to (R, c_2) \mid R(m, D(c_2))] < \frac{1}{k^c}$$

このとき，m は平文空間からランダムに選ばれているものとする．これはつまり，暗号文 c_1 に対する平文と一定の関係を持つ平文の暗号文 c_2 を出力する確率は，任意の平文と c_2 に対する平文との関係をもつ暗号文 c_2 を出力する確率と変わらないと言うことを意味している．

3.3.5 頑強性の定義２ (Non Malleability の定義２)

もう 1 つの NM の定義は Bellare,Sahai によって 1999 年に提案されたもので [6], Indistinguishability の定義のようにゲームを考えたものによる定義であり，図 2 のように表される．ここで E は暗号オラクルであり， D は復号オラクルである．

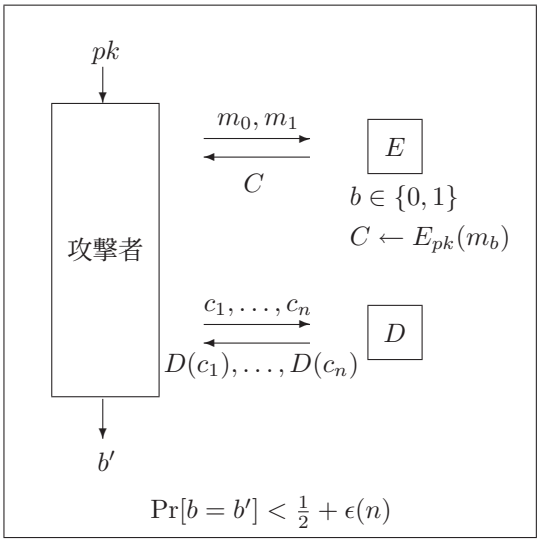


図 2: Non Malleability の定義

この NM の定義の中で用いられている復号オラクルには 1 度だけしかアクセスできないという制約が課せられている．その代わり，一度に暗号文は複数個送ることができ ($c_1, \ldots, c_n$ の n 個)，それに対する復号結果を受け取ることができる（１ラウンドのみに限定したパラレル攻撃）．よって，CCA の時に用いているような何度も利用できる復号オラクルとは利用方法が異なる点に注意が必要である．

3.3.6 頑強性の定義の比較

-定義１ (BDPR'98)
　頑強性の意味を反映させた定義

-定義２ (BS'99)
　頑強性の意味をゲーム化.

-等価性
　Bellare,Sahai は定義１と定義２が等価であることを示した.

この定義において，IND と NM の差は暗号文を受け取った後で (1 回限りの) 復号オラクルを利用できるか否かである．このとき，CCA2 において「暗号文を受け取った後で復号オラクルに何度もアクセスできる」という状況の中では IND と NM は等価であるので，IND-CCA2 と NM-CCA2 とは等価であることが Bellare,Desai,Pointcheval,Rogaway によって示された．つまり，公開鍵暗号において，望ましい安全性は「適応的選択暗号文攻撃に対して強秘匿」(IND-CCA2) である．

4 ディジタル署名

4.1 ディジタル署名の定義

ディジタル署名は以下のような 3 つのアルゴリズム (G, S, V) で表される．

G：鍵生成 - セキュリティパラメータ 1^n を入力とし，検証鍵と署名のペア (pk, sk) を出力するアルゴリズム．

$$1^n \rightarrow \boxed{G} \rightarrow (pk, sk)$$

S：署名 - 平文 m と署名鍵 sk を入力とし，署名 σ を出力するようなアルゴリズム．

$$(m, sk) \rightarrow \boxed{S} \rightarrow \sigma$$

V：検証 - 署名 σ と検証鍵 pk を入力とし，1 or 0（合格/不合格）を出力するアルゴリズム．

$$(\sigma, pk) \rightarrow \boxed{V} \rightarrow 1 \text{ or } 0$$

4.2 ディジタル署名の安全性

ディジタル署名の安全性は「偽造の困難性」と「攻撃法」の 2 通りから考える．その中でも偽造の困難性に関しては，

偽造困難性：

- 一般的偽造困難性 - ある文書に対して偽造が困難であるもの．
- 選択的偽造困難性

- 存在的偽造困難性 (EUF:Existentially Unforgeable) - いかなる文書ー署名対に対して偽造が困難であるもの.（すなわち，文書に意味が有ろうが無かろうが，署名を作ることが出来ない）

に分けられ，攻撃法に関しては

攻撃法：

- 受動的文書攻撃
- 一般選択文書攻撃
- 適応的選択文書攻撃 (CMA:Chosen Message Attack) - 署名者による署名クエリを利用できるもの．すなわち，何回も一般選択文書攻撃を繰り返すことが出来る．

の 3 つずつにそれぞれ分かれている．一番強力なのは EUF-CMA であり，通常ディジタル署名では EUF-CMA が求められる [16].

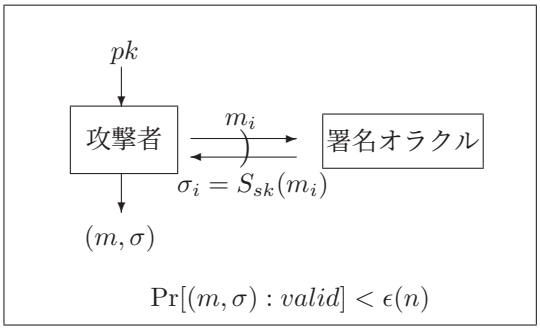


図 3: EUF-CMA の定義

5 公開鍵暗号の代表例

5.1 RSA-OAEP

まずは，公開鍵暗号の代表例として RSA-OAEP を紹介する．RSA-OAEP は 1994 年に Bellare,Rogaway によって提案され，random oracle model および RSA 仮定の下で NM-CCA2 であることが証明されている公開鍵暗号である [5] [13]．以下が RSA-OAEP の暗号化および復号を示したものである．

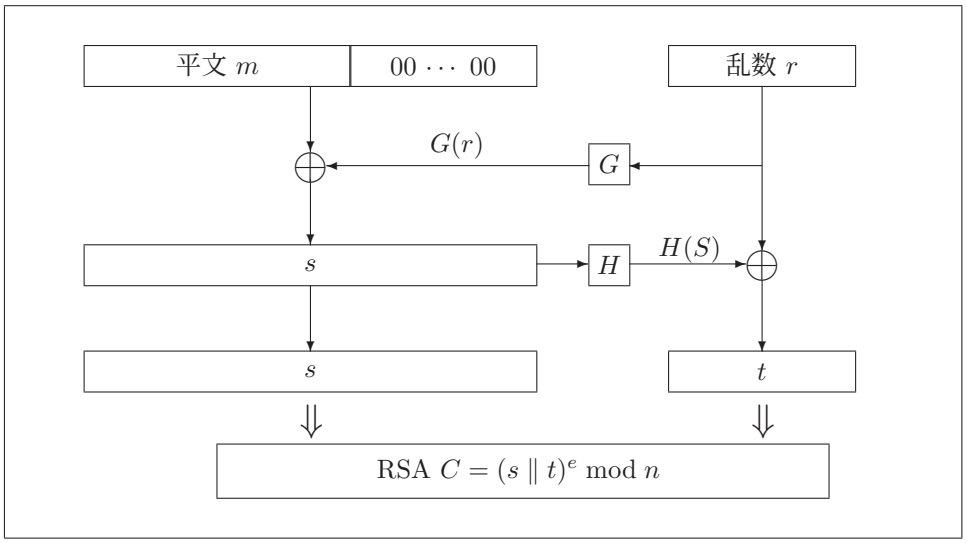


図 4: RSA-OAEP

RSA-OAEP では，平文 m の他に 0 の k ビット列を用意し，平文の後ろに付け加えたものを用意する．そして，暗号化する際には，

1. k ビットの乱数 r を選び，ハッシュ関数 G を用いて $G(r)$ を生成する ($G(r)$ の長さは平文に 0 のビット列を付け加えたものと同じ長さにする)．
2. 平文に 0 のビット列を付け加えたものと，$G(r)$ との排他的論理和を取る．その値を s とする．
3. ハッシュ関数 H を用いて s から $H(s)$ を求める．
4. 乱数 r と $H(s)$ との排他的論理和を取る．その値を t とする．
5. s の後ろに t を付け加えたものに対して，RSA 暗号を行ない暗号文 C を出力する (DES 等と同じく，このように 2 つの入力を混ぜ合わせる手法を feistel という)．

という手順を踏まえる．このとき，復号する際には，

1. RSA 暗号における復号処理を行い，s と t を得る．
2. ハッシュ関数 H を用いて s から $H(s)$ を求める．
3. $H(s)$ と t との排他的論理和を取ることで r を得る．
4. ハッシュ関数 G を用いて r から $G(r)$ を求める．

5. $G(r)$ と s との排他的論理和を取る．そして，その値のビット列の後ろに 0 のビット列（parity check) がついていれば平文 m を出力し，そうでなければ正しくない暗号文であるとして reject する．

という手順を踏まえる．

注意：

-RSA-OAEP では，G, H に暗号学的ハッシュ関数と呼ばれる SHA-1,SHA-2,SHA-3 などを用いることで安全であることが保たれるとしているが，安全性の証明の中ではハッシュ関数を理想的ランダム関数とみなしている．RSA-OAEP は random oracle model という理想的なランダム関数であることと RSA-仮定に基づき IND-CCA2 であることが示されている [13].

-Random oracle model におけるハッシュ関数では，ビット列に関してのテーブルを設け，そのときの出力をランダムにコイン投げなどによって決める．通常の暗号学的ハッシュ関数の場合は，入力値に対して何らかの計算によって出力が決まるが，random oracle model ではランダムな値が出力されるのである．

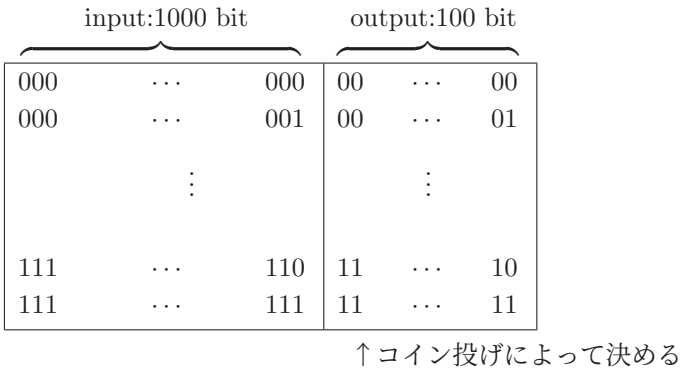


図 5: random oracle model

5.2 Diffie-Hellman 鍵配送

Deffie-Hellman は 1976 年に現代暗号の基礎である

- 一方向性関数
- 公開鍵暗号の概念
- DH 鍵配送（整数論が初めて暗号に使われた）
- ディジタル署名の概念

の諸概念を提示しているが [11]，不思議なことに DH 鍵配送を公開鍵暗号に書き直すことは，ElGamal まで出来なかった．

Diffie-Hellman 鍵配送は，ある 2 人のパーティ A と B が値をやり取りすることでお互いのみが求めることが出来るような鍵を求める鍵配送であり，以下のようなものである．

1. G を素位数 p の有限可換群とし，G の要素を $g \in G$ とする．
2. A は $a \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び，g^a を B に送る．
3. B は $b \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び，g^b を A に送る．
4. A は $SK_A = (g^b)^a$ を求める．
5. B は $SK_B = (g^a)^b$ を求める．

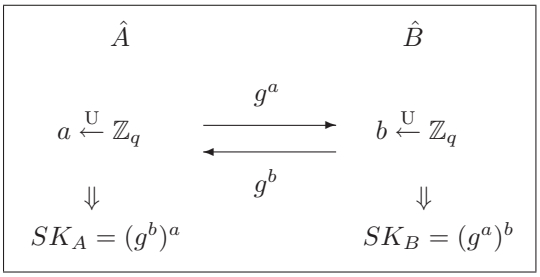


図 6: DH 鍵交換プロトコル

この Diffie-Hellman 鍵交換は，(g, g^a, g^b) が与えられたとき，g^{ab} を求めることは難しいという CDH (Computational Diffie-Hellman) 問題（または単に DH 問題とも言う）や，(g, g^a, g^b, g^{ab}) あるいは (g, g^a, g^b, g^c) $(c \neq ab)$ のどちらか一方を受け取ったとき，どちらであるのかの区別をすることが困難である DDH (Decisional Diffie-Hellman) 問題を安全性の根拠としている．DH 問題が難しいことを DH 仮定という．

5.3 ElGamal 暗号

ElGamal 暗号は，Diffie-Hellman 鍵配送を利用した公開鍵暗号で，以下のようなアルゴリズムである [12].

Setup: G; 素数位数 p の群，g: G の生成元とし，公開鍵と秘密鍵のペア $(A = g^a, a)$ を生成する．(G, g, A) を公開する．

Enc: 平文 $m \in G$ を送りたい場合，$b \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び，$(C_1, C_2) = (g^b, A^b \cdot m)$ を暗号文として出力する．

Dec: 秘密鍵 a を用いて，$C_2/C_1^a = m$ を出力する．

ElGamal 暗号は，DDH 仮定の下で IND-CPA であることを示すことが出来る．しかし，頑強性は満たされない．

5.4 公開鍵暗号の分類

大体の公開鍵暗号は，素因数分解 (IF:Integer Factorization) に基づくものと，離散対数 (DL:Discrete Logalithm) に基づくものに分けることができ，おのおのの代表的な安全性の根拠を表に示す．

	IF	DL
基本仮定	IF 仮定	DL 仮定
標準的仮定	RSA 仮定 QR 仮定	CDH 仮定 DDH 仮定
基本暗号	RSA 暗号	ElGamal 暗号
random oracle で IND-CCA2	RSA-OAEP	DHIES PSEC

表 1: 公開鍵暗号のおおまかな分類

5.5 Cramer-Shoup 暗号

公開鍵暗号における実際の安全性証明の方法を，Cramer-Shoup 暗号を用いながら説明する．Cramer-Shoup 暗号は 1998 年に作られたもので，random oracle を用いずに IND-CCA2 であることが証明された初めての公開鍵暗号である [9]．Cramer-Shoup 暗号は以下のようなアルゴリズムである．

Setup: 秘密鍵を $(x_1, x_2, y_1, y_2, z_1, z_2) \in \mathbb{Z}_p^6$ とし，素数位数 p の群 $G, \hat{G}$ 上の生成元 $g \in G, \hat{g} \in \hat{G}$ から

$$e = g^{x_1}\hat{g}^{x_2},\ f = g^{y_1}\hat{g}^{y_2},\ h = g^{z_1}\hat{g}^{z_2}$$

を求める．また，ハッシュ関数のための鍵 hk を選び，$(g, \hat{g}, hk, e, f, h)$ を公開する．

Enc: 平文 $m \in G$ を送りたい場合，以下の計算を行う．

E1: $u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選ぶ．

E2: $a \leftarrow g^u$ を求める．

E3: $\hat{a} \leftarrow \hat{g}^u$ を求める．

E4: $c \leftarrow h^u \cdot m$ とする．

E5: ハッシュ関数 HF_{hk} を用いて $v \leftarrow \mathsf{HF}_{hk}(a, \hat{a}, c)$ を求める．

E6: $d \leftarrow e^u f^{uv}$ を求める．

そして，$(a, \hat{a}, c, d)$ を暗号文として出力する．

Dec: 秘密鍵 $(x_1, x_2, y_1, y_2, z_1, z_2)$ を用いて，以下の計算を行う．

D1: フォームが正しいかどうかの確認を行う．

D2: $a, \hat{a}, c \in G$ であるかを確認する．

D3: ハッシュ関数 HF_{hk} を用いて $v \leftarrow \mathsf{HF}_{hk}(a, \hat{a}, c)$ を求める．

D4: $d = a^{x_1+y_1 v} \hat{a}^{x_2+y_2 v}$ であるかを確認する．

D5: $m \leftarrow c \cdot (a^{z_1} \hat{a}^{z_2})^{-1}$ を求める．

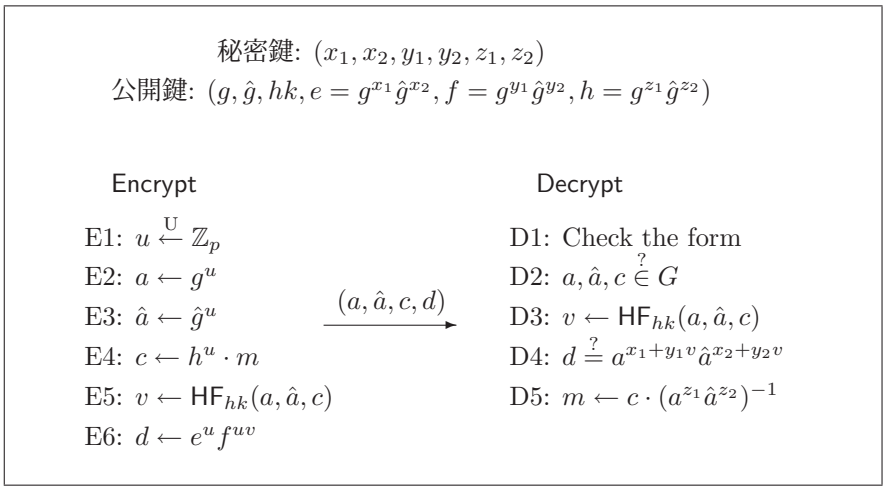


図 7: Cramer-Shoup 暗号

Cramer-Shoup 暗号では，攻撃者が無限の能力を持っている，つまり離散対数問題を解くことが出来る能力を持っていたとしても，攻撃者は公開鍵から秘密鍵の情報について何らかの情報を得ることは出来ない．$g, \hat{g}$ から $\hat{g} = g^{\omega}$ となるような ω を得て，公開鍵 $e = g^{x_1}\hat{g}^{x_2} = g^{x_1+\omega x_2} = g^{\gamma}$ となるような γ を得たとする．このとき導かれる式は

$$\gamma = x_1 + \omega x_2 \pmod{p}$$
$$x_2 = \frac{\gamma - x_1}{\omega} \pmod{p}$$

であるが，これを満たすような (x_1, x_2) の組は p 通り存在する．元々 x_1 は $\mathbb{Z}_p$ 上の値なので，総当りで探しても p 通りである．よって，攻撃者は無限の能力があったとしても公開鍵から秘密鍵に関する情報を得られないわけである．

このとき，暗号文が正しければ，暗号化の手順 E6 および復号の手順 D4 における値は，

$$d = e^u f^{uv} = (g^{x_1}\hat{g}^{x_2})^u (g^{y_1}\hat{g}^{y_2})^{uv} = g_1^{u(x_1+y_1v)}\hat{g}^{u(x_2+y_2v)} = a_1^{x_1+y_1v}a_2^{x_2+y_2v}$$

によって確かめることが可能である．では，もし仮に $a = g^u, \hat{a} = \hat{g}^{u'}$ となる値が暗号文に用いられた場合はどうなるだろうか．この場合，$d = (g^u)^{x_1+y_1v}(\hat{g}^{u'})^{x_2+y_2v}$ である．このとき，$\hat{g}, d, e, f$ それぞれの値に対しての g を底とした離散対数を考えると，

$$\begin{aligned}
\log_g \hat{g} &= \omega \\
\log_g d &= u_1x_1 + u'x_2 + (u_1y_1 + u'y_2)v \\
\log_g e &= x_1 + \omega x_2 \\
\log_g f &= y_1 + \omega y_2
\end{aligned}$$

である．これらを行列を用いて表すと，

$$\begin{pmatrix} \log_g e \\ \log_g f \\ \log_g d \end{pmatrix} = \begin{pmatrix} 1 & \omega & 0 & 0 \\ 0 & 0 & 1 & \omega \\ u & \omega u' & uv & \omega u'v \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix}$$

となる．行列式を考えれば，$u = u'$ ならば線形従属であり，$u \neq u'$ ならば線形独立になることが分かる．つまり，復号を行うときに a_1 と a_2 の指数が異なるものに対して復号を行なう場合，上記の行列が線形独立になることから正しく復号されることはない．よって，復号において D4 の検証式が通るならば，そのとき受け取っている a と $\hat{a}$ は同じ指数となる．

5.5.1 従来の証明手法による Cramer-Shoup 暗号の安全性証明

Cramer-Shoup 暗号が standard model において，DDH 仮定およびハッシュ関数 HF が target collision resistant (TCR) であるという仮定の下に IND-CCA2 安全であることの証明を行う．

ハッシュ関数の安全性に関しては，ハッシュ関数族 $\mathbf{H} = (\mathbf{H}_h)_h$ （h はパラメーターである）が Target Collision Resistant (TCR) であるとは，$(\mathbf{H}, h, x)$ を受け取ったとき，$\mathbf{H}_h(x) = \mathbf{H}_h(y)$ となるような y $(y \neq x)$ を見つけることが困難である場合をいう．また，ハッシュ関数族が Collision Resistant (CR) であるとは，$(\mathbf{H}, h)$ を受け

取ったとき，$\mathbf{H}_h(x) = \mathbf{H}_h(y)$ となるような (x, y) $(y \neq x)$ の組を見つけることが困難である場合をいう．

Cramer-Shoup 暗号に対しては traditional security proof (1998) と呼ばれる reduction に基づいた証明法と，ゲーム列を用いた証明法の 2 通りがある．まずは traditional security proof による安全性の証明において述べる．

traditional security proof では，ある公開鍵暗号を破るような攻撃者が存在するならば，その攻撃者を使ってある数論仮定を破るような simulator を構成することが出来る，という reduction に基づいて安全性の証明を行う．今回は，IND-CCA2 の能力を用いて Cramer-Shoup 暗号をやぶるような攻撃者が存在するとすると，その攻撃者を用いることで DDH 問題を 1/2 よりも優位な確率で判別できるような simulator を構成することができればよい．

まず，simulator は DDH 問題の tuple $(g, \hat{g}, a, \hat{a})$ を受け取る．このとき，$\log_g a = \log_{\hat{g}} \hat{a}$ であるかどうかの判断を行う．simulator は攻撃者を動作させるため，秘密鍵を自身で $(x_1, x_2, y_1, y_2, z_1, z_2) \in \mathbb{Z}_p^6$ を選び $(e = g^{x_1}\hat{g}^{x_2}, f = g^{y_1}\hat{g}^{y_2}, h = g^{z_1}\hat{g}^{z_2})$ を計算して公開鍵 $(g, \hat{g}, hk, e, f, h)$ を攻撃者に入力する．今回の攻撃者は IND-CCA2 であるので，decryption oracle に暗号文を聞いてくることがある．その場合，simulator は秘密鍵を自身で生成しているため，通常の復号処理と同じ動作を行って結果を攻撃

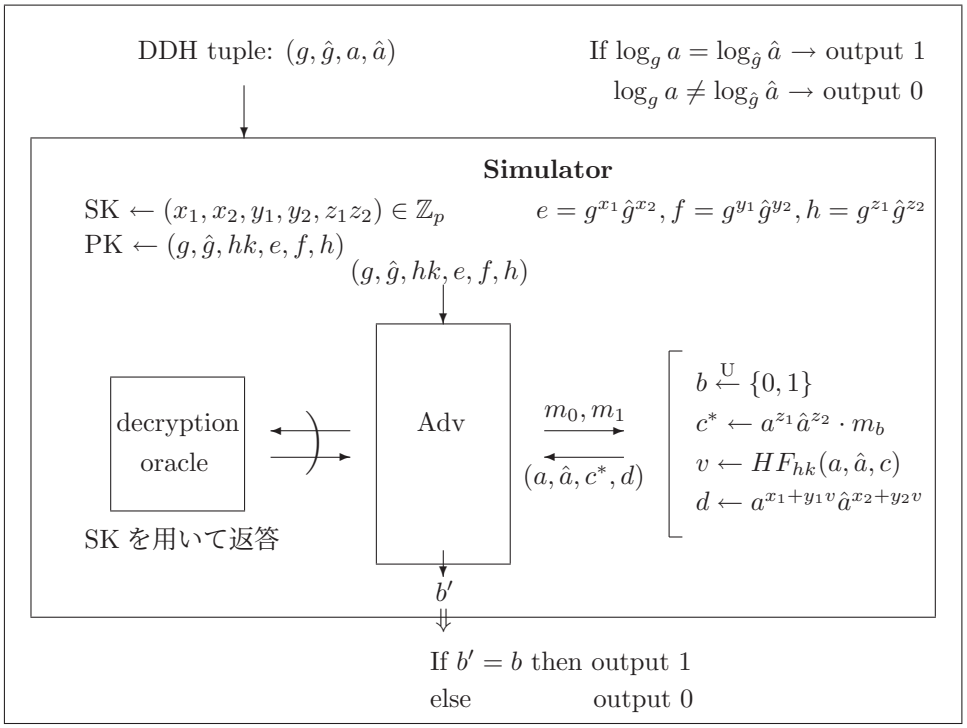


図 8: Cramer-Shoup 暗号に対する traditional security proof

者に返せばよい．攻撃者が平文 m_0, m_1 のどちらかに対しての暗号文を生成するように要求してきた場合，simulator は DDH 問題の tuple として受け取っている値を用いて

$$b \xleftarrow{\mathrm{U}} \{0,1\},\ c^* \leftarrow a^{z_1}\hat{a}^{z_2} \cdot m_b,\ v \leftarrow HF_{hk}(a, \hat{a}, c),\ d \leftarrow a^{x_1+y_1 v}\hat{a}^{x_2+y_2 v}$$

という計算を行い，$(a, \hat{a}, c^*, d)$ を暗号文として攻撃者に返す．攻撃者はこの暗号文を受け取った後も decryption oracle に暗号文を聞くことができ (ただし $(a, \hat{a}, c^*, d)$ 自身を聞くことは禁止する)，それに対しては simulator は秘密鍵を用いることで正しく返答を行うことが出来る．もし攻撃者が m_0, m_1 どちらの平文が暗号化されたのかに対する guess としてビット b' を出力したならば，simulator は $b' = b$ であれば入力された DDH tuple は $\log_g a = \log_{\hat{g}} \hat{a}$ であったとし，$b' \neq b$ であれば入力された DDH tuple は $\log_g a \neq \log_{\hat{g}} \hat{a}$ であったという出力を行う．

5.5.2 ゲーム列による Cramer-Shoup 暗号の安全性証明

次に，Cramer-Shoup 暗号に対してのゲーム列による証明を述べる．ゲーム列による証明では，最初は通常の Cramer-Shoup 暗号に対しての IND-CCA2 ゲーム (Game 0 とする) を考える．そして，そこから攻撃者が識別できないような negligible で抑えられるような部分的変換を施してゲームを変えていき，あるゲームにたどり着いたときには攻撃者が IND-CCA2 ゲームに対しての識別に対する advantage が 0 しかない状態に持っていくことで，元の Game 0 での識別に対する advantage もゲームの最終段階から Game の変換に要した negligible な分を足した程度しかなかった，という変換による証明方法である．

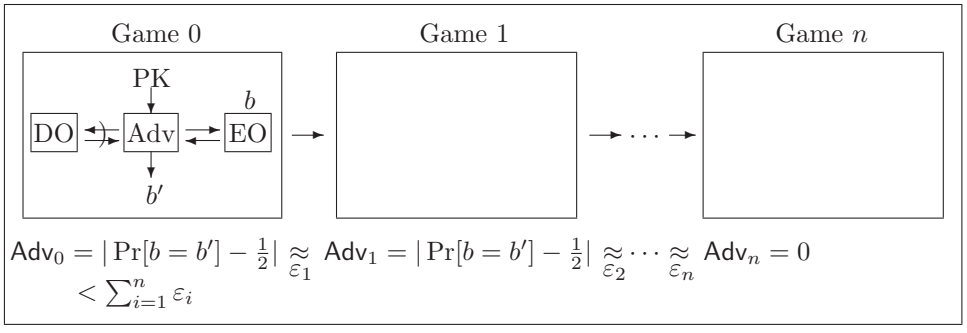


図 9: ゲーム列による証明

このようなゲーム列による証明を Cramer-Shoup 暗号に対して行うと次のようになる．

Game 0: Game 0 は通常の Cramer-Shoup 暗号に対しての IND-CCA2 ゲームである．ここで，Adv は自由に Encryption oracle(EO), Decryption oracle(DO) にアクセスすることが出来る．ただし，DO へのアクセスは $c*$ と同じものを送ってはいけない．

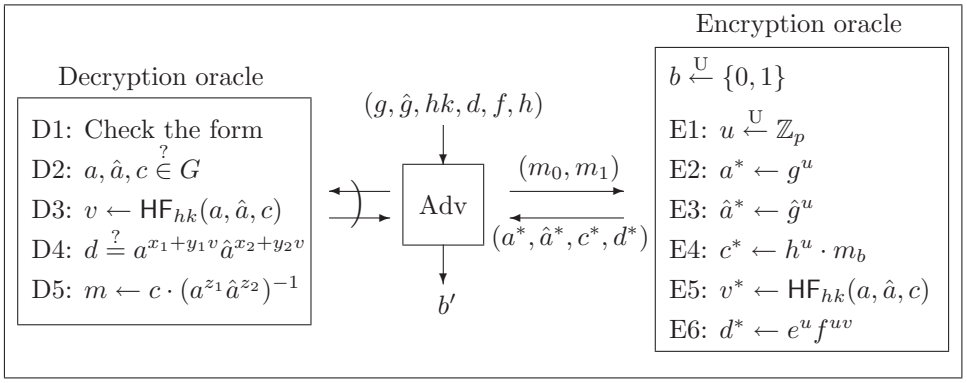


図 10: Game 0

Game 1: Game 1 では，Game 0 における E4 と E6 をそれぞれ

E4: $c^* \leftarrow h^u \cdot m_b \;\Rightarrow\; c^* \leftarrow (a^*)^{z_1} (\hat{a}^*)^{z_2} \cdot m_b$

E6: $d^* \leftarrow e^u f^{uv} \;\Rightarrow\; d^* \leftarrow (a^*)^{x_1+y_1 v^*} (\hat{a}^*)^{x_2+y_2 v^*}$

へと変換する．この変換は conceptual change と呼ばれている変換方法の一つである．送信者が用いる u を用いて書いている値を，受信者の秘密鍵 $(x_1, x_2, y_1, y_2.z_1, z_2)$ を用いることで同じ値を表現しているので，攻撃者からみてまったく同じ値に見える．

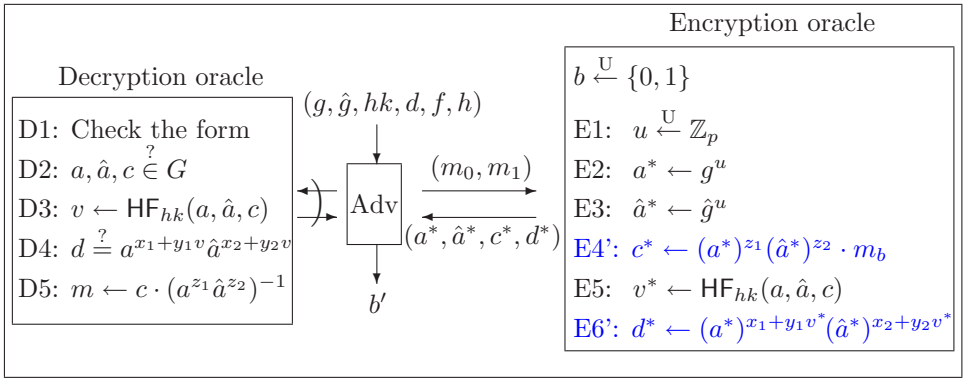


図 11: Game 1

Game 2: Game 2 では，Game 1 における E3 を

$$\text{E3: } \hat{a}^* \leftarrow \hat{g}^u \;\Rightarrow\; \hat{a}^* \leftarrow \hat{g}^{\hat{u}},\ \hat{u} \xleftarrow{\text{U}} \mathbb{Z}_p \setminus \{u\}$$

へと変換する．この変換によって，DDH tuple $(\log_g a = \log_{\hat{g}} \hat{a})$ から non DDH tuple $(\log_g a \neq \log_{\hat{g}} \hat{a})$ へと変更している．もしこの変換によって攻撃者から見て違いが生じるのであれば，それは DDH 問題に対しての advantage があることを意味するため，この変換によって攻撃者が違いを判断する確率は negligible である．

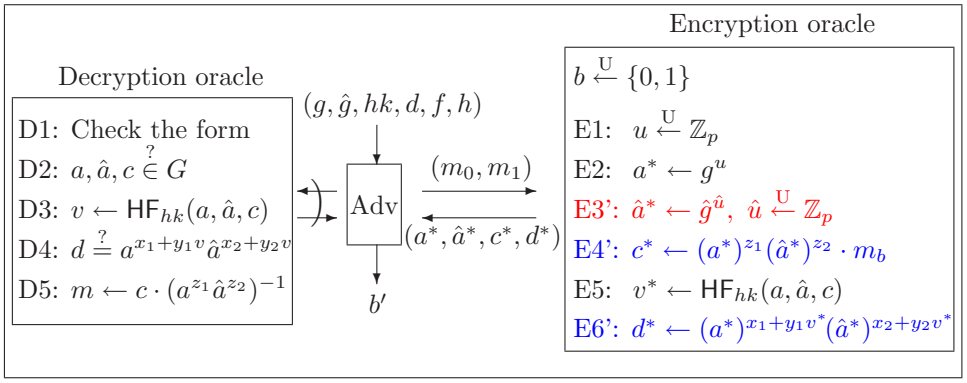


図 12: Game 2

Game 3: Game 3 では，Game 2 における D4 および D5 をそれぞれ

$$\text{D4: } d \stackrel{?}{=} a^{x_1+y_1 v} \hat{a}^{x_2+y_2 v} \;\Rightarrow\; \hat{a} \stackrel{?}{=} a^w,\ d \stackrel{?}{=} a^{x+yv},\ v \stackrel{?}{=} v^* \quad (\text{ただし } e = g^x, f = g^y, h = g^z, \hat{g} = g^w)$$

$$\text{D5: } m \leftarrow c \cdot (a^{z_1} \hat{a}^{z_2})^{-1} \;\Rightarrow\; m \leftarrow c(a^z)^{-1}$$

へと変換する．この変換は，攻撃者が入力とする値 $a, \hat{a}$ に関しての離散対数が異な

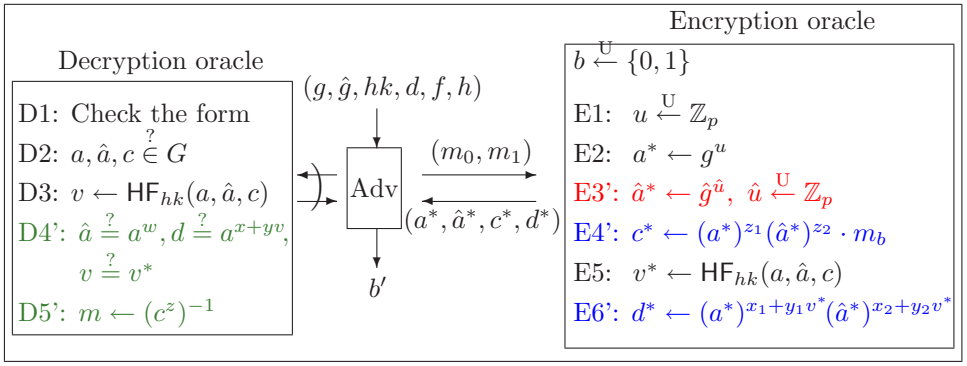


図 13: Game 3

るときに D4 における検証に通るようなものが作れない場合は等価であり，攻撃者が decryption oracle に暗号文を聞く回数を q_{DO} としたとき，運良く検証に通るようなものを見つけるような確率は $\frac{q_{DO}}{p}$ であり，negligible であるといえる．このとき，ハッシュ関数が TCR ハッシュ関数であれば検証に通るようなものを攻撃者が見つけることはできない (※)．**Game 4:** Game 4 では，Game 3 における E4 を

$$\text{E4: } c^* \leftarrow (a^*)^{z_1}(\hat{a}^*)^{z_2} \cdot m_b \;\Rightarrow\; c^* \leftarrow g^r, r \xleftarrow{\text{U}} \mathbb{Z}_p$$

へと変換する．この変換は，Game 3 において既に攻撃者に対して z_1, z_2 の情報を与える機会がなくなっているところから (公開鍵 $h = g^{z_1}\hat{g}^{z_2}$ から z_1, z_2 に関しての情報が得られないことは Traditional Security Proof 参照)，E4 の値 c^* を z_1, z_2 を用いてランダム化させているところを単なる乱数 r を用いて置き換えても攻撃者からみて識別ができない，という考えに基づいている．

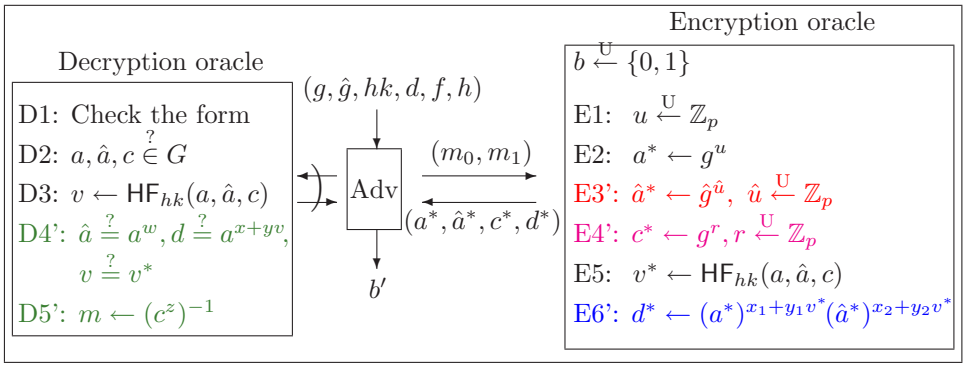


図 14: Game 4

Game 4 の状態に至ると，既にゲームの中で攻撃者に返答している値に関して m_0 の情報も m_1 の情報も含まれていない．つまり，このときの攻撃者の advantage は 0 である．これを逆に辿っていくと，すべてのゲームの変換に関して攻撃者の advantage は negligible であるので，最初の Game 0 における攻撃者の advantage も 0 に negligible な分を足しただけである．よって，ゲーム列による証明によって Cramer-Shoup 暗号は DDH 仮定および TCR 仮定の下で IND-CCA2 安全であることが証明された．

(※) ハッシュ関数が TCR ハッシュ関数でなければいけない理由は以下のためである．まず，攻撃者に入力している値 $a^* = g^u, \hat{a}^* = \hat{g}^{u'}$ に対して，攻撃者が decryption oracle に対して $a = g^t, \hat{a} = \hat{g}^{t'}$ となるような a と $\hat{a}$ の離散対数が異なるような値を用

いて聞いてきた場合,

$$\left.\begin{array}{l} a^* = g^u \\ \hat{a}^* = \hat{g}^{u'} \\ a = g^t \\ \hat{a} = \hat{g}^{t'} \end{array}\right\} \Rightarrow \begin{array}{l} e = g^{x_1}\hat{g}^{x_2} \\ f = g^{y_1}\hat{g}^{y_2} \\ d^* = (a^*)^{x_1+y_1v^*}(\hat{a}^*)^{x_2+y_2v^*} \\ d = a^{x_1+y_1v^*}\hat{a}^{x_2+y_2v^*} \end{array}$$

となる．$\hat{g} = g^\omega$ としてそれぞれ e, f, d^*, d に対して g を底とした離散対数を考えると,

$$\begin{aligned} \log_g e &= x_1 + \omega x_2 \\ \log_g f &= y_1 + \omega y_2 \\ \log_g d^* &= u(x_1 + v^* y_1) + u'\omega(x_2 + v^* y_2) \\ \log_g d &= t(x_1 + v y_1) + t\omega(x_2 + v y_2) \end{aligned}$$

であり，この 4 式を行列を用いて表すと

$$\begin{pmatrix} \log_g e \\ \log_g f \\ \log_g d^* \\ \log_g d \end{pmatrix} = \begin{pmatrix} 1 & \omega & 0 & 0 \\ 0 & 0 & 1 & \omega \\ u & \omega u' & uv^* & \omega u' v^* \\ t & \omega t' & tv & \omega tv \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \end{pmatrix}$$

となる．このときの 4×4 の行列 A に関しての行列式を求めると，$|A| = \omega^2(u-u')(t-t')(v^*-v)$ となる．行列式が 0 でなければ行列 A は正則になるため，このとき d が正しい検証式として通るような値になる確率は $1/p$ である．そして，この行列式が 0 でないための条件としての $v^* \neq v$ は，ハッシュ関数が TCR ハッシュ関数であることにより満たされるのである．

尚，上記ゲーム間における誤差に関しては，一般に次の補題が適用できる．

補題 5.1. *Event1, Event2* について，次が成り立つ．

$$\mathit{Event1} \wedge \neg F = \mathit{Event2} \wedge \neg F \quad \Longrightarrow \quad |\Pr[\mathit{Event1}] - \Pr[\mathit{Event2}]| \leqq \Pr[F]$$

6 擬似乱数と擬似ランダム関数

6.1 確率変数の識別不可能性

確率変数 X_n, Y_n に対しての識別不可能性といった場合，以下の 3 つが考えられる．

- 完全識別不可能：$X_n = Y_n \Leftrightarrow {}^\forall\alpha,\ \Pr[X_n \to \alpha] = \Pr[Y_n \to \alpha]$
- 統計的識別不可能：$\sum_{\alpha\in\{0,1\}^*} |\Pr[X_n \to \alpha] = \Pr[Y_n \to \alpha]| < \epsilon$

- 計算量的識別不可能：任意の多項式時間識別器 D に対して

$$|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| < \epsilon$$

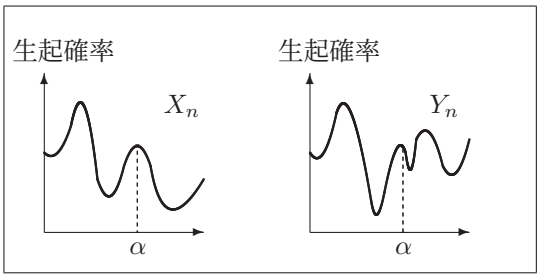


図 15: 2 つの確率変数に関しての生起確率の分布

上図のような確率分布を持つような確率変数 X_n, Y_n において，完全識別不可能である場合は 2 つの図が重なり，統計識別不可能である場合は限りなく U と V の図が似ている場合である．統計識別不可能 (statistically indistinguishable) を別の言い方で表すならば，確率変数族 $\{X_n\}_{n\in\mathbb{N}}, \{Y_n\}_{n\in\mathbb{N}}$ および $\ell(n)$bit ($\ell(n)$ は n の多項式) のビット列に関して統計距離 (statistical distance)

$$\delta(X_n, Y_n) = \sum_{\alpha\in\{0,1\}^{\ell(n)}} |\Pr[X_n \to \alpha] - \Pr[Y_n \to \alpha]| \cdot \frac{1}{2} < \epsilon(n)$$

が negligible であることである．計算量的識別不可能 (computational indistingushability) である場合は，図としてみた場合にはかなりばらつきがある場合でもよく，多項式時間識別器 D からみてどちらか一方の確率分布のみが与えられたとき，それがどちらであるのかが判断できなければ良い．上記の定義を別の言い方で書くと，

$$\left|\Pr[D(x) \to 1 \mid x \xleftarrow{\mathrm{R}} X_n] - \Pr[D(y) \to 1 \mid y \xleftarrow{\mathrm{R}} Y_n]\right| < \epsilon(n)$$

と表すことができる．計算量的に識別不可能な例としては，DDH 問題などがある．

6.2 擬似乱数

ある短い (真の) 乱数 r から，確定的関数 G を用いることで長いビット列 $G(r)$ を出力したとき，$G(r)$ が同じ長さの真の乱数 R と計算量的に識別不可能である場合，$G(r)$ を擬似乱数と呼ぶ．本来の情報理論的なエントロピーの観念からみれば，$G(r)$ のエントロピー $H(G(r))$ は r のエントロピー $H(r)$ と同じであり，真の乱数 R のエントロピー $H(R)$ とはエントロピーが異なる．しかし，計算量的なエントロピーとしてみた場合，$G(r)$ が擬似乱数であるならば，$H(G(r))$ と $H(R)$ とが同じである，と

いうことである．

$$\text{エントロピー（情報理論的）}\quad H(G(r)) = H(r) = |r| << |G(r)|$$
$$\text{疑似エントロピー（計算量的）}\quad CH(G(r)) = H(R) = |R| = |G(r)|$$

ここで $G(r) \approx R$ (計算量的識別不可). ただし，A と B が計算量的識別不可とは，どのような多項式時間アルゴリズム D に対しても次が成り立つことである．

$$|\Pr[D(A) = 1] - \Pr[D(B) = 1]| < \varepsilon$$

例えば，100bit の乱数を確定的関数 G を用いて 1000bit にした場合，情報理論的に見ればエントロピーは 100bit のままであるが，計算量的に見た場合は 1000bit の乱数と識別ができない，ということを言っている．

また，擬似乱数の計算量的識別不可能性に対して，Blum,Micali は 1982 年に next bit test という考えを提案した．next bit test とは，ある短い乱数 r から生成された擬似乱数からある程度の長さのビット列を攻撃者に与えたとき，その攻撃者が擬似乱数の次のビットが 0 であるのか 1 であるのかを判定できない場合，擬似乱数の性質を持つ，と考えたものである．この next bit test は，上記の計算量的識別不可能性と等価であることが同年 Yao によって示された．

6.3 擬似ランダム関数

乱数に対しての擬似ランダム関数があるように，ランダム関数に対して擬似ランダム関数というのも存在する．擬似ランダム関数に対しての計算量的識別不可能性とは，ある識別者がランダム関数か擬似ランダム関数のどちらかにアクセスして出力値を受け取った場合に，どちらの関数にアクセスしているのかを識別が不可能である場合をいう．

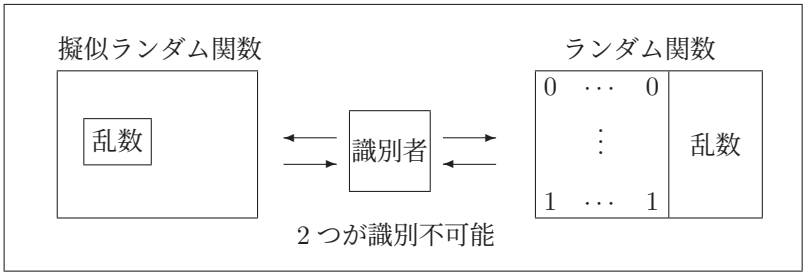


図 16: 擬似ランダム関数の識別不可能性

また，1984 年 Goldreich,Goldwasser,Micali は擬似乱数から擬似ランダム関数が構成できることを示した．まず，擬似乱数は n bit の乱数から $2n$ bit の乱数を生成するようなものとする．このとき，$2n$ bit の乱数を 2 つの n bit の値に分け，そしてその n

bit に対して擬似ランダム関数を用いて $2n$ bit の擬似乱数を生成し，また n bit に分けてそれぞれに擬似乱数を適応し，という操作を繰り返し行う．関数の入力値をビット列で表したとき，そのビットが 1 ならば右側，0 ならば左側，というように辿っていき最終的な擬似乱数として生成された値を返す，という方法である．この操作において，ランダム関数と擬似ランダム関数とが識別できる場合，乱数と擬似乱数との識別ができる，という安全性に基づいている．

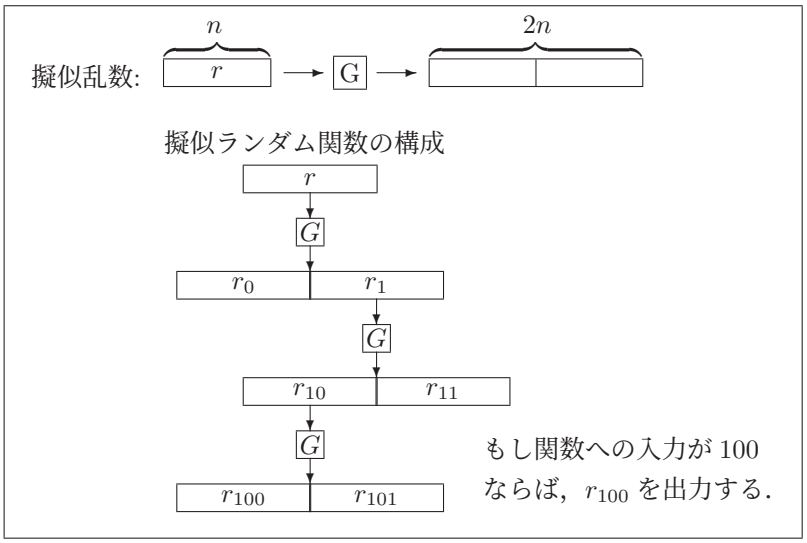


図 17: 擬似ランダム関数の構成例

7 マルチパーティプロトコル

マルチパーティプロトコルとは，複数の人数がそれぞれ秘密情報を持っているとき，その情報を明かさずに何らかの計算を行うプロトコルである．例えば，電子投票などで賛成/反対の票決を行う場合，総和計算によって Alice の秘密 $x_1 \leftarrow \{0,1\}$ と Bob の秘密 $x_2 \leftarrow \{0,1\}$ と Carol の秘密 $x_3 \leftarrow \{0,1\}$ によって $f(x_1, x_2, x_3) = x_1 + x_2 + x_3$ という値を，それぞれの秘密を明かすことなく計算するようなものである．

マルチパーティプロトコルでは参加者が多数いるため，不正者による安全性を考慮しなければならない．このとき，全体の参加者 n 人に対して不正者が t 人いるとき，

- 正確性 (Correctness): $n-t$ 人による出力結果は $f(x_1, \cdots, x_n)$ である．
- 秘匿性 (Security): t 人の不正者によって，残りの $n-t$ 人の中の秘密情報は得ることはできない．

という性質を満たす場合，そのマルチパーティプロトコルは安全であるという．このことに関して，1987 年 Goldreich,Micali,Wigderson はトラップドア一方向性関数 (つまり公開鍵暗号) が存在するならば，任意の関数 f に対して $t < n$ を満たす任意 t の不正者に対して計算量的安全なマルチパーティプロトコルが構成可能である，という完全性定理を証明した [18]．また，1988 年に Ben-Or,Goldwasser,Wigderson は情報理論的に安全な通信路が存在すると仮定した場合，任意の関数 f に対して $t < \frac{3}{n}$ を満たす任意 t の不正者に対して情報理論的安全なマルチパーティプロトコルが構成可能である，というもう一つの完全性定理を証明した [**?**].

定理 7.1 (完全性定理 1（Goldreich-Micali-Wigderson 1987))．トラップドア一方向性関数が存在するならば，任意の関数 f に対して，$t < n$ 人の不正者がいても計算量的に安全なマルチパーティプロトコルが実現可能である．

定理 7.2 (完全性定理 2（Ben-Or-Goldwasser-Wigderson 1988))．情報理論的に安全な通信路が存在すると仮定するならば，任意の関数 f に対して，$t\ (< \frac{n}{3})$ 人の不正者がいても情報理論的に安全なマルチパーティプロトコルが実現可能である．

彼らによる構成では，まずトラップドア一方向性関数からビットコミットメントや紛失通信が作れること，そしてビットコミットメントからコイン投げプロトコルと任意の NP 命題に対してのゼロ知識証明が作れることから，それらを用いて計算量的安全なマルチパーティプロトコルが構成できる，という過程を経ている．

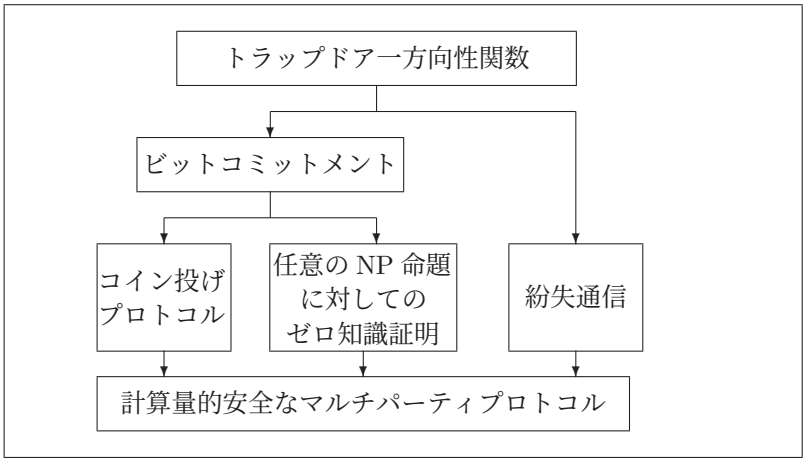


図 18: 計算量的安全なマルチパーティプロトコルの構成

8　ビットコミットメント

ビットコミットメントとは，送り手側があるビット $b \in \{0,1\}$ を相手に知られないような状態にして受け手側に送り (コミットメントフェーズ)，その後でそのビットがどちらであったのかを求めることができるような鍵を送ることで相手が正しく b がコミットメントされて送られてきたことを確認する (オープンフェーズ) ものである．ビットコミットメントでは，

- 秘匿性: コミットメントフェーズの時点では受け手は b に関しての情報を知ることができない．
- 結合性: コミットメントフェーズを終えた後で，受け手が異なるビットに対してのコミットである，と納得させることができてはいけない．

という 2 つの性質が必要とされる．

例えば，Alice が金庫の中に「1」と書かれた紙か「0」と書かれた紙のどちらかを入れて鍵をかけ，Bob に預けたとする．このとき，Bob は金庫の鍵を持っていないのでどちらの紙が入っているかは分からない (秘匿性)．数日後に Alice は金庫の鍵を Bob に渡し，Bob に中身の確認をさせる．このとき，Alice は金庫に「1」とかかれた紙を入れていたならば，Bob に預けた後で『金庫には「0」の紙が入っているだろ』と主張することはできない (結合性)．

9　コイン投げプロトコル

コイン投げプロトコルとは，Alice と Bob とがお互いに通信のやり取りを行い，どちらにもバイアスがかかっていないようなコイン投げの結果を得るようなプロトコルである．例えば，インターネット上で「じゃんけん」などを行おうとすると，通信路などの遅延によって「後出し」のようなことが簡単にできてしまう．しかし，このコイン投げプロトコルを動作させることによって，後出しのような不正のないじゃんけんの正しい結果が得られる．

10　紛失通信

紛失通信 (OT:Oblivious Transfer) とは，Alice が Bob に対して 2 つのメッセージ m_0, m_1 を送ったとき，Bob はどちらか一方を $b \leftarrow \{0,1\}$ として選んで片方のメッセージ m_b のみを受け取れるようなものである．紛失通信では，Alice は送信している時点では Bob がどちらのメッセージを選ぶかを知ることはできない．また，Bob は片方のメッセージのみが得られ，もう片方のメッセージに関しての情報を得ることはできないものという制約がある．

11　ゼロ知識証明

11.1　ゼロ知識

まず，ゼロ知識証明とは何であるかを述べる前に，攻撃者にとって何が知りえる情報で，何が知りえない情報なのかの情報量に関する定式化を行っておく．

- 識別不可能性：攻撃者にとって識別が出来ないような 2 つの情報（確率変数）(例: 真の乱数と擬似乱数) は暗号の観点では同じものとみなす．
- シミュレーション：秘密情報を用いないでシミュレーションを行うことが出来るような情報（確率変数）は，秘密情報を含まない．つまり，秘密情報を漏らさない．

この定式化を用いたとき,「公開情報のみから敵の見える世界 (view) と識別不可能な情報をシミュレーションすること」がゼロ知識ということである.（ゼロ知識証明の概念は 1985 年に提出された.）

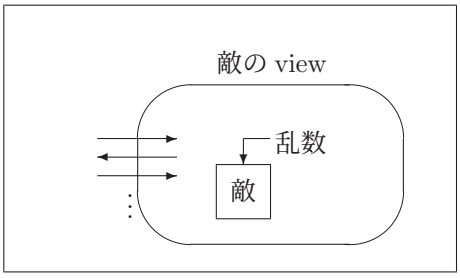


図 19: ゼロ知識

11.2　ゼロ知識対話証明

上記のゼロ知識を用いることでゼロ知識対話証明 (ZKIP: Zero Knowledge Interactive Proof) という以下のようなものを構成することが出来る．ある証明を行いたい証明者 P と，その証明を検証する検証者 V がいるとする．このとき，P が V に証明の正当性を示したとき，V が証明の正当性以外の何の情報も得ていないような証明である．ゼロ知識証明においても，確率変数のときのように (完全/統計的/計算量的)-ゼロ知識 という 3 つを考えることができ，(P, V) が (完全/統計的/計算量的)-ゼロ知識であるとは，どのような V に対しても (PPT アルゴリズム) シミュレータ M_V が存在し，任意の x に対して V が見ることができる情報 $\mathrm{View}_V(x)$ と $M_V(x)$ がほぼ等しい，つまり

$$^{\forall}V, ^{\exists}M_V, ^{\forall}x, M_V(x) \approx \mathrm{View}_V(x)$$

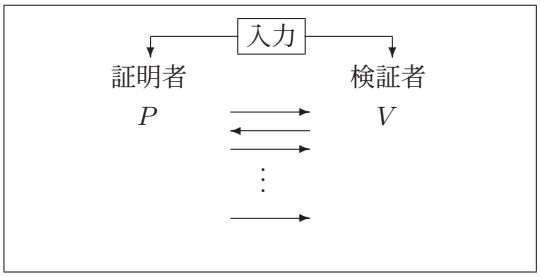


図 20: ゼロ知識対話証明

であることをいう.
ある命題 L に対して (P,V) がゼロ知識対話証明を行う際に求められる条件として以下の 3 つが挙げられる.

1. 完全性: $x \in L \Rightarrow \Pr[(P,V)(x) \text{ 受理}] \approx 1$
2. 健全性: $x \notin L \Rightarrow \Pr[(P,V)(x) \text{ 受理}] \approx 0$
3. (外部入力) ゼロ知識性: ${}^{\forall}V^*,{}^{\exists}M_{V^*},{}^{\forall}x \in L,{}^{\forall}y$ (外部入力), $\mathrm{View}_{V^*}(x,y) \approx M_*V(x,y)$

このとき，完全性および健全性において確率が完全に 1 あるいは 0 でないのは，非常に少ない確率で誤りが存在するためである．また，外部入力を導入しているのは汎用性を考えたときの条件を考えており，何らかの状況で検証者 V が前もっていくらかの情報を持っている可能性を考慮に入れているためである．外部入力ゼロ知識性では，1 つの V に対して 1 つの M_V を決めて証明を行う必要がある.

ゼロ知識対話証明の具体例:平方剰余

検証者 P が，$n = pq$ という n に対してある値 x が平方剰余である ($x = y^2 \pmod n$)，という証明を検証者 V に納得させたい．このとき，ZKIP を用いた証明は以下の手順で行われる.

1. P は $r \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_n$ を選び，$a = r^2 \pmod n$ を V に送る (commit フェーズ).
2. V は $e \overset{\mathrm{U}}{\leftarrow} \{0,1\}$ を選び，e を P に送る (challenge フェーズ).
3. P は $b = y^e \cdot r \pmod n$ を求め，b を V に送る (response フェーズ).
4. V は $b^2 = x^e \cdot a \pmod n$ であるかの検証を行う.

この操作を n 繰り返し，もし一回でも検証に失敗したならば V は正しくないとして abort する．もしすべてに成功したならば，V は P が平方剰余の値を知っているのだと納得する.

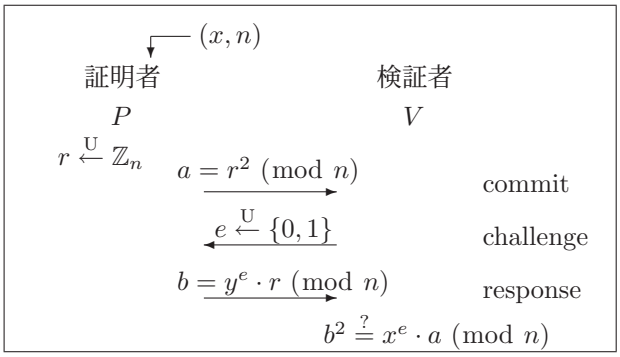


図 21: 平方剰余に対するゼロ知識対話証明

11.3 外部入力/ブラックボックスゼロ知識証明

先ほどのゼロ知識では，1 つの V に対して 1 つの M_V を決めるとゼロ知識を示すことが出来る，という定式化であった．すなわち，
外部入力ゼロ知識証明：

$$\begin{aligned} &{}^{\forall}V^* \text{ (検証者)}, \exists M_{V^*} \text{ (シミュレータ：多項式時間 TM)}, \\ &\mathrm{View}_{V^*}(x, y) \approx M_{V^*}(x, y) \quad \text{over } (x, y) \in L \times \{0, 1\} \end{aligned}$$

しかしながら，V の異なる動作毎にシミュレータ M_V を変える必要があるため，個別の V に対して証明を行うことは難しいという欠点を持っている．そのため，より強い概念として 1 つの M さえあればどんな V^* に対してのゼロ知識性が保たれる，つまり
ブラックボックスゼロ知識証明

$$\begin{aligned} &{}^{\exists}M \text{ (シミュレータ)}, {}^{\forall}V^*, {}^{\forall}x \in L, \\ &M^{V^*}(x) \approx \mathrm{View}_{V^*}(x) \end{aligned}$$

という定式化があり，これをブラックボックスゼロ知識と呼ぶ．ブラックボックスゼロ知識を用いたゼロ知識対話証明も定式化を行うことができ，先ほどの平方剰余の例で考えると，

- 完全性:

  これは，(P, V) が正直者であれば

$$\begin{aligned} e = 0 \quad &\Rightarrow \quad b = r \\ e = 1 \quad &\Rightarrow \quad b = yr \end{aligned}$$

であることから $x \in L,\ \Pr[(P,V)(x)\ \text{受理}] = 1$ である．

- 健全性:

  V は正直者であるが，$\tilde{P}$ は正直者でない (無限の計算能力かついかなる戦略も取れる) とする．この場合，

$$\begin{aligned} e = 0 \quad &\Rightarrow \quad b^2 = a \pmod{n} \\ e = 1 \quad &\Rightarrow \quad b'^2 = xa \pmod{n} \end{aligned}$$

  である．もし，この 2 つを同時に満たすのであれば，

$$x = \left(\frac{b'}{b}\right)^2 \pmod{n}$$

  であることから，元の x が平方非剰余であることに矛盾する．よって

$$x \notin L,\ {}^{\forall}\tilde{P},\ \Pr[(\tilde{P},V)(x)\ \text{受理}] < 1/2^{|n|}$$

  である．

- (ブラックボックス) ゼロ知識性

  この場合では，P は正直者であるが，$\tilde{V}$ は正直者でない (ただし多項式時間チューリングマシン) とする．このとき，いかなる V に対してもシミュレーションができるような $M^{\tilde{V}}$ を ($\tilde{V}$ をブラックボックスとして使う) 以下のように構成することができる．

  (1) $\tilde{V}$ が $e^* \overset{\mathrm{U}}{\leftarrow} \{0,1\}$ として選ぶであろうチャレンジビットに対しての guess を行う．

  (2) $b \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_n^*$ を選び，$a = b^2/x^{e^*}$ を求める．

  (3) $\tilde{V}$ を動作させ，$\tilde{V}$ に a を入力する．

  (4) $\tilde{V}$ が e を返答してきたら，

  $e = e^*$ ならば $\Rightarrow b$ を送り，次のラウンドへと移る．

  $e \neq e^* \Rightarrow$ (1) へと戻る ($\tilde{V}$ をリセットする)．

  この構成によって，$\tilde{V}$ がどのような不正を行おうとシミュレーションを行うことができ，このゼロ知識対話証明を $|n|$ ラウンド行うためには，上記のシミュレーションを平均 $2|n|$ 回行えばよい．このシミュレーションによって，${}^{\forall}x \in L_{QR_n}, \mathrm{View}_{\tilde{V}}(x) = M^{\tilde{V}}(x)$，つまり完全識別不可能である，ということができる．

上記のシミュレーションにおいては毎回 1 ラウンドごとに guess を行い，成功したら次のラウンドへ，という形式を用いていた．しかしながら，このシミュレーションは並列的に動作させてはいけない．もし並列的に $|n|$ ラウンド分を動作させた場合，$\tilde{V}$ が選ぶであろう $|n|$ 個分の e^* をすべて当てる確率は $1/2^{|n|}$ である．そのため，並列的に動作させた場合はシミュレーションの guess に成功する確率が $1/2^{|n|}$ となり，うまく動作させることができないためである．

また，この並列的な動作に関しては，一般的にブラックボックスゼロ知識の場合は 3move のものは並列的なものが構成することができない，という結果が発表されている．

11.4 計算量のクラスとゼロ知識証明

Goldwasser,Micali,Wigderson は 1986 年に，ビットコミットメントがあれば NP に関するすべての問題 (より一般的に PSPACE) は計算量的ゼロ知識証明できる，ということを示した [17]．また，ビットコミットメントは擬似乱数生成器が存在するならば構成できることを Naor が 1986 年に示し，擬似乱数生成器は一方向性関数から構成できることを Håstad,Impagliazzo,Levin,Luby が 1990 年に示したことで,「一方向性関数から計算量的ゼロ知識証明が構成できる」といえるようになった [19]．また，統計的ゼロ知識証明に関しては，CO-NP と NP の共通部分に関しては構成できることが示されている．

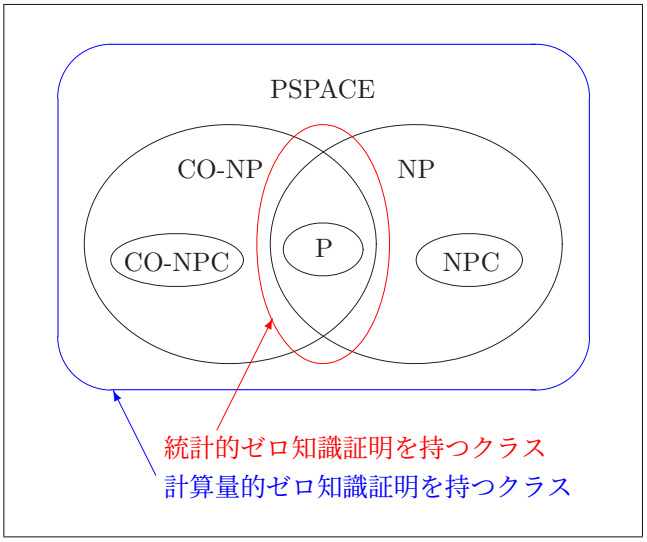


図 22: ゼロ知識証明と計算量のクラス

図における (CO-)NPC というのは，(CO-)NP の中でも最も強い問題で，(CO-)NPC に属する問題が解けるならば (CO-)NP の問題はすべて解ける，というような問題である．NPC の代表例としてはハミルトン閉路問題などがある．

ゼロ知識証明の合成は，逐次適合性 (sequential composition) と 並列的合成 (parallel composition) があり，concurrent ゼロ知識証明として非同期並列合成（インターネットでのゼロ知識証明の合成）等がある．

現在のゼロ知識対話証明の問題点は，先ほど平方剰余の例でも挙げたように並列的な動作が行えないこと (Concurrent 結合に関して閉じていない)，そして適応的攻撃者に対しては対応していない点などが挙げられる．

12 汎用的結合可能性 (Universal Composability: UC)

安全性に関して，シミュレーションによる定式化がなされている．これにより，さまざまな安全性の暗号を組み合わせてハイブリッド暗号を作り上げたときの安全性の保障がなされるようになった．これを汎用的結合可能性 (Universal Composability：UC) という．UC は，2001 年に Canetti が提唱した新しいパラダイムで，単体で保証された安全性がどのような利用環境でも保持される，という新しい安全性の概念である [7]．どのような環境においても安全であることを要求するため，従来の定式化による安全性よりも強い安全性が求められる．また，UC においてはすべての暗号機能 (公開鍵暗号，署名，ビットコミットメント，ゼロ知識証明など) の安全性を統一的に定式化することができるようなフレームワークを用いている．

どのような環境においても安全であることを定式化するためには，そのプロトコルが理想的な状況においてエミュレートできればよい．あるプロトコル π に対して理想的な機能 F を考えたとき，その機能によって理想的な状況と (攻撃者 A を含む) 現実的な状況とを区別できないような振る舞いを行うようなシミュレータ S が存在するならば，そのプロトコルは理想的な状況においてもエミュレートできている，言うことができる．理想的な状況 (理想プロセス) と現実的な状況 (プロトコル実行) とを区別するものを環境 Z として上記を UC の定義とすると

$$^{\forall}A, ^{\exists}S, \text{ s.t. } ^{\forall}Z \text{ は理想プロセスとプロトコル実行とを区別できない．}$$

ということができる．

例 1: 鍵交換における理想機能 F_{KE}

鍵交換における理想機能 K_{KE} は以下のように考えることができる．まず A と B が鍵交換を行うものとし，攻撃者が間にいるとする．A と B が session を行うとき，まず両者は理想機能 F_{KE} に対して鍵交換をする旨を伝える．ある 1 つの session 毎に割り当てられている s という値を用いて，(A, B, s) と表すことにする．F_{KE} はそれを受け取ると，攻撃者に対して (A, B, s) という session を構成することを伝える．攻撃者

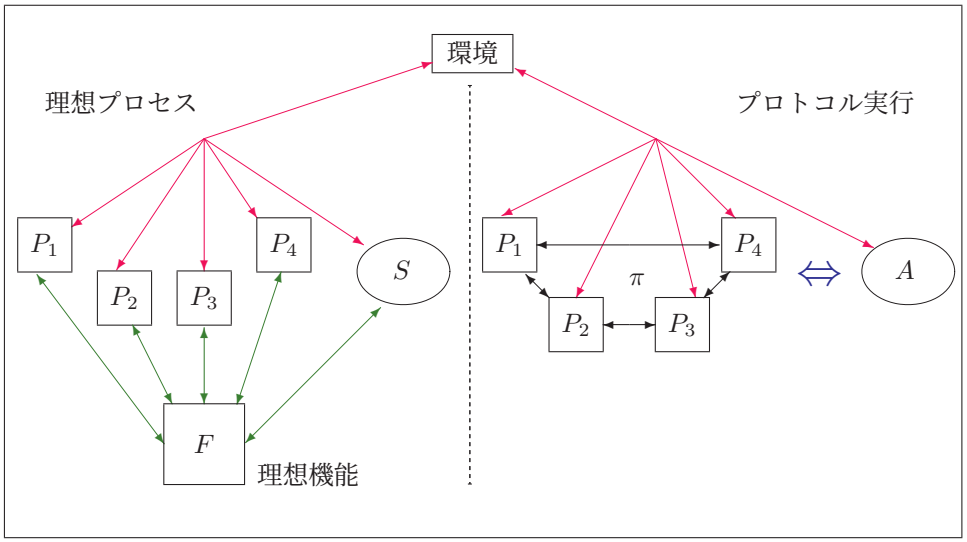


図 23: 汎用的結合可能性

が OK という指示を出すと，F_{KE} はその鍵交換における session key $SK \stackrel{\mathrm{U}}{\leftarrow} \{0,1\}^n$ を選び，A と B に対して SK を渡す．

つまり，鍵交換を理想的な状況において考えた場合，session key は理想的に random に選ばれ，それが Alice と Bob に伝わればよいわけである．

例 2: ゼロ知識証明における理想機能 F_{ZK} ゼロ知識証明における理想機能 K_{ZK} は以下のように考えることができる．まず証明者 P と検証者 V がゼロ知識証明を行うものとし，攻撃者が間にいるとする．理想機能 F_{KE} によってゼロ知識証明を行うために，P はある session s において共通に入力されている x および証明したいものの証拠 w を用いて，(P, V, x, w, s) を理想機能 F_{ZK} に伝える．このとき，V は (P, V, x) のみを F_{ZK} に伝えるものとする．F_{ZK} は x と w に関しての関係 R を導き，攻撃者に $(P, V, x, R(x, w), \mathsf{s})$ を行うことを伝える．攻撃者から OK という指示が出されたら，F_{ZK} は V に対して $(P, V, x, R(x, w), \mathsf{s})$ を伝える．

このとき，検証者 V が x を受理した場合は $R(x, w) = 1$ であり健全性を満たし，また理想機能によって V は $R(x, w)$ 以外の情報は得ていないので，ゼロ知識性を満たしていると言うことができる．

これらの理想機能を用いてプロトコル π が F を安全に実現しているときに関して，どのような組み合わせを持ってしても単体としての安全性が保障されていることに対

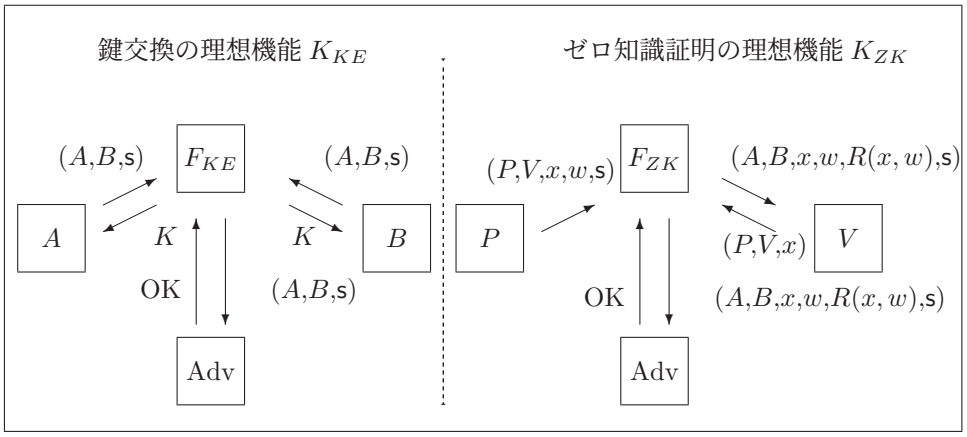


図 24: 鍵交換およびゼロ知識証明の理想機能

する定式化を考える．ある他のプロトコルの中で π を動作させたときの安全性を考えるため，プロトコル ρ^F というものがあり，ρ^F は F へアクセスしているものとする．このときにプロトコルを組み合わせる，つまり結合プロトコル ρ^π が実現しているとは，ρ^F における F へのアクセスを π へのサブルーチンコールへと置き換え，π から返答される値を F からの応答へと対応させる，ということを行うことで定めることができる．また，プロトコルを動作させるパーティが複数いる場合には並列的に処理が行われることを加味するため，ρ^F において F へのアクセスする際は複数の F コピーを考え，それぞれのパーティが各々の F に対してアクセスしているものと定める．それを ρ^{pi} において考えた場合は，F の代わりにサブルーチンコールされる π が複数あり，それが並列的に実行されているものである，と捉える．

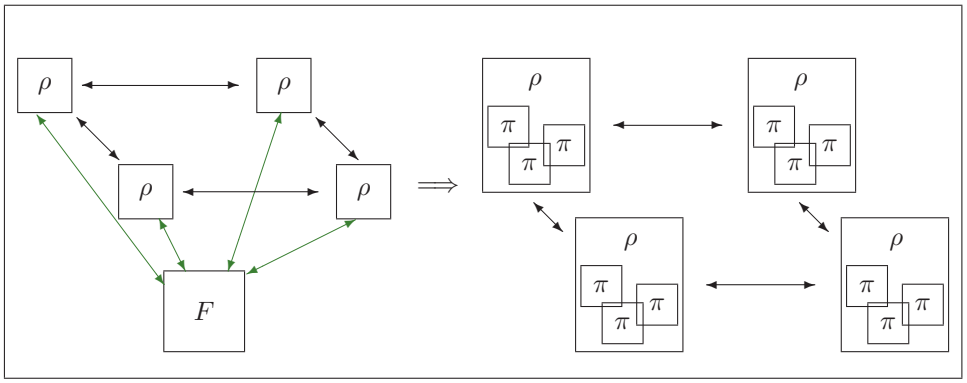


図 25: 結合処理

このような結合処理に関して，2001 年に Canneti は UC 定理と呼ばれる，汎用的結

合性定理 (The Universal Composition Theorem) を導いた．それは，上記の ρ^{π} に対しての攻撃者 A に対し，ρ^{F} に対しての攻撃者 A' が存在し，そのとき環境 Z が ρ^{F} であるか ρ^{π} であるかが識別できない，つまり ρ^{π} は ρ^{F} をエミュレートしている，という定理である．

UC 定理： $^{\forall}A,^{\exists}A',^{\forall}Z, Z$ はρ^{F}とρ^{π}のインタラクションを区別できない．

また，ここから導かれる系として，(ρ^{F}, A') が安全に機能を実現しているならば，(ρ^{π}, A) も安全に機能を実現している，という結果が得られる．

様々な暗号プロトコルがある中で，UC 安全という安全性と既存のプロトコルに対して定式化されている安全性に関しては以下のことが示されている．まず，正しい動作を行う人が過半数を超えているならば，どのような機能も UC 安全を実現可能である，という結論が得られている．例えば，3 人中 2 人が正しい動作を行うならば，UC 安全を実現できる．しかしながら，ビットコミットメントやコイン投げなどの 2 人によるプロトコルの場合は標準モデルの下では UC 安全を実現不可能であることが知られており，その代わりに Setup の段階で何らかの共通とされている入力値を用いた CRS(Common Reference String) モデルというモデルにおいては UC 実現が可能であることが示されている．また，公開鍵暗号に関しては IND-CCA2 を満たしていることが UC 安全と等価であることが示されており，署名に関しては EUF-CMA を満たしていることが UC 安全と等価であることが示されている．

13　2 者間鍵共有プロトコル

2 者間鍵共有プロトコルとは，盗聴などが起きても構わないような通信ネットワーク上で Alice と Bob の 2 者が値のやり取りを行い，そこから Alice と Bob のみにしか求めることが困難であるような session key を共有するプロトコルである．しかしながら，近年では攻撃者が盗聴だけでなく値の改変を行ったり，あるいはパーティの秘密の情報などが得られても session key が安全であることを考える Authenticated Key Exchange という安全性を考えるものが主流になっている．まずは，一番基本的な Diffie-Hellman 鍵交換から，様々な攻撃やその対抗策などを紹介していかに鍵共有プロトコルを構成すべきかを考察する．

13.1　プロトコルと攻撃例

13.1.1　Ephemeral Diffie-Hellman 鍵交換

Ephemeral Diffie-Hellman 鍵交換は，5.2 章にて述べた以下のような鍵共有のプロトコルである [11]．ここでは，2 人のパーティを $\hat{A}$ および $\hat{B}$ で表すことにする．

1. G を素位数 p の有限可換群とし，G の要素（単位元以外）を $g \in G$ とする．
2. $\hat{A}$ は $x \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $X \leftarrow g^x$ を計算し，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．
3. $\hat{B}$ は $y \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $Y \leftarrow g^y$ を計算し，$(\hat{A}, \hat{B}, Y)$ を $\hat{A}$ に送る．
4. $\hat{A}$ は session key $SK_A \leftarrow Y^x$ を出力する．
5. $\hat{B}$ は session key $SK_B \leftarrow X^y$ を出力する．

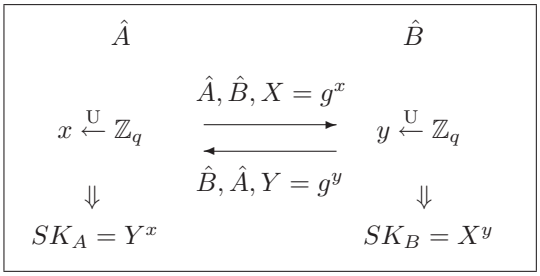


図 26: Ephemeral DH 鍵交換プロトコル

この Diffie-Hellman 鍵交換は，(g, g^a, g^b) が与えられたとき，g^{ab} を求めることは難しいという CDH (Computational Diffie-Hellman) 問題や，(g, g^a, g^b, g^{ab}) あるいは (g, g^a, g^b, g^c) $(c \neq ab)$ のどちらか一方を受け取ったとき，どちらであるのかの区別をすることが困難である DDH (Decisional Diffie-Hellman) 問題を安全性の根拠としている．攻撃者が X や Y の値を盗聴しているだけであれば Diffie-Hellman 鍵交換は安全であること示すことが出来る．しかしながら，攻撃者が鍵交換を行う際の値を改変させた場合，Diffie-Hellman 鍵交換は安全ではない．これは **Man-in-the Middle Attack (MIM Attack)** という攻撃で，攻撃者が値を改変させた場合に Diffie-Hellman 鍵交換は安全でないことを以下のように示すことができる．

1. $\hat{A}$ は $x \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $X \leftarrow g^x$ を計算し，$(\hat{A}, \hat{B}, X)$ を B に送る．
2. 攻撃者 $\hat{C}$ は $(\hat{A}, \hat{B}, X)$ が $\hat{B}$ に届く前に通信上から削除し，$z \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $Z \leftarrow g^z$ を計算し，$(\hat{A}, \hat{B}, Z)$ を $\hat{B}$ に送る．
3. $\hat{B}$ は $y \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $Y \leftarrow g^y$ を計算し，$(\hat{B}, \hat{A}, Y)$ を $\hat{A}$ に送る．
4. $\hat{C}$ は $(\hat{B}, \hat{A}, Y)$ が $\hat{A}$ に届く前に通信上から削除し，$w \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び $W \leftarrow g^w$ を計算し，$(\hat{A}, \hat{B}, W)$ を $\hat{A}$ に送る．
5. $\hat{A}$ が $SK_A \leftarrow W^x$ を出力し，$\hat{B}$ が $SK_B \leftarrow Z^b$ を出力するのに対し，$\hat{C}$ は $SK'_A \leftarrow X^z$ および $SK'_B \leftarrow Y^z$ を出力する．

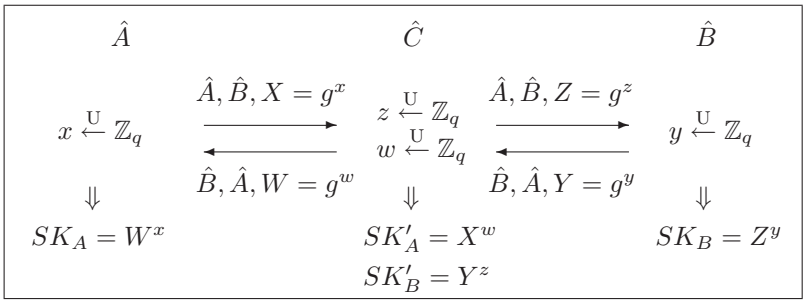


図 27: Ephemeral DH 鍵交換プロトコルに対しての MIM Attack

Alice と Bob はお互いに鍵を共有しているものだと思っているにもかかわらず，DH 鍵交換によって出力した値は攻撃者と共有しているものであり，その後 session key を用いて共通鍵暗号などを用いて通信したとしてもその通信内容は攻撃者に知られてしまう．

こうした攻撃を避けるためには認証付き鍵交換 (AKE: Authenticated Key Exchange) が必要となり，それには二つのタイプがある：

1. Public-key infrastructure(PKI)-based AKE.
   Authentication for messages/parties is ensured by PKI.
2. Password@based AKE(PAKE)
   Authetication for messages/parties is ensured by password.

13.1.2 Static Diffie-Hellman 鍵交換プロトコル

PKI-AKE の一つの解答が Static Diffie-Hellman 鍵交換プロトコルであり，Static Diffie-Hellman 鍵交換プロトコルは Ephemeral Diffie-Hellman 鍵交換において値のやり取りを行っていた所を，公開鍵/秘密鍵を用いることによって代用したプロトコルである．この場合，2 人のパーティ $\hat{A}, \hat{B}$ はおのおの以下のような動作を行う．

1. $\hat{A}$ は $a \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び，$A \leftarrow g^a$ を公開鍵として公開しておく．
2. $\hat{B}$ は $b \xleftarrow{\mathrm{U}} \mathbb{Z}_p$ を選び，$B \leftarrow g^b$ を公開鍵として公開しておく．
3. $\hat{A}$ は公開鍵を持っていることを示すための証明書 $cert(A)$ を用いて $(\hat{A}, cert(A))$ を $\hat{B}$ に送る．
4. $\hat{B}$ は $(\hat{B}, cert(B))$ を $\hat{A}$ に送る．
5. $\hat{A}$ は session key $SK_A \leftarrow B^a$ を出力する．

6. $\hat{B}$ は session key $SK_B \leftarrow A^b$ を出力する．

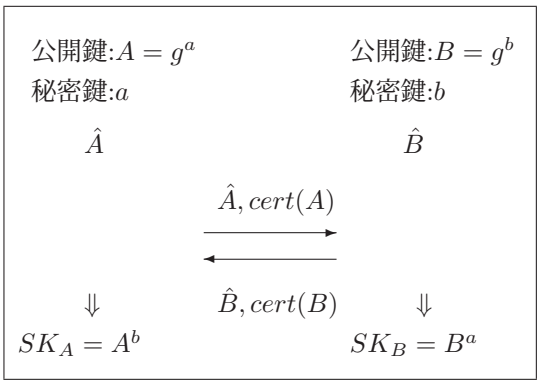


図 28: Static DH 鍵交換プロトコル

このような鍵交換であれば，先ほどのような値の改変などが起こったとしても，計算を行う際にはすべて相手の公開鍵と自身の秘密鍵のみを用いているため，攻撃者は $\hat{A}$ や $\hat{B}$ の session key を求めることは出来ない．

しかしながら，このようなプロトコルなら安全かというと，そうではない．上記の鍵交換では常に session key が同じ値になるため，毎日鍵交換を行ったとしても常に同じ session key しか出力されることはない．仮に攻撃者がある段階でこのパーティによる session key を知ったとすると，それ以降に鍵交換が行われたとしても攻撃者は session key 知ってしまう．このような安全性は known key security と呼ばれており，session key が漏洩したとしても他の session key が求まらないことが鍵交換のよりよい安全性として求められている．

この攻撃に対しては Static Diffie-Hellman を改良し，鍵共有を行う都度に値が足されるカウンターとハッシュ関数を用いて session key を $H(\text{counter}, g^{ab})$ とする方法によって回避することが出来る．しかしながら，鍵交換には秘密鍵の漏洩に対しての安全性として **Forward secrecy** という安全性がある．秘密鍵が漏洩したとしても，過去に行われた session の session key は求められないこと，というのが Forward secrecy の定義である．Forward secrecy を考えた場合は，counter の有無に関わらず Static Diffie-Hellman 鍵交換は秘密鍵によって session key を求めることが出来るため安全ではない，と結論付けられる．

13.1.3 Blake-Johnson-Menezes 鍵交換

先ほどの Ephemeral DH 鍵交換と Static DH 鍵交換を合わせたものが，Blake-Willson, Johnson, Menezes によって提案された BJM 鍵交換プロトコルである [3].

これは公開鍵/秘密鍵のペアを持っている中でさらに値のやり取りを行い，session key を導く以下のようなプロトコルである．

1. $\hat{A}$ は $a \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び，$A \leftarrow g^a$ を公開鍵として公開しておく．
2. $\hat{B}$ は $b \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び，$B \leftarrow g^b$ を公開鍵として公開しておく．
3. $\hat{A}$ は $x \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び $X \leftarrow g^x$ を計算し，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．
4. $\hat{B}$ は $y \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び $Y \leftarrow g^y$ を計算し，$(\hat{A}, \hat{B}, Y)$ を $\hat{A}$ に送る．
5. $\hat{A}$ はハッシュ関数 H を用いて session key $SK_A \leftarrow H(Y^x, B^a)$ を出力する．
6. $\hat{B}$ は session key $SK_B \leftarrow H(X^y, A^b)$ を出力する．

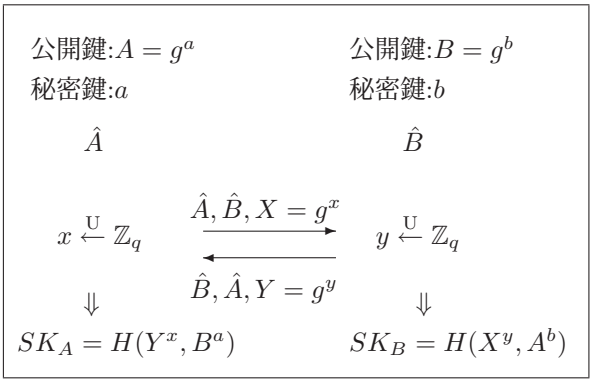


図 29: BJM 鍵交換プロトコル

では，BJM プロトコルは安全であろうか．攻撃者が値の改変を行ったとしても，Ephemeral DH 鍵交換の影響により攻撃者は g^{ab} という値を計算量的に求めることは困難である．また，秘密鍵が漏洩した状況を考えたとしても g^{xy} は攻撃者によって求められない．

しかしながら，BJM プロトコルは並列的な動作を行なった場合に Known key security が満たされていないことが以下のような攻撃によって示されている．

1. $\hat{A}$ は 1 回目の session において $x \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び $X \leftarrow g^x$ を計算し，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．
2. $\hat{A}$ は 2 回目の session において $x' \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ を選び $X' \leftarrow g^{x'}$ とし，$(\hat{A}, \hat{B}, X')$ を $\hat{B}$ に送る．
3. 攻撃者 $\hat{C}$ は $\hat{A}$ の 1 回目の session に対して $(\hat{A}, \hat{B}, X')$ を送る．

4. $\hat{C}$ は $\hat{A}$ の 2 回目の session に対して $(\hat{A}, \hat{B}, X)$ を送る．

5. $\hat{A}$ は 1 回目の session における session key として $SK_A \leftarrow (X')^x$ を出力し，2 回目の session における session key として $SK'_A \leftarrow (X)^{x'}$ を出力する．

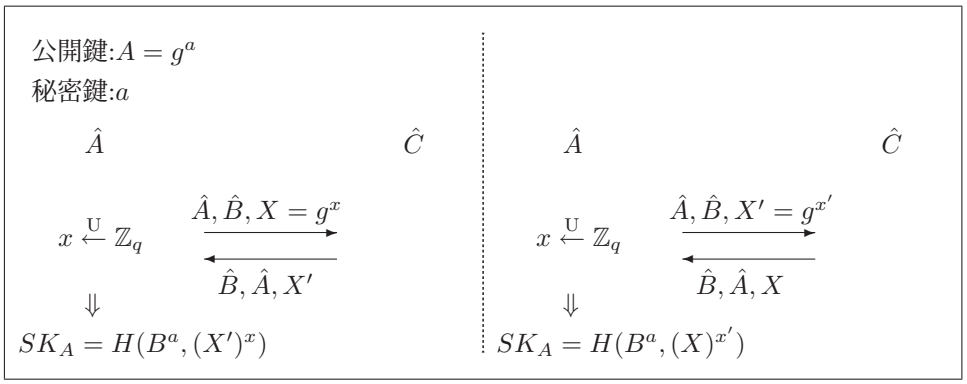


図 30: BJM 鍵交換プロトコルに対しての Known Key Attack

攻撃者が上記のような動作をした場合，$\hat{A}$ は並列的に動作させた session において同じ session key を出力している．つまり片方の session key が漏洩した場合に，もう片方の session key が攻撃者に知られることになる．

また，この BJM 鍵交換プロトコルの Known key Security に対しての脆弱性に対し,「MAC をつけることによって安全になるのではないか」という改良プロトコルが考えられた．MAC(Message Authentication Code) というのは，共通鍵暗号を用いたメッセージ認証で，秘密鍵 k およびメッセージ m を入力して MAC $\sigma = \mathsf{MAC}_k(m)$ を出力するような関数である．このとき，いかなるメッセージと MAC のペア $(m, \sigma = \mathsf{MAC}_k(m))$ が与えられたとしても，秘密鍵 k さえ知られなければ検証に通るような $(m', \sigma' = \mathsf{MAC}_k(m'))$ を生成することは出来ない，という偽造不可能性が保証されている．このような MAC を用いた BJM 鍵交換は以下のようになる．

1. $\hat{A}$ は $a \xleftarrow{U} \mathbb{Z}_p$ を選び，$A \leftarrow g^a$ を公開鍵として公開しておく．

2. $\hat{B}$ は $b \xleftarrow{U} \mathbb{Z}_p$ を選び，$B \leftarrow g^b$ を公開鍵として公開しておく．

3. $\hat{A}$ は $x \xleftarrow{U} \mathbb{Z}_p$ を選び $X \leftarrow g^x$ を計算し，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．

4. $\hat{B}$ は $y \xleftarrow{U} \mathbb{Z}_p$ を選び，$Y \leftarrow g^y$ を計算する．また，ハッシュ関数 H を用いて $k_1 \leftarrow H(X^y, A^b)$ を求め，MAC 関数を用いて $h \leftarrow \mathsf{MAC}_{k_1}(\hat{B}, \hat{A}, Y, X)$ を求めて $(\hat{A}, \hat{B}, Y, h)$ を $\hat{A}$ に送る．

5. $\hat{A}$ は $k_1' \leftarrow H(X^y, A^b)$ から $h = \mathsf{MAC}_{k_1'}(\hat{B}, \hat{A}, Y, X)$ であるかどうかを検証する．検証が正しければ，$h' \leftarrow \mathsf{MAC}_{k_1'}(\hat{A}, \hat{B}, X, Y)$ を求めて $(\hat{A}, \hat{B}, h')$ を $\hat{B}$ に送る．そして，session key $SK_A \leftarrow H(Y^x, B^a)$ を出力する．

6. $\hat{B}$ は $h = \mathsf{MAC}_{k_1}(\hat{A}, \hat{B}, X, Y)$ であるかどうかを検証し，検証が正しければ session key $SK_B \leftarrow H(X^y, A^b)$ を出力する．

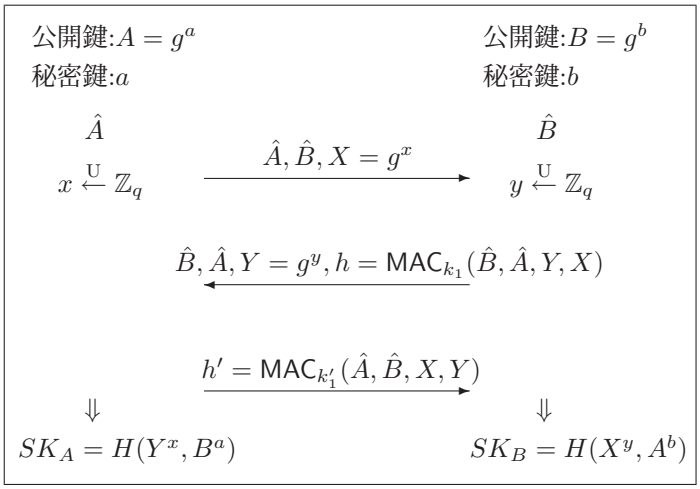


図 31: MAC 認証つき BJM 鍵交換プロトコル

今回の場合，MAC の鍵として用いている値 $k = H(g^{xy}, g^{ab})$ に関して攻撃者は g^{ab} となる値を計算することができない限り k を求めることができない．よって，MAC がついているならば値の改ざんなどが行われた場合に検出することができ，先ほどのように並列的に動作を起こした場合であっても安全性が保たれる．

しかし，鍵交換には他にも安全性としての条件がある．このプロトコルに対して安全性を破るような攻撃として，**Key Compromise Impersonation (KCI) Attack** というものがある．これは，Alice の秘密鍵が漏れているような状況を考えたとき，Alice 以外のパーティに成りすますことができるかどうかを捉えたものであり，もし Alice 以外のパーティに成りすますことができるならば，そのプロトコルは KCI resistance を満たさない，と考える．上記のプロトコルにおいて攻撃者が Alice の秘密鍵を得ている状況で Bob に成りすまし (このとき攻撃者は Bob の秘密鍵は知らない)，Alice と session key を共有できるかどうかを考えてみる．Alice の出力 $X = g^x$ に対して，本来ならば MAC の鍵 $H(g^{xy}, g^{ab})$ が攻撃者にとって求めることができないから安全性が保たれていた．しかしながら，もし Alice の秘密鍵が漏洩していたとしたら，攻撃者は a の知識を用いることで $B^a = g^{ab}$ を計算することが可能になるため，攻撃者が

$z \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ から $Z \leftarrow g^z$ とし，MAC を $k' \leftarrow H(X^z, B^a)$ によって Alice が正しいと認識するような MAC を生成することも可能であり，session key も求めることができる．

13.1.4 署名付 DH 鍵交換プロトコル

今度は，MAC による認証ではなく署名によって相手が正しいことを検証するようなプロトコルを考えてみる．署名付の鍵交換ならば，Alice の出力値 $X = g^x$ に署名を行うことで Alice の出力値を改変するには Alice の秘密鍵が無ければいけないし，Bob の出力値 $Y \leftarrow g^y$ に対しても署名が行われていれば同様に改変されたときに署名の偽造が起きなければ安全である．そこで，以下のようなプロトコルを考えてみる．

1. $\hat{A}$ は署名における署名鍵 Sig_A および検証鍵 Ver_A を生成し，検証鍵を公開しておく．
2. $\hat{B}$ は署名における署名鍵 Sig_B および検証鍵 Ver_B を生成し，検証鍵を公開しておく．
3. $\hat{A}$ は $x \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ を選び，$X \leftarrow g^x$ を計算し，また署名アルゴリズムによって $\sigma_A = \mathsf{Sign}_A(X)$ を生成し，$(\hat{A}, \hat{B}, X, \sigma_A)$ を $\hat{B}$ に送る．
4. $\hat{B}$ は $y \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ を選び，$Y \leftarrow g^y$ を計算し，また署名アルゴリズムによって $\sigma_B = \mathsf{Sign}_B(Y)$ を生成し，$(\hat{B}, \hat{A}, Y, \sigma_B)$ を $\hat{A}$ に送る．
5. $\hat{A}$ は署名が正しいかどうかの検証を行い，検証に通れば session key$SK_A \leftarrow Y^x$ を出力する．
6. $\hat{B}$ は署名が正しいかどうかの検証を行い，検証に通れば session key$SK_B \leftarrow X^y$ を出力する．

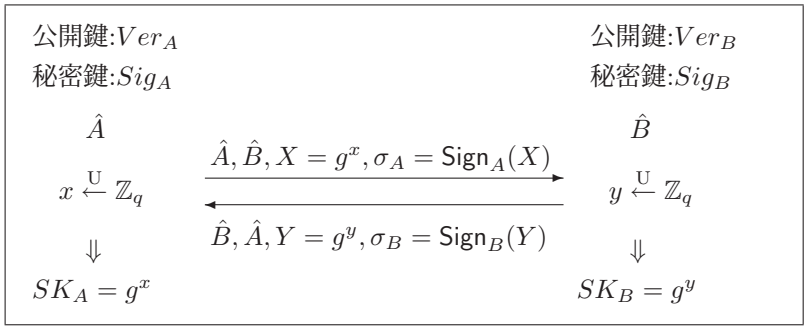


図 32: 署名を利用した DH 鍵交換プロトコル

このようなプロトコルであれば，攻撃者が Alice の秘密鍵 (署名鍵) を得たとしても Bob の署名鍵がない限り Bob に成りすますことができないため，KCI attack に対して安全である．また，session key は毎回選ばれている乱数 x, y によって決まっているため，Known key security も満たしているし，Forward secrecy も満たしている．

しかしながら，近年ではさらなる安全性として一時的に選んでいる秘密鍵に対しての漏洩を考える場合もある．これは，**No Resistance for Leakage of Ephemeral Private Key (RLE)** と呼ばれる安全性の性質であり，session 毎に選ばれている一時的な秘密鍵 x や y の漏洩に関して考えたものである．上記のプロトコルでは x や y が漏洩したとすると，署名に関係なく攻撃者は session key を求めることができるため，RLE が満たされていない．

13.1.5 MQV 鍵交換プロトコル

1995 年に Menezes,Qu,Vanstone によって提案された MQV 鍵交換プロトコルは，これまでに挙げた様々な攻撃が自明な形で効かないような構造をしたプロトコルである．MQV 鍵交換プロトコルは以下のようなプロトコルである．

1. $\hat{A}$ は署名における秘密鍵 $a \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ から公開鍵 $A = g^a$ を公開しておく．
2. $\hat{B}$ は署名における秘密鍵 $b \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ から公開鍵 $B = g^b$ を公開しておく．
3. $\hat{A}$ は $x \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ を選び $X \leftarrow g^x$ を求め，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．
4. $\hat{B}$ は $y \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ を選び $Y \leftarrow g^y$ を求め，$(\hat{B}, \hat{A}, Y)$ を $\hat{A}$ に送る．
5. $\hat{A}$ は X の先頭 ℓbit を d，Y の先頭 ℓbit を e とし，$\sigma_A \leftarrow (YB^e)^{x+ad}$ を計算する．そして，key derivation hash KDF を用いて session key を $SK_A \leftarrow \mathsf{KDF}(\sigma_A)$ と出力する．
6. $\hat{A}$ は $\sigma_B \leftarrow (YA^d)^{y+be}$ を計算し，session key を $SK_B \leftarrow \mathsf{KDF}(\sigma_B)$ と出力する．

13.2 Canetti-Krawczyk Security model

これまでは具体的な鍵交換プロトコルと，それに対する様々な攻撃を示してきた．それでは，鍵交換はどのような安全性を満たすべきであろうか．公開鍵の安全性の定式化のように，鍵交換においても安全性の定式化を行おう．鍵交換の安全性の定式化に関しては 1993 年に Bellare,Rogaway によって行われたものや [4]，2001 年に Canetti,Krawczyk によって行われたもの [8]，2007 年に Lamacchia,Lauter,Mitchagin によって行われたもの [21] 等があるが，今回はまず Canetti,Krawczyk による安全性の定式化を紹介する．

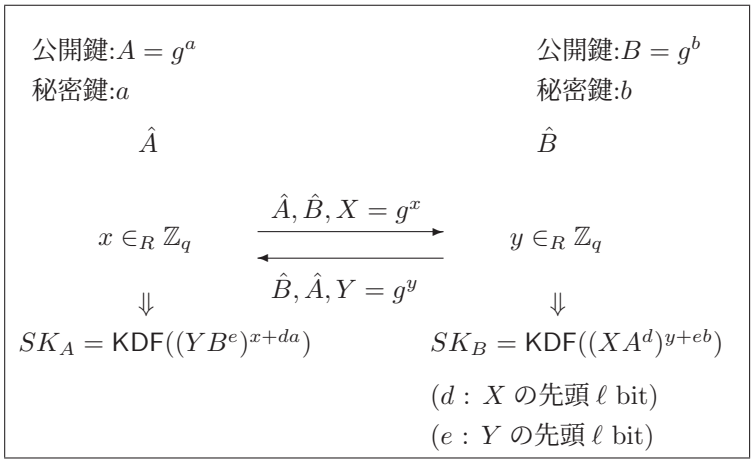


図 33: MQV

13.2.1 パーティと session

鍵交換において最初に行動を起こすパーティを owner，owner の入力によって行動を起こすパーティを peer と呼ぶことにする．このとき，session というのは owner と peer によって値のやり取りを行った後に session key を出力するまでの一連のやり取りのことである．つまり，ある鍵共有プロトコルを動作させた場合，owner である Alice の session と peer である Bob の session の 2 つが構成される．また，鍵交換によって値のやり取りを行ったときに入出力が一致している場合，その 2 つの session は **Matching session** である，という．

13.2.2 攻撃者の能力と安全性の定式化

攻撃者は値の盗聴を行うだけでなく，パーティが鍵交換を行う際の通信路をコントロールする能力を持っているとする．また，先ほどの攻撃などで用いられているようなパーティの内部情報を得られるような以下の 3 つの query を行うことができる．

- State reveal query : ある session においての指定したパーティの内部情報 (session 毎に選ばれる一時的な秘密の値) を得ることができる．
- Session key query : ある session においての指定したパーティの session key を得ることができる．
- Party corrupt query : ある指定したパーティのすべての内部情報を得，完全に操ることができるようになる．

また，鍵共有の安全性を考えるために攻撃者はある session を指定し，Test query という query を 1 度だけ行うものとする．Test query として指定された session (Test session と呼ぶ) の session key を SK とした場合，challenger はコイン投げ $b \xleftarrow{U} \{0,1\}$ を行い，

$$b = 1 \ \Rightarrow \ SK \text{ を攻撃者に与える．}$$
$$b = 0 \ \Rightarrow \ random \text{ な値を攻撃者に与える．}$$

とする．そして，攻撃者に test session に対しての guess b' を出力させ，もし $b' = b$ であれば攻撃者の勝ちとする．ただし，test session として選ばれた session およびその matching session に対しては上記のような攻撃者が内部情報を得られるような 3 つの query は行われていないものとする．攻撃者が 1bit でも test session の session key に関して情報を知っているならこのような guess を当てることができる．逆に，攻撃者の advantage $|\Pr[b' = b] - 1/2|$ が negligible であるならば，その鍵共有プロトコルは安全である，ということができる．

Canetti-Krawczyk security model はいくつかの情報漏えいに関する query を定式化しているが，Canetti-Krawczyk security model において保証しているものは実は攻撃者がパーティの内部情報を得ていない状態での session key の安全性と，Known key security に対しての安全性だけであり，Forward secrecy や KCI attack，session 毎に選ばれる ehemeral な値の漏えいに関しての安全性が捉えられているわけではない．なぜならば，test session および matching session に対しては情報を得るような query を許していないためである．よって，これらの安全性を考える場合は別途考察しなければならない．

13.3 HMQV

HMQV というのは 2005 年に Krawczyk によって提案された鍵交換プロトコルで [20]，MQV 交換プロトコルを基にしている．HMQV は，Canetti-Krawczyk security model を CDH 仮定および random oracle model において満たしていることが示されており，また KCI attack や wPFS と呼ばれる攻撃者が値の改変を行わないとすればパーティの両者の秘密鍵が漏洩したとしても安全性が保たれる，という性質を満たしていることが証明されている．また，session 毎に選ばれる ehemeral な値の漏えいに関しても，Gap Diffie-Hellman 仮定および KEA1 仮定，そして random oracle model において安全性が証明されている．HMQV 鍵交換プロトコルは以下のようなプロトコルである．

1. $\hat{A}$ は署名における秘密鍵 $a \xleftarrow{U} \mathbb{Z}_q$ から公開鍵 $A \leftarrow g^a$ を公開しておく．

2. $\hat{B}$ は署名における秘密鍵 $b \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ から公開鍵 $B \leftarrow g^b$ を公開しておく．

3. $\hat{A}$ は $x \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ を選び $X \leftarrow g^x$ を求め，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．

4. $\hat{B}$ は $y \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_q$ を選び $Y \leftarrow g^y$ を求め，$(\hat{B}, \hat{A}, Y)$ を $\hat{A}$ に送る．

5. $\hat{A}$ はハッシュ関数 H を用いて $d \leftarrow H(X, \hat{B}), e \leftarrow H(Y, \hat{A})$ とし，$\sigma_A \leftarrow (YB^e)^{x+ad}$ を計算する．そして，ハッシュ関数 $\bar{H}$ を用いて session key を $SK_A \leftarrow \bar{H}(\sigma_A)$ と出力する．

6. $\hat{A}$ は $\sigma_B \leftarrow (YA^d)^{y+be}$ を計算し，session key を $SK_B \leftarrow \bar{H}(\sigma_B)$ と出力する．

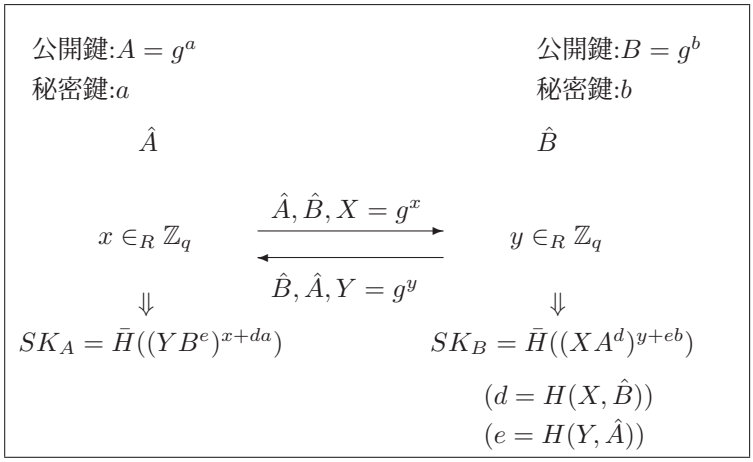


図 34: HMQV

13.4 Extended Canetti-Krawczyk security model

HMQV では高い安全性を示すため，Canetti-Krawczyk security model に対しての証明にさらに付け加えて様々な攻撃に対して安全であることを証明していた．それらの攻撃や安全性の性質をすべて定式化の中で考えたのが LaMacchia,Lauter,Mityagin による security model である [21]．彼らの security model は Canetti-Krawczyk security model の拡張であり，Extended CK (eCK) security model とも呼ばれている．まず，攻撃者が内部情報を得るような query を

- Ephemeral key reveal query : ある session においての指定したパーティの内部情報 (session 毎に選ばれる一時的な秘密の値) を得ることができる．
- Static key reveal query : ある指定したパーティの秘密鍵を得ることができる．

- Session key query : ある session においての指定したパーティの session key を得ることができる.

と言うように分け，さらに Test session やその matching session に対しても上記の query を最大限利用できるように

1. test session に対しては Session key reveal query が行なわれていないこと．また，test session に対しての Ephemeral key reveal query と test session の owner に対しての Static key reveal query が同時には行なわれていないこと.
2. test session に対する matching session が存在する場合，その matching session に対して Session key reveal query が行なわれていこと．また，matching session に対しての Session state reveal query と test session の peer に対しての Static key reveal query が同時には行なわれていないこと.
3. test session に対する matching session が存在しない場合，test session の peer に対しての Static key reveal query が行なわれていないこと.

という制約のみを課すものとしている．eCK security model では，Caneeti-Krawczyk security model では満たされていなかった wPFS や KCI，RLE などの安全性を捉えることができる.

13.5 CMQV

CMQV は HMQV の改良型であり，2007 年に Ustaoglu によって提案された eCK security model を満たしているような鍵交換プロトコルである．CMQV では，session 毎に選ぶ一時的な値を選ぶ際に，秘密鍵とのハッシュを行った後の値を用いる，という作業が追加されることによって eCK security model を満たしている．CMQV 鍵交換プロトコルは以下のようなプロトコルである.

1. $\hat{A}$ は署名における秘密鍵 $a \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ から公開鍵 $A = g^a$ を公開しておく.
2. $\hat{B}$ は署名における秘密鍵 $b \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ から公開鍵 $B = g^b$ を公開しておく.
3. $\hat{A}$ は $\tilde{x} \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ を選び，ハッシュ関数 H_1 を用いて $x \leftarrow H(\tilde{x}, b)$ とする．そして，$X \leftarrow g^x$ を求め，$(\hat{A}, \hat{B}, X)$ を $\hat{B}$ に送る．このとき，x は消去する.
4. $\hat{B}$ は $\tilde{y} \xleftarrow{\mathrm{U}} \mathbb{Z}_q$ を選び，ハッシュ関数 H_1 を用いて $y \leftarrow H(\tilde{y}, b)$ とする．そして，$Y \leftarrow g^y$ を求め，$(\hat{B}, \hat{A}, Y)$ を $\hat{A}$ に送る．このとき，y は消去する.
5. $\hat{A}$ はハッシュ関数 H_2 を用いて $d \leftarrow H_2(X, \hat{B}), e \leftarrow H_2(Y, \hat{A})$ とし，$\sigma_A \leftarrow (YB^e)^{x+ad}$ を計算する．そして，ハッシュ関数 $\bar{H}$ を用いて session key を $SK_A \leftarrow \bar{H}(\sigma_A, X, Y, \hat{A}, \hat{B})$ と出力する.

6. $\hat{B}$ は $\sigma_B \leftarrow (YA^d)^{y+be}$ を計算し，session key を $SK_B \leftarrow \bar{H}(\sigma_B, X, Y, \hat{A}, \hat{B})$ と出力する．

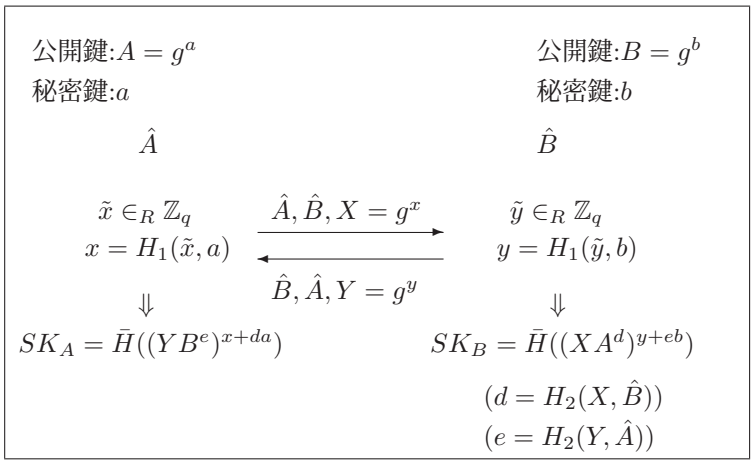


図 35: CMQV

CMQV では，攻撃者が ephemeral key reveal query を行ったときには $\tilde{x}$ や $\tilde{y}$ を返すもの，x や y は消去されているものとして返答しないようにしている．つまりこの場合，攻撃者は session 毎に選ばれている一時的な値を得ることはできるが，出力値である X の指数 x に関しての情報は得られていない．そのため，eCK security model において安全性が満たされていることが証明されている．

参考文献

[1] M. Bellare, A. Desai, D. Pointcheval and P. Rogaway, “Relations Among Notions of Security for Public-Key Encryption Schemes”, *Advances in Cryptology – CRYPTO ’98*, volume 1462 of *LNCS*, pages 26-45, 1998.

[2] M. Ben-Or, S. Goldwasser, A. Wigderson, “Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation(Extended Abstract)”, In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 1-10, 1988.

[3] S. Blake-Wilson, D. Johnson and A. Menezes, “Key exchange protocols and their security analysis”, In *6th IMA International Conference on Cryptography and Coding*, volume 1355 of *LNCS*, pages 30-45, 1997.

[4] M. Bellare and P. Rogaway, "Entity Authentication and Key Distribution", *Advances in Cryptology - CRYPTO '93*, volume 773 of *LNCS*, pages 644-654, 1993.

[5] M. Bellare and P. Rogaway, "Optimal Asymmetric Encryption" *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *LNCS*, pages 92-111, 1994.

[6] M. Bellare and A. Sahai, "Non-malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization", *Advances in Cryptology - CRYPTO '99*, volume 1666 of *LNCS*, pages 519-536, 1999.

[7] R. Canetti, "Universally Composable Security: A New Paradigm for Cryptographic Protocols", In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 136-145, 2001.

[8] R. Canetti and H. Krawczyk, "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels", *Advances in Cryptology - EUROCRYPT '01* volume 2045 of *LNCS*, pages 453-474, 2001.

[9] R. Cramer and V. Shoup, "A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack", *Advances in Cryptology - CRYPTO '98*, volume 1462 of *LNCS*, pages 13-25, 1998.

[10] D. Dolev, C. Dwork and M. Naor, "Non-Malleable Cryptography", In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 542-552, 1991.

[11] W. Diffie and M. Hellman, "New Directions in Cryptography", In *IEEE Transactions on Information Theory*, volume IT-22(6), pages 644-654, 1976.

[12] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", *Advances in Cryptology – CRYPTO '84*, volume 196 of *LNCS*, pages 10-18, 1984.

[13] E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern, " RSA-OAEP Is Secure under the RSA Assumption", In *Journal of Cryptology*, volume 17 number 2, pages 81-104 ,2004.

[14] O. Goldreich, S. Goldwasser and S. Micali, "How to Construct Random Functions (Extended Abstract)", In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science*, pages 464-479, 1984.

[15] S. Goldwasser and S. Micali, "Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information", In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 365-377, 1982.

[16] S. Goldwasser, S. Micali and R. L. Rivest, "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks", In *SIAM Journal on Computing*, volume 17 number 2, pages 281-308, 1988.

[17] O. Goldreich, S. Micali nad A. Wigderson, "How to Prove all NP-Statements in Zero-Knowledge, and a Methodology of Cryptographic Protocol Design", *Advances in Cryptology - CRYPTO '86*, volume 263 of *LNCS*, pages 171-185, 1986.

[18] O. Goldreich, S. Micali nad A. Wigderson, "How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority", In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218-229, 1987.

[19] J. Håstad, R. Impagliazzo, L. A. Levin and M Luby, "A Pseudorandom Generator from any One-way Function", In *SIAM Journal of Computing*, volume 28 number 4, pages 1364-1396, 1999.

[20] H. Krawczyk, "HMQV: A High-Performance Secure Diffie-Hellman Protocol", *Advances in Cryptology - CRYPTO '05*, volume 3621 of *LNCS*, pages 546-566, 2005.

[21] B. A. LaMacchia, K. Lauter and A. Mityagin, "Stronger Security of Authenticated Key Exchange", In *Proceedings of the First International Conference of Provable Security*, volume 4784 of *LNCS*, pages 1-16, 2007.

[22] A. Menezes, P. C. Oorschot, S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.

[23] A. Menezes, M. Qu and S. Vanstone, "Some new key agreement protocols providing implicit authentication", In *2nd Workshop on Selected Areas in Cryptography*, pages 22-32, 1995. 1995.

[24] M. Naor and M. Yung, "Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks", In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 427-437, 1990.

[25] M. O. Rabin, "Digitalized signatures and publickey functions as intractable as factorization", MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[26] C. Rackoff and D. R. Simon, "Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack", In *Proceedings of CRYPTO 1991*, volume 576 of *LNCS*, pages 433-444, 1991.

[27] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", *Communications of the ACM*, volume 21 number 2, pages 120-126, 1978.

[28] B. Ustaoglu, "Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS", In *Designs, Codes and Cryptography*,volume 46 number 3, pages 329-342, 2008.