SCIS 2014 The 31st Symposium on
Cryptography and Information Security
Kagoshima, Japan, Jan. 21 - 24, 2014
The Institute of Electronics,
Information and Communication Engineers

計算論的に完全な記号的攻撃者と鍵交換に関する分析手法
Computationally Complete Symbolic Adversary and Key Exchange

バナ・ゲルゲイ * 長谷部　浩二 † 岡田　光弘 ‡
Gergei Bana Koji Hasebe Mitsuhiro Okada

あらまし セキュリティ・プロトコルの安全性を自動的に検証する方法として，記号検証と呼ばれる方法がある．記号検証は，その安全性が計算論的モデルにおいても成り立つという性質（計算論的健全性）を備えていることが望ましい．近年，こうした計算論的健全性を示す試みが数多くなされている．しかしながら，これまでの研究における計算論的健全性は，ビット列が一意に記号列にパースできることや，暗号文から暗号化に用いられた公開鍵を知ることができること，また鍵のサイクルがないことなどの強い仮定を必要としていた．これに対してバナとコモンは,「計算論的に完全な記号的攻撃者」という概念を定式化した [5, 6]．この記号的攻撃者を用いて示されたプロトコルに関する性質は，先述のような仮定無しで直ちに計算論的モデルの上でも成り立つことが保証される．本研究では，この定式化に対して，さらに鍵の危殆化を表す述語を用いて公理化する．これにより，鍵の送信が許される場合での暗号の安全性を分析することができる．特に CCA2 暗号化を用いるプロトコルにおいて，鍵のサイクルが計算論的安全性を脅かすか否かを，記号的攻撃者の発見もしくは安全性の証明によって示すことができる．なお本稿は，著者らによる CCS ’13 で発表した論文 [7] に基づいている．

キーワード プロトコル, 安全性, 自動検証

1 はじめに

セキュリティ・プロトコルの安全性を自動的に検証する方法として，記号検証と呼ばれる方法がある．記号検証では，メッセージをたとえば「$encrypt(N, K_{ab})$」などの記号列とし，攻撃者はたとえば「$encrypt(N, K_{ab})$ と K_{ab} から N を計算する」のようないくつかのルールのみを用いて攻撃を行うと仮定する．このようなルールによってのみ許される計算能力を備えた攻撃者は，Dolev-Yao 攻撃者と呼ばれる．この手法によるプロトコルの自動検証は多くの成功を収め，これを行うツールもいくつか存在する．

しかし，記号検証によってプロトコルの安全性が示されたとしても，実際のプロトコルの使用においてその安全性が保証されるとは限らない．例えば，Needham-Schroeder-Lowe プロトコル [13] は記号検証で安全とされているが，メッセージのペアの構成が結合法則を満たすとき [14] や，乱数をメッセージのペアと見なせるとき [4, 6] など，計算論的なモデルを想定すると攻撃が成立してしまうことがある．そのため記号検証では，計算論的健全性が成り立つことが望ましい．すなわち，記号検証によって示されたプロトコルの安全性は，計算論的モデルにおいても成り立つというものである．近年，こうした計算論的健全性を示す試みが数多くなされている [3, 2, 11]．しかしその多くは，健全性のために暗号プリミティブとプロトコルの実装に非常に強い仮定を用いている．そのような仮定の例としては，ビット列が一意に記号列にパース（parse）できる，暗号文から暗号化に用いられた公開鍵を知ることができる，鍵のサイクル（秘密鍵に対応する公開鍵による暗号化）がない，攻撃者は参加者をプロトコル実行中にコラプト（corrupt，すなわち正規の参加者が自分の持っている知識を攻撃者に与えること）をしない，などがある．しかも，これらの仮定は計算論的健全性を成り立たせるためにのみ必要なものであって，プロトコルの安全性を成り立たせるために必要であるとは限らない．

計算論的モデルにおいて，こうした強い仮定を排除するため，バナとコモン（以下 “BC”）の研究 [5] では，攻撃者に許される計算をルールとして列挙するのではなく，暗号の安全性などの計算論的仮定から導かれる「攻撃者

* INRIA Paris-Rocquencourt, 23 avenue d’Italie, 75013 Paris, France, bana@math.upenn.edu
† 筑波大学大学院システム情報工学研究科，〒 305-8573 茨城県つくば市天王台 1-1-1, hasebe@cs.tsukuba.ac.jp
‡ 慶應義塾大学文学部哲学科，〒 108-8345 東京都港区三田 2-15-45, mitsu@abelard.flet.keio.ac.jp

が破り得ないこと」を記号的なルールとして列挙し，これと矛盾しないあらゆるメッセージを送信することができる計算論的に完全な記号的攻撃者を考える．

従来法と本研究の違いは次のように説明できる．従来法の攻撃者は，計算論的なモデルにおいて明らかにできることのみが可能なため，一般に計算論的な攻撃者よりも弱い (図 1 左)．このため，健全性を成り立たせるため

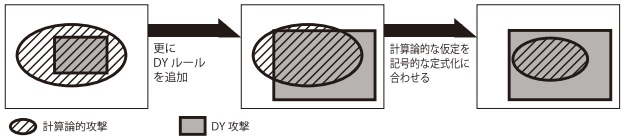

図 1: Dolev-Yao の方法の計算論的健全性

ルールを追加して攻撃者の能力を強める．しかし，ルールを追加しても記号的攻撃者の能力が十分に強くならない（図 1 中央）ため，計算論的な仮定を追加し，計算論的な攻撃を限定する必要がある．(図 1 右)．

一方，BC による研究では計算論的な攻撃よりもはるかに強い攻撃者を考え，それがプロトコルを検証するためには強すぎる場合は必要に応じて公理を追加し，攻撃者の能力を制限する (図 2)．このため，公理を加えると

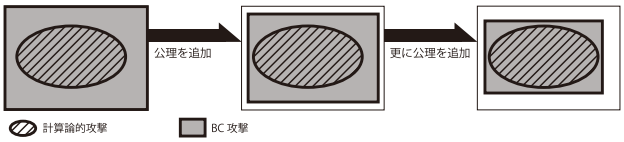

図 2: Bana-Comon の方法の計算論的健全性

きにそれが強すぎる制限でないことを証明しておけば，計算論的モデルを変更することなく計算論的健全性を得ることができる．このような公理を，計算論的な定式化における標準的な仮定（例えば CCA2 安全性）から導き，計算論的モデルの上で成り立つ性質（すなわち，計算論的に健全な性質）としなければならない．これにより，記号的攻撃は，公理と矛盾しないどのような動作も許される．このようにすれば記号的攻撃者は全ての計算論的攻撃者をカバーすることができる．つまり，記号的攻撃者が計算論的に完全であるといえる．あるいは，計算論的に健全な公理のみを含む BC の記号的モデルは計算論的に健全であるといえる．

また，バナ-アダオ-櫻田らの研究 [4] では，計算論的に健全な公理系を与えるとともに，この公理系を用いて Needham-Schroeder-Lowe プロトコルの秘匿性と認証成立を証明し，彼らの方法が実際のプロトコルに適用可能であることを示した．この公理系の特長の一つとして，モジュール化が可能であることが挙げられる．すなわち，整合性の問題を引き起こすこと無く，いくつかの公理の集合を組み合わせて新しい公理系を定義することが可能である．さらに，NSL プロトコルに対する新しい攻撃を発見した．この結果は [6] においても示されている．しかしながら，彼らの公理系は，復号鍵の交換を含むプロトコルを考慮すると，正しく検証することができない．本研究は，この問題の解決を目指すものである．

鍵の交換を伴うプロトコルを正しく扱うために，BC のフレームワークに新たに必要となるのが「鍵の危殆化（key compromise）」の概念である．これは，プロトコル実行時に正しく生成された鍵がある時点においても安全に使用できるか否かを表現するためのものである．もし復号鍵（対称鍵暗号の場合なら，復号鍵は暗号鍵と同一である）がその秘匿性を失って送られると，この復号鍵に対応する暗号鍵は安全に使用できなくなってしまう．あるいは，サイクルがある鍵を送信してしまうと，もしもその暗号スキームが IND-CCA2 安全性のみを仮定していると，同様に安全な暗号化が行われなくなってしまう．こうした鍵の危殆化は，一般にはより複雑で，プロトコルの個々の実行トレースを分析するだけでは判定できない．

本研究では，対称鍵暗号と公開鍵暗号に対して，それぞれ鍵の危殆化を表す述語を導入する．またこれらの危殆化について，IND-CCA2[8] や KDM-CCA2[1, 10]，さらに INT-CTXT[9] での安全性を対象とする．

次に，鍵の危殆化に関する公理を導入し，これらが計算論的に健全であることを示す．なお，本稿で与える公理系もまたモジュール化されていることを指摘しておく．すなわち，既に定義された基本となる公理系に対して新たに公理を加えても，公理系全体の計算論的健全性は保証されるため，再び健全性の証明をやり直す必要はない．実際の検証においては，公開鍵暗号もしくは対称鍵暗号だけを用いるプロトコルを扱うためには基本となる公理系だけで十分であり，また電子署名などを用いるプロトコルの安全性を証明する際には，その証明に必要となる公理のモジュールを加えるだけでよい．こうしたモジュール化が可能であるという性質を用いて，公理を徐々に加えることにより公理系のライブラリを作ることができる．また本稿では，我々の提案する公理系を使って，実際に対称鍵暗号による Needham-Schroeder プロトコルや Otway-Rees プロトコル，また Needham-Schroeder Lowe プロトコルの安全性を示す．

2 記号モデル

ここでは，バナとコモン-ルンド [5] によるフレームワークを用いて記号モデルを定義する．このモデルは次のような考え方に基づく．

プロトコルの実行で交換されるメッセージは，項とよばれる記号列によって表わされる．たとえば項 $\{\langle N_1, A\rangle\}^{R_1}_{eK_B}$ はそれぞれノンス，参加者の名前，公開鍵，乱数を表わす

記号 N_1，A，eK_B，R_1 および，ペアを表わす記号 $\langle \_, \_ \rangle$ と暗号化を表わす記号 $\{\_\}_\_$ からなる．従来のいわゆる Dolev-Yao の記号モデルでは，攻撃者の送るメッセージも項によって表現されるが，BC の手法ではそれと異なり，攻撃者からのメッセージは「ハンドル」と呼ばれる記号一つからなる項で表現される．このハンドルが表す内容は論理式によって記述される．

Dolev-Yao の方法では，たとえば「$\{x\}_K$ および K から x を作れる (復号できる)」ことを表わすルール $\{x\}_K, K \vdash x$ などを考え，ある状態では，攻撃者がこのようなルールを繰り返し適用することによって，それまで受け取ったメッセージの集合 ϕ から作れる項だけを送信できる．攻撃者が送信した項が t であるとき，t が Dolev-Yao ルールを繰り返し適用して ϕ から計算されていることを，$\phi \vdash t$ によって自然に記述することができる．一方，BC の方法では,「攻撃者が項の集合 ϕ からハンドル h を計算することができる」ことを表す述語 $\phi \rhd h$ を考え，$\phi \rhd h$ を示すための Dolev-Yao のようなルールは定義されていない．条件 $\phi \rhd h$ 以外の ϕ と h が満たすべき条件は公理と参加者のチェックの論理式によって記述される．参加者のチェックの論理式はプロトコルのステップごとに決まる．たとえば参加者がハンドル h で表わされるメッセージを受信し，それが以前に送信したノンス N_1 と一致することをチェックするとき，参加者が正しく次の動作に移るならば，$h = N_1$ という性質が成り立つ．ハンドル h はこれらの条件に矛盾しない限りどのような性質を持っていてもよい．また，公理は計算論的な実装における攻撃者の能力をもとに定義される．このような公理の例としては,「正しく生成されたノンスは，攻撃者によって推測されることはない」ことを表す命題などが挙げられる．このため，公理が少ないほど強い攻撃者を考えることになり，公理が少なくても現実に可能な攻撃を網羅できる．

BC のモデルでは，プロトコルの安全性を表す論理式の否定，公理，そして正規参加者のチェックの論理式が互いに矛盾しないとき，プロトコルへの攻撃が成功すると定義する．

2.1 項とフレーム

BC の手法では，参加者が送信するメッセージは項によって表される．項は変数および定数に関数記号を繰り返し適用して得られる．定数には名前とハンドルがある．ここでいう「名前」は応用 π 計算の用語であり，通常の意味の「名前」とは異なる．名前は参加者名だけでなく，参加者が生成した乱数などのデータも含む．通常の意味の名前は「参加者名」あるいは「参加者の名前」と記述して区別する．ハンドルは攻撃者が送信したデータを表す．たとえば項 $\langle N_1, A \rangle$ は名前 N_1 および参加者の名前 A にペアを表わす 2 引数の関数記号 $\langle \_, \_ \rangle$ を適用して得られる．さらに，項 $\{\langle N_1, A \rangle\}^{R_1}_{eK_B}$ は項 $\langle N_1, A \rangle$ および eK_B，および名前 R_1 に暗号化を表す 3 引数の関数記号 $\{\_\}_\_$ を適用して得られる．ここで項 eK_B は名前 K_B に公開鍵を表わす関数記号を適用して得られる．本稿ではこの他，復号を表わす関数記号 $dec(\_, \_)$ を用いる．

フレームは名前の列 $\overline{n}$ と項の列 $t_1, \ldots, t_n$ からなり，$\nu\overline{n}.(t_1, \ldots, t_n)$ で表される．項の列 $t_1, \ldots, t_n$ は正規参加者が送信した項を表し，名前の列 $\overline{n}$ は正規参加者によってランダムに生成されたデータを表す．たとえばフレーム $\nu N.\langle N, A \rangle$ は，攻撃者が送ったメッセージ $\langle N, A \rangle$ を表し，このメッセージは正規参加者が生成したノンス N を用いて作られる．なお，$\nu\overline{n}$ を束縛子と呼ぶ．

2.2 論理式

攻撃者が送信するメッセージはそれまでに受信したメッセージから計算することができるものに限られる．また，正規参加者がこれを受信して次のステップに進むためには，プロトコルで決められた条件を満たさなければならない．このような条件を一階述語論理の論理式によって表す．特に述語記号として等号 $=$，導出可能性 $\_, \ldots, \_ \rhd \_$ および $\hat{\phi}, \_, \ldots, \_ \rhd \_$ を用いる．これらの論理式は記号的実行についても計算論的実行についても用いられる．

論理式の計算論的な意味は第 3 節において詳しく説明するが，直観的には次のような意味である．$t_1, \ldots, t_n \rhd t_{n+1}$ は，項 $t_1, \ldots, t_n$ に対するビット列から項 t_{n+1} に対するビット列を確率的多項式時間アルゴリズムで計算できるという意味である．$\hat{\phi}, t_1, \ldots, t_n \rhd t_{n+1}$ は $t_1, \ldots, t_n$ に加えてそれまでに攻撃者が受けとったメッセージから t_{n+1} を計算できるという意味である．ただし，$\hat{\phi}, \_, \ldots, \_ \rhd \_$ はこれで 1 つの述語記号であり，$\hat{\phi}$ は変数でも具体的なフレームでもない．

BC の手法と Dolev-Yao の方法では，記号的実行に対する述語の意味論が異なる．Dolev-Yao の方法では，述語 $t_1, \ldots, t_n \rhd t_{n+1}$ の記号的意味は項 $t_1, \ldots, t_n$ から t_{n+1} を Dolev-Yao のルールで計算できることである．しかしながら，BC の手法ではこのようなルールを定義しない．この述語の意味は一階述語論理のモデル $\mathcal{M}$ によって定義され，$\mathcal{M}$ は第 5 節の公理と第 2.3 節の正規参加者のチェックを表す論理式を充足すればどのようなモデルでもよい．それぞれのモデルは何らかの攻撃をモデル化している．さらに，$\hat{\phi}, t_1, \ldots, t_n \rhd t_{n+1}$ の意味は記号的実行の各状態に対して定義されている．ある状態での $\hat{\phi}, t_1, \ldots, t_n \rhd t_{n+1}$ の意味は，正規参加者がその状態までに送信したメッセージの列 $u_1, \ldots, u_n$ を $\hat{\phi}$ に代入して得られる $u_1, \ldots, u_n, t_1, \ldots, t_n \rhd t_{n+1}$ と同じである．論理和や限量子などの論理演算子の記号的意味は一階述語論理と同様である。

2.3 プロトコルの実行

記号的な実行および計算論的な実行は応用 π 計算の用語を用いて定義される．攻撃者はネットワークを完全にコントロールすることができる．記号モデルにおけるプロトコルの実行では，攻撃者がメッセージを送信することより参加者を含むネットワークの状態が変化する．ネットワークの状態を次の 4 つ組で表す．

- 制御状態 $q \in Q$ と名前 $n_1, ..., n_k$ の組 $q(n_1, ..., n_k)$
- ハンドルの列 $h_1, \ldots, h_n$
- 基底フレーム ϕ
- 論理式の集合 Θ

フレーム ϕ はこの状態までの実行で正規参加者が送信した項を表し，ハンドル $h_1, \ldots, h_n$ は攻撃者が送信したハンドルである．論理式の集合 Θ はハンドルの性質を表す論理式の集合である．正規参加者はハンドルで表わされるメッセージを受信するとき，その内容をチェックして次のステップに進む．このチェックの内容が集合 Θ に記録される．

例として，次の Needham-Schroeder-Lowe の公開鍵認証プロトコル [13] を取り上げる．

1. $A \to B : \{N_1, A\}_{eK_B}$
2. $B \to A : \{N_1, N_2, B\}_{eK_A}$
3. $A \to B : \{N_2\}_{eK_B}$

このプロトコルの定式化のためにコンストラクタという関数記号の集合 $\mathcal{F}_c = \{\{\_\}^{\_}, \langle \_, \_ \rangle, e_{\_}, d_{\_}, K_{\_}\}$ およびデストラクタという関数記号の集合 $\mathcal{F}_d = \{dec(\_, \_), \pi_1(\_), \pi_2(\_)\}$ を固定する．また，次の等式も仮定する：暗号文の復号は平文と等しい，すなわち $dec(\{x\}^R_{eK}, dK) = x$．ペアと投影に対する等式: $\pi_1(\langle x, y \rangle) = x$，$\pi_2(\langle x, y \rangle) = y$．

ここであるプロトコルのセッションにおいて，B が既に 2 番目のメッセージを送ったとする．また ϕ_2 をフレームとし，Θ_2 をその時点での論理式の集合とする．このとき，次のラウンドで A が最後のメッセージを受け取って返信する手順は，図 3 のように表される．

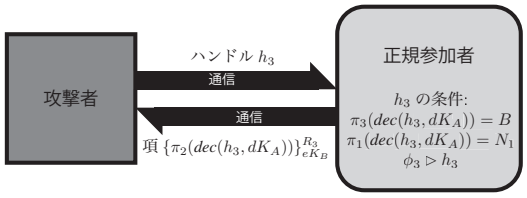

図 3: BC 実行ラウンドの例

次の状態のフレーム ϕ_4 は ϕ_3 に加えて束縛子に R_3 が，項として $\{\pi_1(\pi_2(dec(h_3, dK_A)))\}^{R_3}_{eK_B}$ が追加される．また，論理式の集合は $\Theta_4 = \Theta_3 \cup \{\phi_3 \rhd h_3,$
$\pi_1(dec(h_3, dK_h)) = N_1, \pi_2(\pi_2(dec(h_3, dK_A))) = B\}$ となる．2 つの等式は暗号文を復号したその一部とそれぞれ N_1 および B との比較である．これは参加者 A によるチェックに対応する．(この例に関する詳細については，[6] を参照のこと．)

2.4 制約

これまでに述べた述語記号に加え，次のような制約を用いる．Handle(x) は x がハンドルであるという意味である．RandGen$(x, \hat{\phi})$ は，x がフレーム ϕ の束縛子に出現するという意味である．すなわち，x は x が正規参加者によって生成された乱数である．$x \sqsubseteq \hat{\phi}$ は x がフレームの項に出現するという意味である．すなわち，x は参加者が送ったメッセージの一部である．$x \sqsubseteq \vec{x}$ は x が $\vec{x}$ の一部であるという意味である．$dK \sqsubseteq_{\not d} \hat{\phi}$ は，dK が ϕ の $dec(\_, dK)$ という形以外で出現するという意味である．$dK \sqsubseteq_{\not d} \vec{x}$ も同様である．[1] 同様に，$K \sqsubseteq_{\not ed} \hat{\phi}$ は「対称鍵 K が ϕ において，暗号化もしくは復号化（すなわち，$\{\!|\_|\!\}_K$ もしくは $sdec(\ , K)$）以外の箇所に現れている」ことを表すものとする．また $K \sqsubseteq_{\not ed} \vec{x}$ についても同じように定義する．(このとき K は，$\vec{x}$ における暗号化もしくは復号化で現れてもよいが，他の場所にも現れている必要がある．)

さらに，次のような略記を用いる．

- $x \sqsubseteq \hat{\phi}, \vec{x} \equiv x \sqsubseteq \hat{\phi} \vee x \sqsubseteq \vec{x}$
- fresh$(x; \hat{\phi}, \vec{x}) \equiv$ RandGen$(x, \hat{\phi}) \wedge x \not\sqsubseteq \hat{\phi}, \vec{x}$
- keyfresh$(K; \hat{\phi}, \vec{x})$ 公開鍵の場合：
  keyfresh$(K; \hat{\phi}, \vec{x}) \equiv$ RandGen$(K, \hat{\phi}) \wedge dK \not\sqsubseteq_{\not d} \hat{\phi}, \vec{x}$
- keyfresh$(K; \hat{\phi}, \vec{x})$ 対称鍵の場合：
  keyfresh$(K; \hat{\phi}, \vec{x}) \equiv$ RandGen$(K, \hat{\phi}) \wedge K \not\sqsubseteq_{\not ed} \hat{\phi}, \vec{x}$
- $x \preccurlyeq \hat{\phi}, \vec{x} \equiv \forall h(h \sqsubseteq x \wedge$ Handle$(h) \to \hat{\phi}, \vec{x} \rhd h)$
  $\vec{x} \preccurlyeq \hat{\phi}, \vec{y} \equiv \bigvee_p (x_{p_1} \preccurlyeq \hat{\phi}, \vec{y} \wedge x_{p_2} \preccurlyeq \hat{\phi}, \vec{y}, x_{p_1} \wedge \ldots$
  $\wedge\, x_{p_n} \preccurlyeq \hat{\phi}, \vec{y}, x_{p_1}, ..., x_{p_{n-1}})$

fresh$(x; \hat{\phi}, \vec{x})$ は,「x が正しく生成されているが，フレーム ϕ や $\vec{x}$ には現れていない」ことを表す．また keyfresh$(K; \hat{\phi}, \vec{x})$ は,「鍵が正しく生成されており，復号鍵 dK がフレーム ϕ もしくは $\vec{x}$ において，$dec(\_, dK)$ かつ対称鍵暗号（$K = dK$）の場合に $\{\!|\_|\!\}_K$ という形でしか現れ得ない」ことを表す．さらに，$x \preccurlyeq \hat{\phi}, \vec{x}$ は,「攻撃者による x での入力が $\hat{\phi}, \vec{x}$ から計算できる」ことを表す．

3 計算論的実行とその意味

計算論的実行もまた応用 π 計算の言葉で定義される．攻撃者はネットワークを完全にコントロールすることができるが，その計算能力は確率的多項式時間に限定される．記号的な実行との主な違いは，フレームに加えて参加者名やノンスやハンドルなどの記号に対応するビット列が記録される点である．さらに，記号的実行ではハン

[1] この論文では，乱数 R を入力とする対称鍵暗号と公開鍵暗号の両方について，その暗号化と復号化を表すために，それぞれ $\{x\}^R_{eK}$ と $dec(y, dK)$ を用いる．対称鍵暗号の場合，$eK = dK = K$ となる．また，対称鍵暗号のみについて，その暗号化と復号化を表すために，それぞれ $\{\!|x|\!\}^R_K$ と $sdec(y, K)$ を用いる．

ドル h の記号だけからは対応するメッセージの性質がわからないため，参加者がチェックする論理式を記録してこれを表現したが，計算論的実行ではこれは不要である．

記号的実行では論理式の真偽は実行の各状態において定義されるが，計算論的実行ではそうではない．なぜなら，以下に述べるように述語によっては計算論的実行の全体を確率分布とともに考える必要があるためである．項の計算論的な意味は，実行で用いるランダムネスの空間で定義される多項式時間アルゴリズムである．関数記号は多項式時間アルゴリズムとして解釈され，項 $f(t_1,\ldots,t_n)$ の解釈は項 $t_1,\ldots,t_n$ の解釈に関数記号 f の解釈を適用して得られる．述語記号の解釈は次のように確率的，計算量的に定義される．

等式 $t_1 = t_2$ が充足されるのは，項 t_1 および t_2 の解釈が無視できる確率を除いて等しいときである．より正確には，t_1 および t_2 の解釈を与えるアルゴリズムに，実行で用いたランダムネスを与えると無視できる確率を除いて同じ値が出力されるときである．

導出可能性 $t_1,\ldots,t_n \rhd t_{n+1}$ の解釈はより難しい．直観的には，ある確率的多項式時間アルゴリズム $\mathcal{A}$ が存在し，項 $t_1,\ldots,t_n$ の出力を与えられるときの出力が項 t_{n+1} の出力と一致する，というのがその解釈である．しかし，確率空間全体を 1 つのアルゴリズム $\mathcal{A}$ でカバーできるとは限らない．このため確率空間を無視できない大きさの部分集合に分割し，そのそれぞれで $t_1,\ldots,t_n$ の出力から t_{n+1} の出力を計算するアルゴリズムが存在するとき，導出可能性 $t_1,\ldots,t_n \rhd t_{n+1}$ が真であると定義する．$\hat{\phi}, t_1,\ldots,t_n \rhd t_{n+1}$ についても同様である．ただし，t_{n+1} の計算のために，$t_1,\ldots,t_n$ 以外にも攻撃者はその時点まで見ていたフレーム ϕ に現れるメッセージを使うことができる．

一般に述語の解釈は，その引数の解釈であるアルゴリズムを，無視できる確率を除いて同じ値を出力する他のアルゴリズムで置き換えても変わらないと仮定する．

最後に，論理式の合成は通常の一階述語論理と同じようには定義されないことを強調したい．論理積 ($\wedge$) と全称限量子 ($\forall$) は通常と同じだが，それ以外は異なる．たとえば，論理和 $\theta_1 \vee \theta_2$ が真となるのは，確率空間を θ_1 が真になる空間と θ_2 が真になる空間に分けられるときである．存在限量子の意味も同様に，確率空間をさらに多くの空間に分けることで定義される．論理式 θ の否定 $\neg\theta$ が真となるのは，大きさが無視できないどの部分空間でも θ が真にならないときである．

これらの解釈は一階述語論理の通常の (Tarski 流の) 解釈と異なるが，一階述語論理の推論規則は健全である．実際，この解釈は Fitting による一階述語論理の (一階) 様相論理 S4 への埋め込みと深い関係にある [12].

以上で述べた解釈について，以下の計算論的健全性に関する定理 [5] が成り立つ．

定理 3.1 公理が計算論的に健全であると仮定する．任意のプロトコルについて，プロトコルが同時に有限個しか実行されないと仮定する．このとき，このプロトコルの安全性を無視できない確率で成立不可能にするような計算論的攻撃者が存在すれば，プロトコルの安全性を表す論理式の否定，公理，そして正規参加者のチェックの論理式が互いに矛盾しないような記号的実行が存在する．

いいかえれば，無視できない確率で実行可能な計算論的に成功する攻撃が存在すれば，記号的に成功する攻撃も存在するということである．

4 鍵の危殆化とオラクルの導出可能性

暗号が IND-CCA2 安全であるとき，導出可能性に関する公理の中でも重要なものの一つとして，以下の式に，「K と R が正しく生成され，かつ R 他のものとは独立である」ことを保証するいくつかの仮定を加えたものである．

$$\hat{\phi}, \vec{x}, \{x\}^R_{eK} \rhd y \;\longrightarrow\; dK \sqsubseteq_d \hat{\phi}, \vec{x}, x \;\vee\; \hat{\phi}, \vec{x} \rhd y$$

この公理は，復号化以外で dK が送信されない限り，正しい暗号化である $\{x\}^R_{eK}$ が y の計算に貢献しないことを意味するものである．明らかに，(復号化の) 鍵が交換されることを許すと，この公理は使えなくなってしまう．我々の最初のアイデアは，鍵の危殆化を表す述語として $\blacktriangleright$ を導入し，これにより，秘匿性に関する公理に関して復号化の鍵が交換されないことを仮定するのではなく，鍵の危殆化が起こらないことを仮定するというものであった．その公理は以下のように定式化することが望ましい．

- $\hat{\phi}, \vec{x}, \{x\}^R_{eK} \rhd y \;\longrightarrow\; \hat{\phi}, \vec{x}, x \not\blacktriangleright K \;\vee\; \hat{\phi}, \vec{x} \rhd y$
- $\hat{\phi}, \vec{x}, \{x\}^R_{eK} \blacktriangleright K' \;\longrightarrow\; \hat{\phi}, \vec{x}, x \not\blacktriangleright K \;\vee\; \hat{\phi}, \vec{x} \blacktriangleright K'$
- $\mathsf{keyfresh}(K; \hat{\phi}) \;\longrightarrow\; \hat{\phi} \not\blacktriangleright K.$

ここで，最初の 2 つの公理は，正しい暗号化である $\{x\}^R_{eK}$ が，y の計算に貢献しないか，あるいは K が $\hat{\phi}, \vec{x}, x$ によって危殆化しない限り K' の危殆化に貢献しないことを意味している．また最後の公理は，もしも正しく生成された鍵を復号化する鍵が交換されないならば，K は危殆化しないことを意味している．

変数 x が公理の式の $\hat{\phi}, \vec{x}, x \not\blacktriangleright K$ に現れる理由は，IND-CCA2 安全性を仮定する場合，この部分を $\hat{\phi}, \vec{x} \not\blacktriangleright K$ に置き換えてしまうと十分でないためである．例えば，もし $x = \langle dK, y\rangle$ とすると，暗号化 $\{x\}^R_{eK}$ は鍵のサイクルを含んでしまい，鍵が危殆化するためである．

しかしながら，先に挙げた公理は，$\blacktriangleright$ をどのように解釈しても計算論的健全性が成り立たない．これは，IND-CCA2 安全性の通常の定義での暗号および復号オラクル

の使い方と関係している．だが一方で，もし暗号・復号オラクルの使用を許すような別の導出可能性によって $\triangleright$ を置き換えると，先の公理は鍵の危殆化について計算論的健全性が成り立つ．以上のことから，第 3 節で導入した $\triangleright$ とほぼ同じ計算論的意味論をもつ述語 $\triangleright^{\mathfrak{O}}$ を導入する．ただし $\triangleright^{\mathfrak{O}}$ は，暗号の IND-CCA2 安全性の定義におけるオラクルと同様に，クエリを暗号・復号オラクルに送ることができる点で異なっている．これにより，対称鍵暗号の IND-CCA2，公開鍵暗号の IND-CCA2，対称鍵暗号の KDM-CCA2，公開鍵暗号の KDM-CCA2 を考えた場合に，それぞれオラクルを伴う導出可能性は $\triangleright^{\mathrm{sic2}}$，$\triangleright^{\mathrm{aic2}}$，$\triangleright^{\mathrm{skc2}}$，および $\triangleright^{\mathrm{akc2}}$ で表される．

また，鍵の危殆化を表す述語もオラクルを含むため，$\blacktriangleright^{\mathrm{sic2}}$，$\blacktriangleright^{\mathrm{aic2}}$，$\blacktriangleright^{\mathrm{skc2}}$，および $\blacktriangleright^{\mathrm{akc2}}$ はそれぞれ区別される．

計算論的意味論の基本的なアイデアは以下の通りである．任意の $\mathfrak{O}$ について，$\hat{\phi}, t_1, ..., t_n \blacktriangleright^{\mathfrak{O}} K$ の計算論的意味論は，$t_1, ..., t_n$ とプロトコルの フレーム ϕ に現れる過去のメッセージを用いて，IND（または KDM）-CCA2 ゲームの攻撃者が，K を用いた暗号を破ってゲームで勝つことができることとする．実際の定義はより複雑であり，その詳細は [7] に示している．

これらの定義により，先の公理にいくつかの仮定を加えたものは計算論的に健全となる．次の節で，公理全体を完全に示す．

最後に，第 7 節で示した対称鍵の Needham-Schroeder プロトコルなどの，対称鍵暗号を用いたプロトコルのいくつかについては，CCA2 安全性だけでなく暗号文の完全性（integrity）も必要となる．そのために我々は，INT-CTXT 安全性における鍵の危殆化を表す $\blacktriangleright^{\mathrm{ic}}$ を導入する．これは，IND-CCA2 安全性と同じオラクルを使う．$\hat{\phi}, t_1, ..., t_n \blacktriangleright^{\mathrm{ic}} K$ の計算論的健全性は，本質的には，たとえ攻撃者が $t_1, ..., t_n$ と ϕ に現れる過去のメッセージが与えられたとしても，K を用いたいかなる暗号文も計算できないことを意味している．

5 公理

この節では，我々の公理系の中から重要な公理のみを紹介する．$\triangleright^{\mathfrak{O}}$ と $\blacktriangleright^{\mathfrak{O}}$ については，[6] で示された $\triangleright$ についての単純な公理と類似の公理もある．こうした公理としては，等号の合同の公理，推移律，関数記号の等式などが挙げられる．さらにいくつかの自明な公理として，$\hat{\phi}, \vec{x} \triangleright x \rightarrow \hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} x$ や $\hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} x \rightarrow \hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} x$ がある．これら全ての公理は，[7] で示されている．

• 復号オラクルの有用性を表す公理．

$$\hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} y \wedge \forall x R(y = \{x\}^R_{eK} \rightarrow \{x\}^R_{eK} \not\sqsubseteq \hat{\phi}, \vec{x}) \wedge \mathsf{RandGen}(K, \hat{\phi}) \longrightarrow \hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} dec(y, dK).$$

この公理は，もしも y が計算可能でかつ $\hat{\phi}, \vec{x}$ に現れる暗号文でないならば，$dec(y, dK)$ もまた復号化のオラクルを呼び出すことができるため，同じものから計算することができることを意味している．

• 「危殆化していない鍵は安全に暗号化する」ことを表す公理．

– もし $\mathfrak{O}$ が aic2 か sic2 であるならば，

$$\mathsf{RandGen}(K, \hat{\phi}) \wedge \mathsf{fresh}(R; \hat{\phi}, \vec{x}, x, y, K) \wedge \vec{x}, x, y \preccurlyeq \hat{\phi} \wedge \hat{\phi}, \vec{x}, \{x\}^R_{eK} \triangleright^{\mathfrak{O}} y \longrightarrow \hat{\phi}, \vec{x}, x \blacktriangleright^{\mathfrak{O}} K \vee \hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} y$$

は計算論的に健全である．この公理は，もし鍵が危殆化していないならば（すなわち $\hat{\phi}, \vec{x}, x \not\blacktriangleright^{\mathfrak{O}} K$ ならば），$\{x\}^R_{eK}$ は y を導出する上で役に立たないことを意味している．つまり，もしも $\{x\}^R_{eK}$ を用いて y を導出できれば，それ無しでも y を導出することができることを意味している．実際に，Freshness と乱数による生成に関する条件は，$\{x\}^R_{eK}$ が安全な暗号化であること（例えば，$\{N\}^N_{eK}$ や $\{N\}^{eK}_{eK}$ は安全ではない）や y は $\{x\}^R_{eK}$ に依存することができないこと（例えば、$y = \{x\}^R_{eK}$ できない）を保証している．さらに，$\vec{x}, x, y \preccurlyeq \hat{\phi}$ は，この項のハンドルが攻撃者によって計算可能な値であることを保証する．

– もし $\mathfrak{O}$ が akc2 か skc2 ならば，

$$\mathsf{RandGen}(K, \hat{\phi}) \wedge \mathsf{fresh}(R; \hat{\phi}, \vec{x}, x, y, K) \wedge \vec{x}, x, y \preccurlyeq \hat{\phi} \wedge \hat{\phi}, \vec{x}, \{x\}^R_{eK} \triangleright^{\mathfrak{O}} y \longrightarrow \hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} K \vee \hat{\phi}, \vec{x} \triangleright^{\mathfrak{O}} y$$

は計算論的に健全である．この公理と IND-CCA2 安全性に関する公理との違いは，この公理では $\hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} K$ に x が現れないことである．詳細については第 6 節で述べる．

• 「危殆化していない鍵で暗号化された暗号文は他の鍵を危殆化させない」ことを表す公理．

– IND-CCA2 の場合：$\mathfrak{O}$ が aic2 か sic2 のとき，

$$\mathsf{RandGen}(K, \hat{\phi}) \wedge \mathsf{RandGen}(K', \hat{\phi}) \wedge \mathsf{fresh}(R; \hat{\phi}, \vec{x}, x, K, K') \wedge \vec{x}, x \preccurlyeq \hat{\phi} \wedge \hat{\phi}, \vec{x}, \{x\}^R_{eK'} \blacktriangleright^{\mathfrak{O}} K \longrightarrow \hat{\phi}, \vec{x}, x \blacktriangleright^{\mathfrak{O}} K' \vee \hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} K$$

は計算論的に健全である．すなわち，もしも $\hat{\phi}, \vec{x}, \{x\}^R_{eK'}$ が K を危殆化するならば，K が $\{x\}^R_{eK'}$ 無しで既に危殆化しているか，あるいは K' が $\hat{\phi}, \vec{x}, x$ によって既に危殆化しているかのいずれかであることを意味している．

– KDM-CCA2 の場合：もし $\mathfrak{O}$ が akc2 か skc2 ならば，

$$\mathsf{RandGen}(K, \hat{\phi}) \wedge \mathsf{RandGen}(K', \hat{\phi}) \wedge \mathsf{fresh}(R; \hat{\phi}, \vec{x}, x, K, K') \wedge \vec{x}, x \preccurlyeq \hat{\phi} \wedge \hat{\phi}, \vec{x}, \{x\}^R_{eK'} \blacktriangleright^{\mathfrak{O}} K \longrightarrow \hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} K' \vee \hat{\phi}, \vec{x} \blacktriangleright^{\mathfrak{O}} K$$

は計算論的に健全である．この公理は，先の IND-CCA2 の場合と基本的に同じである．ただし，$\hat{\phi}, \vec{x} \blacktriangleright K'$ は x を含まない点で異なっている．

• Fresh な生成に関する公理．

–「Keyresh な鍵は危殆化しない」ことを表す公理：この公理の直観的な意味は，もし K が keyfresh であるならば，それは安全な暗号化のために利用できる（すなわち，$\mathsf{keyfresh}(K;\hat{\phi},\vec{x}) \wedge\ \vec{x} \preccurlyeq \hat{\phi}\ \longrightarrow\ \hat{\phi},\vec{x} \not\blacktriangleright^{\mathfrak{O}} K$）というものである．もし K が（正しく）生成されかつ暗号化アルゴリズムが CCA2 または INT-CTXT 安全であるならば，この公理に関して健全性が成り立つ．公理に含まれる $\mathfrak{O}$ が aic2 か sic2 か akc2 か skc2 か ic かによって，この暗号化が，それに対応するレベルの安全性を持つ必要がある．なおこうした暗号化の安全性を必要とするのは，この公理だけである．

–「Fresh なものは鍵を危殆化させない」ことを表す公理：この公理は，互いに独立に生成されかつ送信されていないものは，他のものを危殆化させることがないということを意味している．すなわち，$\mathsf{fresh}(x;\hat{\phi},\vec{x},K) \wedge \vec{x},K \preccurlyeq \hat{\phi} \wedge \hat{\phi},\vec{x},x \blacktriangleright^{\mathfrak{O}} K \rightarrow \hat{\phi},\vec{x} \blacktriangleright^{\mathfrak{O}} K$ である．

- **「危殆化していない鍵による暗号文は偽造できない」ことを表す公理．**

$$\mathsf{RandGen}(K,\hat{\phi})\ \wedge\ \hat{\phi},\vec{x} \rhd y\ \wedge\ dec(y,dK) \neq \bot$$
$$\wedge\ \forall x R(y = \{x\}^{R}_{eK} \rightarrow \{x\}^{R}_{eK} \not\sqsubseteq \hat{\phi},\vec{x})\ \longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\mathrm{ic}} K$$

この公理は，「もし鍵 K が危殆化していないならば，意味のあるものに復号化するような y を，攻撃者は計算できない」ということを表している．これは，INT-CTXT 安全性で要求される性質，すなわち暗号文を偽造できないことを表している．この公理に関する健全性は，我々の意味論から直接示すことができるものであり，INT-CTXT 安全性は必要ではない．

6 矛盾の証明の例

この節では，先の公理系との矛盾をどのように導出するかについて，いくつかの簡単な例を示す．論文 [4, 6] の中で，最も単純な例がいくつか示されている．したがって，本稿ではもう少し複雑なものとして鍵の送信を含む場合について扱う．また，ここで示される例では対称鍵を用いている．

例 6.1 あるフレームでの最初のメッセージを

$$\phi_3 \equiv \langle (A,B), \{\!|K|\!\}^{R_1}_{K_{AB}}, \{\!|h_2,N|\!\}^{R_2}_{K} \rangle$$

とし，その名前を $KK_{AB}NR_1R_2$ とする．また，対称鍵は IND-（または KDM-）CCA2 安全であるとする．ここで示すことは，$\phi_3 \rhd N$ が公理系と矛盾すること，すなわち N が秘匿性を保つことである．$\mathfrak{O}$ は sic2 もしくは skc2 のいずれかを表すものとする．また，$\phi_3 \rhd N$ が成り立つと仮定する．よって $\phi_3 \rhd^{\mathfrak{O}} N$ が成り立つ．このことはまた $\phi_2, \{\!|h_2,N|\!\}^{R_2}_{K} \rhd^{\mathfrak{O}} N$ と同値である．推測不能性の公理から，$\mathsf{fresh}(N;\phi_2)$ によって $\phi_2 \not\rhd^{\mathfrak{O}} N$ が成り立つ．対称鍵の IND-CCA2 安全性についての「危殆化していない鍵が安全に暗号化する」ことを表す公理から，$\phi_2, \{\!|h_2,N|\!\}^{R_2}_{K} \rhd^{\mathfrak{O}} N$ を仮定しているので，（公理に現れる $\vec{x}$ と x と y に対して，$\langle\rangle$ と $\langle h_2,N\rangle$ と N をこの順序で代入すると）同様に，$\phi_2 \rhd^{\mathfrak{O}} N$ か（$\mathfrak{O} \equiv$ sic2 か $\mathfrak{O} \equiv$ skc2 かによって）$\phi_2,h_2,N \blacktriangleright^{\mathrm{sic2}} K$ か，$\phi_2 \blacktriangleright^{\mathrm{skc2}} K$ のいずれかが成り立つ．$\phi_2 \not\rhd^{\mathfrak{O}} N$ は既に示されている．よって，$\phi_2,h_2,N \blacktriangleright^{\mathrm{sic2}} K$ について考える．IND-CCA2 の場合，「fresh であれば鍵を危殆化させない」ことを表す公理から，N が ϕ_2 に現れないので，$\phi_2,h_2 \blacktriangleright^{\mathrm{sic2}} K$ が成り立つ．ハンドルは常にこのフレームから導出されるので，$\phi_2 \rhd h_2$，よって，$\phi_2 \rhd^{\mathrm{sic2}} h_2$ と，推移律を $\phi_2,h_2 \blacktriangleright^{\mathrm{sic2}} K$ と $\phi_2 \rhd^{\mathrm{sic2}} h_2$ に適用した結果から，$\phi_2 \blacktriangleright^{\mathrm{sic2}} K$ が成り立つ．以上は KDM の場合についてであるが，同様に（IND と KDM の両方について）$\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}} \not\blacktriangleright^{\mathfrak{O}} K$ が成り立つ．「危殆化していない鍵が安全に暗号化する」ことを表す公理から，（公理に現れる K' と $\vec{x}$ と x に対して，K_{AB} と $\langle\rangle$ と K をこの順序で代入すると）$\phi_1 \not\blacktriangleright^{\mathfrak{O}} K$ か $\phi_1,K \blacktriangleright^{\mathrm{sic2}} K_{AB}$，もしくは $\phi_1 \blacktriangleright^{\mathrm{skc2}} K_{AB}$ が得られる．しかし，「keyfresh な鍵は危殆化していない」ことを表す公理から，$\phi_1 \not\blacktriangleright^{\mathfrak{O}} K$ である．また同様に $\phi_1 \not\blacktriangleright^{\mathfrak{O}} K_{AB}$ である．よって，KDM の場合については矛盾が示された．IND の場合についても同様に，「fresh であれば鍵を危殆化させない」ことを表す公理 $\phi_1,K \blacktriangleright^{\mathrm{sic2}} K_{AB}$ と $\mathsf{fresh}(K;\phi_1,K_{AB})$ から，$\phi_1 \blacktriangleright^{\mathrm{sic2}} K_{AB}$ が示され，公理との矛盾が得られた．

例 6.2 まず，$\phi_3 \equiv \langle (A,B), \{\!|K|\!\}^{R_1}_{K_{AB}}, \{\!|K_{AB},h_2,N|\!\}^{R_2}_{K} \rangle$ を仮定する．以下で $\phi_3 \rhd^{\mathfrak{O}} N$ が公理と矛盾することを示す．この例では，K と K_{AB} が互いに暗号化し合う鍵のサイクルが存在することに注意が必要である．よって $\phi_3 \rhd^{\mathfrak{O}} N$ を仮定する．IND-CCA2 安全性から，「危殆化していない鍵が安全に暗号化する」ことを表す公理から，先の例 6.1 の議論を用いて $\phi_2,K_{AB},h_2,N \blacktriangleright^{\mathrm{sic2}} K$ を得ることができる．よって，先の例と同様，h_2 と N を取り除くことができる．また $\phi_2 \equiv \phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}$ であることから，$\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\mathrm{sic2}} K$ を得る．しかし，このことから矛盾を導くことはできない．関数記号の等式の公理から，$K = sdec(\{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB})$，また関数の導出可能性を表す公理から，$\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \rhd^{\mathrm{sic2}} K$ が成り立つ．よって，$\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\mathrm{sic2}} K$ が得られ，そこから「導出可能性から鍵の危殆化が成り立つ」ことを表す公理を用いても，矛盾を導くことはできない．しかしながら，もし KDM 安全を仮定すると，先の例と同様，「危殆化していない鍵は安全に暗号化する」ことを表す公理によって，$\phi_3 \rhd^{\mathrm{skc2}} N$ から直ちに $\phi_2 \blacktriangleright^{\mathrm{skc2}} K$ が導かれ，また残りの議論についても同様である．よってこの場合，$\phi_3 \rhd^{\mathrm{sic2}} N$ は公理と整合的であるが，$\phi_3 \rhd^{\mathrm{skc2}} N$ とは矛盾することが示される．

例 6.3 $\phi_3 \equiv \langle (A,B), \{|K|\}^{R_1}_{K_{AB}}, \{|\{|K_{AB}|\}^{R_2}_{K'}, h_2, N|\}^{R_3}_{K} \rangle$ と名前 $KK'K_{AB}NR_1R_2,R_3$ を考える．より厳密には，K と K_{AB} の間にはサイクルが存在するが，K' によって互いに影響し合わない．ここで再び IND-CCA2 安全性を仮定すると，先の例 6.1 と同様,「危殆化していない鍵は安全に暗号化する」ことを表す公理を用いて，$\phi_3 \rhd^{\mathrm{sic2}} N$ からまず $\phi_2, \{|K_{AB}|\}^{R_2}_{K'}, h_2, N \blacktriangleright^{\mathrm{sic2}} K$ が成り立つ．先の例 6.1 と同様，h_2 と N を取り除くことによって $\phi_2, \{|K_{AB}|\}^{R_2}_{K'} \blacktriangleright^{\mathrm{sic2}} K$ が導かれる．このとき,「危殆化していない鍵で暗号化された暗号文は他の鍵を危殆化させない」ことを表す公理によって，$\phi_2 \blacktriangleright^{\mathrm{sic2}} K$ か $\phi_2, K_{AB} \blacktriangleright^{\mathrm{sic2}} K'$ が成り立つ．前者の場合，先の例 6.1 と同じ議論によって矛盾を導くことができる．また後者の場合については，関数（暗号化）を $\phi_1, \{|K|\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\mathrm{sic2}} K'$ に適用することにより，$\phi_1, K, K_{AB}, R_1 \blacktriangleright^{\mathrm{sic2}} K'$ を得ることができる．しかし,「fresh なものは鍵を危殆化させない」ことを表す公理によって，全ての K, K_{AB}, R_1 を取り除くことができる．よって $\phi_1 \blacktriangleright^{\mathrm{sic2}} K'$ が,「fresh な鍵は危殆化していない」ことを表す公理と矛盾することがわかる．

7 対称鍵の Needham-Schroeder プロトコル

これまで説明してきた公理系を用いて，以下のような対称鍵の Needham-Schroeder プロトコルの修正版の安全性を証明することができる．

1. $A \to B : A$
2. $B \to A : \{A, N_1\}_{K_{BT}}$
3. $A \to T : \langle A, B, N_2, \{A, N_1\}_{K_{BT}} \rangle$
4. $T \to A : \{N_2, B, K, \{K, N_1, A\}_{K_{BT}}\}_{K_{AT}}$
5. $A \to B : \{K, N_1, A\}_{K_{BT}}$
6. $B \to A : \{N_3\}_K$
7. $A \to B : \{N_3 - 1\}_K$

このプロトコルは，最初に鍵の交換を行った上で，その鍵を用いてノンスの安全な暗号化を行うというものである．このプロトコルに対する記号的攻撃が存在しないこと（すなわち計算論的健全性により計算論的攻撃も存在しないこと）は，以下のように示すことができる．IND-CCA2 と INT-CTXT の公理を用いて，まず帰納法により，信頼できる第三者（T）から A および B に対して送られてきた鍵 K が危殆化しないことが示される．(もし危殆化するなら，公理と参加者のチェックに対して矛盾するため.）次に，再び帰納法を用いて，N_3 が漏洩しないことが示される．最後に，プロトコルの合意（agreement）と認証成立が示される．

実際の証明では，先に示した公理以外にも，$+1$ と -1 とが互いに逆関数であることを表す公理と，$x - 1 \neq x$ という公理が必要である．また，正しく生成されたノンスに対して，ペアの 1 番目の要素を抽出する操作を適用して得られたものがノンス自身になる確率は無視できるほど小さい，という性質も必要となる．また，3 重対や 4 重対はペアの組み合わせで構成されるものとする．

この証明では，参加者 A がイニシエータとして，また参加者 B がレスポンダとしてプロトコルを実行しており，また信頼できる第三者は 1 つだけ存在すると仮定している．ただし，これらの参加者は任意の数のセッションを並行して実行することができ，またそれぞれ正規のもしくはコラプトされた参加者とのプロトコルの実行が起こりうると仮定している．

以上で述べた証明の詳細は，著者らのホームページ上に掲載している．

8 結論

本稿では，バナとコモンの研究 [5] によって導入された計算論的に完全な記号的攻撃者のフレームワークをさらに拡張した．これにより，鍵の交換を伴うプロトコルが扱いうることができることを説明した．また，我々の公理系では，ビット列が一意に記号列にパースできることなどといった先行研究で導入されていた仮定を使わずに，安全性証明が行えることを示した．さらに，IND-CCA2 や KDM-CCA2，また INT-CTXT などの安全性について，計算論的健全性が成り立つような公理の集合のモジュール化についても述べた．最後に，いくつかの単純な例を用いて我々の公理系の使い方について説明するとともに，プロトコル全体を対象とした検証の能力を示した．

今後の課題としては，セッション数に制約が無い場合や，識別不可能性に関する性質を扱いうる，より一般的かつ健全なフレームワークへと拡張することが挙げられる．またここで示した理論的成果をもとに，自動検証ツールの開発へと発展させたいと考えている．

参考文献

[1] P. Adão, G. Bana, J. Herzog, and A. Scedrov. Soundness and completeness of formal encryption: the cases of key-cycles and partial information leakage. *Journal of Computer Security*, 17(5):737–797. IOS Press, 2009.

[2] M. Backes, D. Hofheinz, and D. Unruh. CoSP: A general framework for computational soundness proofs. In *CCS'09*, 66–78. ACM, 2009.

[3] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *CCS'03*, 220–230. ACM, 2003.

[4] G. Bana, P. Adão, and H. Sakurada. Computationally Comlete Symbolic Attacker in Action. In *FSTTCS 2012*, 546–560. Dagstuhl, 2012.

[5] G. Bana and H. Comon-Lundh. Towards unconditional soundness: Computationally complete symbolic attacker. In *POST'12*, 189–208. Springer, 2012.

[6] バナ・ゲルゲイ，ウベール・コモン，櫻田 英樹．計算論的に完全な記号的攻撃者とセキュリティ・プロトコルの 計算論的に健全な検証． 2013 年暗号と情報セキュリティシンポジウム予稿集，CD-ROM (4D1-3).

[7] G. Bana, K. Hasebe and M. Okada. Computationally Complete Symbolic Attacker and Key Exchange In *CCS'13*, 1231–1246. ACM, 2013.

[8] M. Bellare, A. Desai, D. Pointcheval, and Ph. Rogaway. Relations among notions of security for public-key encryption schemes. In *CRYPTO'98*, 26-45. 1998.

[9] M. Bellare and Ch. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology*, 21(4):469–491. Springer, 2008.

[10] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In *EUROCRYPT'09*, 351–368. Springer, 2009.

[11] H. Comon-Lundh and V. Cortier. Computational soundness of observational equivalence. In *CCS'08*, 109–118. ACM, 2008.

[12] M. Fitting. An embedding of classical logic in S4. *The Journal of Symbolic Logic*, 35(4):529–534. 1970.

[13] G. Lowe. Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR. In *TACAS'96*, 147–166. Springer, 1996.

[14] J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *CCS'01*, 166–175. ACM, 2001.