特集 フォーマルメソッドの新潮流 7

フォーマルメソッドによる暗号安全性

萩谷昌己 ● 東京大学大学院情報理工学系研究科 コンピュータ科学専攻

近年世界的に研究が活性化している，フォーマルメソッドと暗号理論との境界領域について解説する．

はじめに

近年，情報セキュリティの向上のためにフォーマルメソッドへの期待が高まっている．特に，暗号系を用いるセキュリティ・プロトコルの検証にフォーマルメソッドを応用しようとする試みが盛んである．当初は，フォーマルメソッドに典型的な記号的検証が主であったが，近年では，暗号系の脆弱さも考慮に入れた本格的な解析や検証に，フォーマルメソッドを応用しようとする試みが盛んになってきている．

まず具体的な例をあげよう．いうまでもなく，SSL（Secure Sockets Layer）は Web 上の暗号化と認証機能のために標準的に用いられているセキュリティ・プロトコルである[1]．現在は SSL3.0 が広く用いられているが，それ以前のバージョンである SSL2.0 には多くの攻撃方法が知られていた．その中でも，ciphersuite rollback attack と呼ばれる攻撃は，プロトコルの設計における根本的な過ちによるものである．

以下は，次章で参照する Mitchell たちの論文にある SSL2.0 のハンドシェイク・プロトコル（図 -1）の記述である[2]．

ClientHello	$C \to S$	$C, Suite_C, N_C$
ServerHello	$S \to C$	$Suite_S, N_S, \mathrm{sign}_{CA}\{S, K_S^+\}$
ClientMasterKey	$C \to S$	$\{Secret_C\}_{K_S^+}$
（合意された暗号に変更）		
ClientFinish	$C \to S$	$\{N_S\}_{\mathrm{Master}\,(Secret_C)}$
ServerVerify	$S \to C$	$\{N_C\}_{\mathrm{Master}\,(Secret_C)}$
ServerFinish	$S \to C$	$\{SessionId\}_{\mathrm{Master}(Secret_C)}$

このようなプロトコルの記述方法は，一般に（この例ではアリスもボブも出てこないが）アリス・ボブ記法と呼ばれている．C はクライアント，S はサーバを表す．

最初の行は ClientHello という種類のメッセージの送受信を示している．送信者はクライアント C であり，

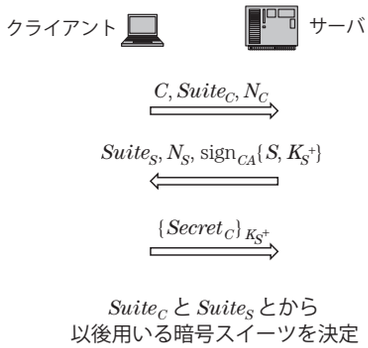

図 -1　SSL2.0 のハンドシェイク・プロトコル

受信者はサーバ S である．右端の $C, Suite_C, N_C$ が送受信されるメッセージを表す，このメッセージは，C と $Suite_C$ と N_C という 3 つの部分から成り立っており，これらの平文を単純に連結したもので暗号などは用いられていない．C はクライアントの ID を表す．N_C はノンスと呼ばれる乱数で，このメッセージを送る際にクライアント C が新たに生成した乱数である．

$Suite_C$ は，プロトコルの以降の部分における暗号化に際してクライアントが用いたい暗号スイーツ（暗号手続きの一揃い）を指定している．同様に，この次のメッセージでは，サーバ S が自分の用いたい暗号スイーツ $Suite_S$ を指定している．このようにして，クライアントとサーバは，自分の用いたい暗号スイーツをお互いに教え合い，その上で実際に用いる暗号（合意された暗号）を決定する．

2 番目の行は ServerHello という種類のメッセージを示し，サーバ S がクライアント C に $Suite_S, N_S, \mathrm{sign}_{CA}\{S, K_S^+\}$ というメッセージを送信する．N_S はサーバ S が生成したノンス，$Suite_S$ は上述したようにサーバ S が用いたい暗号スイーツである．$\mathrm{sign}_{CA}\{S, K_S^+\}$ は，サーバの ID である S とその公開鍵 K_S^+ に認証局（certificate authority）の署名を施したメッセージを表す．

3番目の行はClientMasterKeyという種類のメッセージを示し，クライアントCがサーバSに$\{Secret_C\}_{K_S^+}$というメッセージを送信する．$Secret_C$はハンドシェイク以後の通信に用いるマスタ鍵を生成するための乱数であり，クライアントCがこのメッセージを送信する際に生成する．$\{Secret_C\}_{K_S^+}$は，$Secret_C$をサーバSの公開鍵K_S^+によって暗号化した結果を表す．

以上のプロトコルには致命的な欠陥がある．$Suite_C$と$Suite_S$が暗号化されずに平文の形で送られているにもかかわらず，その内容を後に両者が確認することを怠っている．このために，典型的な中間者攻撃（man-in-the-middle attack）が可能である．すなわち，以下のようにして，攻撃者が指定した暗号スイーツ$Suite_I$によって以後の通信が行われてしまう．このような攻撃をciphersuite rollback attackという（図-2）．

$C \rightarrow S(I)$　$C, Suite_C, N_C$
$I \rightarrow S$　$C, Suite_I, N_C$
$S \rightarrow C(I)$　$Suite_S, N_S, \mathrm{sign}_{CA}\{S, K_S^+\}$
$I \rightarrow C$　$Suite_I, N_S, \mathrm{sign}_{CA}\{S, K_S^+\}$
$C \rightarrow S$　$\{Secret_C\}_{K_S^+}$
（$Suite_I$に合意）

ここで，$S(I)$はサーバへのメッセージを攻撃者が横取りすることを意味する．$C(I)$も同様である．

上の攻撃はプロトコルの根本的な過ちによるものであり，用いる暗号系には依存しない．この攻撃において実際に用いられている暗号操作は，最後に$Secret_C$を送る際の公開鍵K_S^+による暗号化のみであるが，ここでどのように強い暗号系を用いたとしても防ぐことはできない．すなわち，決して破ることのできない理想的な暗号系を仮定したとしても可能な攻撃である．

次章で詳述するが，フォーマルメソッドを用いて以上のような攻撃を検出する研究が，1990年代の終わりから2000年代の初めにかけて盛んに行われていた．モデル検査や定理証明など，フォーマルメソッドの分野で長年にわたって培われてきたさまざまな技術を用いて，与えられたプロトコルに対する攻撃を探索したり，攻撃が存在しないことを形式的に検証することが可能となった．逆に，このような攻撃が存在しないことが証明されたプロトコルは，理想的な暗号系を仮定すれば安全性が保証される．なお，決して破ることのできない理想的な暗号系を仮定したプロトコルの実行モデルは，古くからDolev-Yaoモデルと呼ばれている．

SSL2.0に対する同様の攻撃としては，パディングされるデータに対してMACハッシュが適用されているにもかかわらず，パディングの長さにはハッシュが適用されていないことによるものもよく知られている．

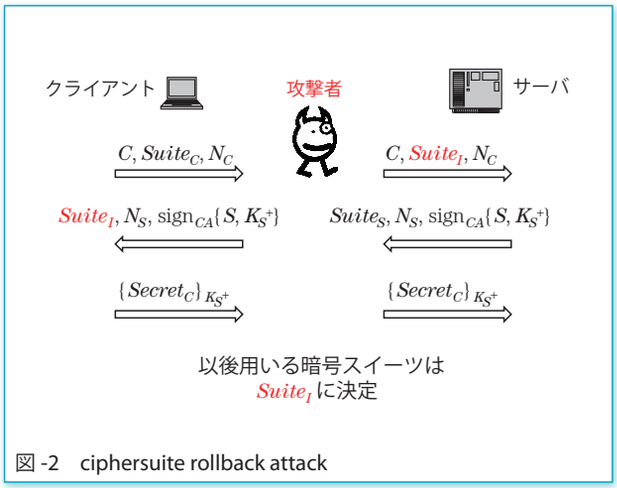

図-2　ciphersuite rollback attack

なお，次章で詳述するように，以上で述べたようなDolev-Yaoモデルにおいても生じる問題，特に中間者攻撃に関する問題は，SSL3.0ではすでに解消されている．

しかしながら，プロトコルの安全性は，実際に用いられる暗号系に大きく依存する．上述のciphersuite rollback attackにおいても，この攻撃自体に成功した後には，$Suite_I$に依存した攻撃を行わなければならない．

暗号系に依存する攻撃としては，PKCS#1（RSA Encryption Standard）の暗号化ブロックのフォーマットを利用したmillion message attackが有名である[3]．PKCS#1の暗号化ブロックは

00 | 02 | padding string | 00 | data block

というフォーマットをしている．Bleichenbacherは，暗号文が与えられたとき，さらに自由に選んだ暗号文がこのフォーマットに適合しているかどうかを知ることができれば，与えられた暗号文を解読できることを示した．このように，与えられた暗号文以外に自由に選んだ暗号文を解読してくれる手続き（オラクル）を仮定した攻撃は，適応的選択暗号文攻撃と呼ばれている．ただし，この攻撃のオラクルは，攻撃者が選んだ暗号文を解読するのではなく，ある一定のフォーマットに適合しているかどうかを判断するだけでよい．

上述のSSL2.0では，サーバの公開鍵で暗号化されたメッセージ$\{Secret_C\}_{K_S^+}$がクライアントからサーバに送られる．これはSSL3.0でも同様であり，しかもSSL3.0ではバージョン番号が含まれているので，

00 | 02 | padding string | 00 | 03 | 00 | premastersecret

というさらに限定されたフォーマットになっているため，より効率的に暗号文の解読を行うことができる．実際に，攻撃者は$\{Secret_C\}_{K_S^+}$を横取りした後，自由に選んだ暗号文で$\{Secret_C\}_{K_S^+}$を置き換えてサーバに送信する．サーバは送られたメッセージがフォーマットに適合しなけ

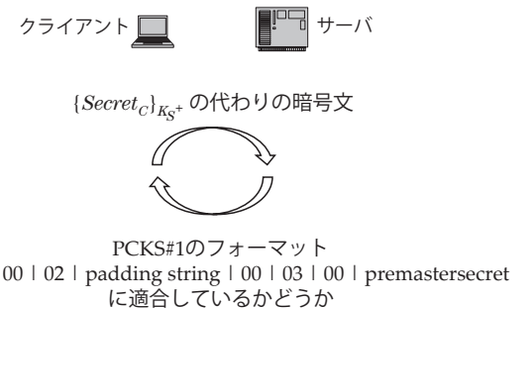

図 -3　million message attack

ればエラーを返す．同様の送信を繰り返すことにより，次第に $Secret_C$ の内容が明らかになっていく（図 -3）．このようにして，攻撃者はプロトコルに巧みに介入して参加者を利用することにより，暗号系に対する適応的選択暗号文攻撃を行うことができる．したがって，より安全な PKCS#1v2.0 を利用することが推奨されている．なお，$\{Secret_C\}_{K_S^+}$ は $Suite_I$ に合意させられる前に行われる暗号化なので，この適応的選択暗号文攻撃は先の中間者攻撃とは独立の攻撃である．

同様に，暗号系の脆弱性によりプロトコルが脆弱になるという事例としては，MD5 の脆弱性を利用して APOP プロトコルでパスワードを特定するという攻撃が知られている．

次々章で詳しく述べるが，2000 年に発表された Abadi と Rogaway の論文が契機となって，上述のような暗号系に依存した攻撃の解析に対してもフォーマルメソッドを適用しようとする一連の研究が始まった [4]．現在でも多くの研究者によって精力的に研究が続けられており，暗号系の研究者や技術者も注目するところとなっている [5]．

本稿の以下の部分では，まず次章で Dolev-Yao モデルにおける記号的解析について簡単に述べる．次に次々章でフォーマルメソッドと暗号理論との境界領域の研究について紹介する．おわりに将来展望について簡単に述べる．

記号的解析

Dolev-Yao モデルに基づいてプロトコルの実行過程を形式的に定義することができれば，Dolev-Yao モデルにおける攻撃の可能性を厳密に解析し，攻撃が存在しない場合は存在しないことを形式的に検証することも可能になる．Dolev-Yao モデルのもとでは，メッセージをビット列ではなく項として表すことができるので，このような解析は記号的（symbolic）であるという．なお，項によって表現されたメッセージは，Dolev-Yao 項と呼ばれる．

最も一般的に，プロトコルの正規の参加者と攻撃者の全体から成るシステムを状態遷移系として定義すれば，状態遷移系に対するさまざまな検証手法を適用することができる．特にモデル検査は，状態空間を網羅的に探索することによって状態遷移系の検証を行う技術であり，セキュリティ・プロトコルの検証に対してもモデル検査の技術が活発に適用されてきた．

正規の参加者はプロトコルの記述に従ってメッセージの送受信を行う．攻撃者は，ネットワーク上のメッセージをすべて受信することができるが，Dolev-Yao モデルのもとでは暗号系は理想的であると仮定しているので，暗号化されたメッセージは秘密鍵を知らなければ解読することはできない．もちろん，そのまま転送することは可能である．また，メッセージを改竄したり偽造したりすることも可能であるが，署名鍵を知らなければ署名を偽造することはできない．

以上のような参加者と攻撃者は，プロセスとして捉えることが自然であろう．これらのプロセスと，ネットワーク上に流れるメッセージのバッファから状態遷移系が構成される．攻撃者は，あらゆる種類の攻撃を行うことができると仮定しているので，きわめて非決定的なプロセスとして定式化されるだろう．

実際に，CSP（communicating sequential process）や π 計算などのプロセス代数を用いてプロトコルの記述を行い，プロセス代数に対するモデル検査系を用いてプロトコルの解析や検証が行われてきた．また，SPIN や Mur φ のように，通常のプログラミング言語に近い形式でプロセスを記述することのできるモデル検査系も用いられている．たとえば，Mitchell たちは Mur φ を用いて SSL3.0 に関する詳細な解析と検証を行っている [2]．

先に述べたように，SSL2.0 では ciphersuite rollback attack が可能であった．これを防ぐために，SSL3.0 ではハンドシェイクの過程でやりとりしたメッセージをすべてひとまとめにした情報（以下の *Messages*）を，クライアントとサーバがお互いに確認し合う．以下は，SSL3.0 そのものではないが，SSL2.0 の問題点を解消したプロトコルである．

ClientHello	$C \to S$	$C, Ver_C, Suite_C, N_C$
ServerHello	$S \to C$	$Ver_S, Suite_S, N_S, \mathrm{sign}_{CA}\{S, K_S^+\}$
ClientMasterKey	$C \to S$	$\mathrm{sign}_{CA}\{C,V_C\}, \{Secret_C\}_{K_S^+}$, $\mathrm{sign}_C\{\mathrm{Hash}(Messages)\}$
（合意された暗号に変更）		
ClientFinish	$C \to S$	$\{\mathrm{Hash}(Messages)\}_{\mathrm{Master}(Secret_C)}$
ServerVerify	$S \to C$	$\{\mathrm{Hash}(Messages)\}_{\mathrm{Master}(Secret_C)}$

詳細は省くが，このように修正したプロトコルにおいても，ある種の cryptosuite attack が可能であることが報

告されている[2]．

モデル検査は基本的に有限状態の遷移系にしか適用できないので，たとえばプロトコルの複数のセッションが同時に実行される際の問題点を解析したい場合には，セッションの数を適当に限定しなければならない．このような制限はあるものの，システムの状態空間の大きさが許容範囲内にあれば，モデル検査系による解析と検証は完全に自動的に行うことが可能であり，攻撃が存在すれば具体的な攻撃方法を出力してくれるし，状態空間を網羅して攻撃が存在しなければその旨が報告される．

モデル検査系と比較して，定理証明系を用いた形式的検証は，完全に自動的に行うことは不可能であるが，状態空間が無限の場合にも適用できる．たとえばセッションの数を限定せずに検証を行うことができる．実際にPaulsonは，SSL3.0に類似したプロトコルであるTLS（Transport Layer Security）の検証を，Isabelleという定理証明系を用いて行っている[6]．Isabelleは対話的な定理証明系であり，タクティクと呼ばれる自動証明手続きを人間が適切に選んで実行することを繰り返しながら，目標の定理を形式的に検証することができる．

最後に，もう1つの記号的解析の手法として，ストランド空間の理論について触れたい[7]．この理論では，プロトコルの実行がストランドの集合として捉えられる．ストランドは特定の参加者もしくは攻撃者のメッセージ送受信の履歴を表し，ストランド内のメッセージの受信は，別のストランド内の送信と関連付けられる（図-4）．このようなストランドの概念を中心に据えて，プロトコルの正当性に関する議論を行う．たとえば，ある参加者のストランドが存在したときに，必ず別の参加者のストランドが対応する，ということが示されれば，プロトコルの認証性が証明される．

ストランド空間の理論は，プロトコルの検証を行うためにちょうどよい抽象度を提供していると考えられる．人手で検証を行うのに適しているとともに自動化も可能である．実際に，ストランド空間に基づく自動証明系が作られている．

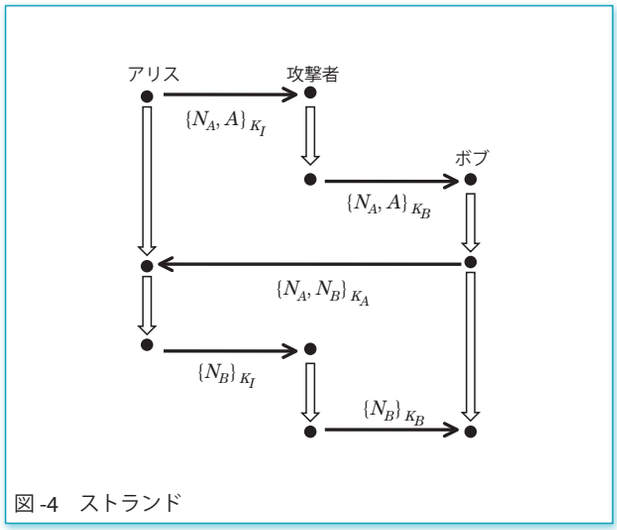

図-4　ストランド

記号的＋計算論的

「はじめに」の最後の方で紹介したように，暗号系の脆弱性も認めつつプロトコルの形式的な解析を行うためには，まず暗号系の脆弱性を厳密に定式化しなければならない．暗号理論においては，確率多項式時間チューリングマシン（PPT ― probabilistic polynomial-time Turing machine）という枠組みが用いられる．すなわち，プロトコルの参加者や攻撃者はPPTとして定式化され，PPT同士はそのテープを介して通信を行う．PPTは確率的な動きをするチューリング機械であり，実行時間がセキュリティ・パラメタの多項式に制限されている．なぜなら，いくらでも計算に時間をかけてよければ，任意の暗号は解読できてしまうからである．セキュリティ・パラメタとは，鍵の長さのように，セキュリティの強さを定めるパラメタのことである．なお，確率的な動作をさせるためには，あらかじめランダムな記号が無限に書かれたテープが用意されていて，チューリング機械はそのテープを読みながら動くと考えてもよい．

暗号系の強さは，このようなPPTを用いた各種のゲームによって定式化される．たとえば，「はじめに」のPKCS#1の場合，ゲームの元締めが暗号文を提示し，攻撃者が自由に暗号文を選んでフォーマットに適合しているかどうかを元締めに問い合わせ，最後に攻撃者が与えられた暗号文を解読することができれば勝ち，というゲームによって定式化される．PKCS#1の脆弱性は，このゲームに十分に高い確率で勝てる攻撃者をPPTとして定義できることを意味している．

「はじめに」の例の場合，プロトコルの攻撃者は，正規の参加者と巧みに通信を行うことにより，暗号系に対する適応的選択暗号文攻撃を行う．結果として，秘密情報を盗んだり正規の参加者を騙したりする．このような攻撃者は暗号系の攻撃者を手続きとして含み，全体はPPTとして定義される．

逆に，暗号系が十分に強く作られていて，所定のゲームに勝つような攻撃者が存在しないことが仮定されているとき，プロトコルの安全性を示すためには，次のような議論を行えばよい．すなわち，プロトコルに対する攻撃が可能であるとき，プロトコルの攻撃者をもとにして，暗号系の攻撃者を構成できることを示す．すると，暗号系の攻撃者は存在しないはずだったので，プロトコルの攻撃者も存在しないことが示される．これがプロトコルの計算論的（computational）な安全性である．

実際に，公開鍵暗号系の強さを定義するために，IND-CCA2 と呼ばれるゲームが広く用いられている（図 -5）．攻撃者はゲームの元締めに 2 つのメッセージ m_0 と m_1 を送る．元締めはそのどちらかを暗号化したメッセージ c を攻撃者に返す．攻撃者は，c 以外の暗号文を自由に選んで元締め（オラクル）に復号してもらうことができる．最後に，m_0 と m_1 のどちらを暗号化したかを攻撃者が当てることができれば勝ち．

PKCS#1 のゲームの元締めよりも，IND-CCA2 のゲームの元締めが提供するオラクルの機能が高いので，PKCS#1 のゲームの攻撃者は IND-CCA2 ゲームにも勝てる．したがって，いうまでもなく，PKCS#1 は IND-CCA2 の強さのセキュリティを持っていないことになる．RSA ベースの公開鍵暗号系として IND-CCA2 の強さのセキュリティを持つものとしては，RSA-OAEP や RSA-KEM などがある．

本章の以下の部分では，以上のような PPT をベースにした計算論的な議論を，フォーマルメソッドの枠組みで行うための 2 種類の方法について解説する．

●間接的方法

間接的方法とは，前章の記号的解析の結果を利用して計算論的な正しさを導く方法である．特に，ゲームによって定式化された暗号系の強さを仮定して，プロトコルの計算論的な検証を行うために用いられる．

先に参照した論文で Abadi と Rogaway は，攻撃者が受動的である場合，すなわち，攻撃者がメッセージの傍受のみを行い改竄や偽造は行わない場合に，記号的に情報が漏えいしないことが示されれば，計算論的にも情報が漏えいしない，すなわち，秘密情報を取得するような PPT の攻撃者は存在しない，ということを示した [4]．

より具体的に，Abadi と Rogaway は，2 つのメッセージ（Dolev-Yao 項）m_0 と m_1 が記号的に等価であれば，受け取ったビット列が m_0 から作られたものか，m_1 から作られたものかを，無視できるほど小さい確率を除いて，PPT を用いる限りは識別できない，ということを示した（ここで，「無視できるほど小さい確率を除いて識別できない」とは，セキュリティ・パラメタ η の任意の多項式 $p(\eta)$ に対して，識別できる確率が $1/p(\eta)$ より小さいことを意味する）．

以上の成果が契機となって，これまでにさまざまな間接的手法が提案されている．たとえば，Herzog は Abadi と Rogaway の結果を，IND-CCA2 を満たす公開鍵暗号を含むメッセージに拡張している [8]．しかも，ビット列を識別する際に，メッセージを作るときに用いられた公開鍵に対する復号オラクルを使ってもよい．すなわち，たとえ攻撃者が復号オラクルを使ったとしても，

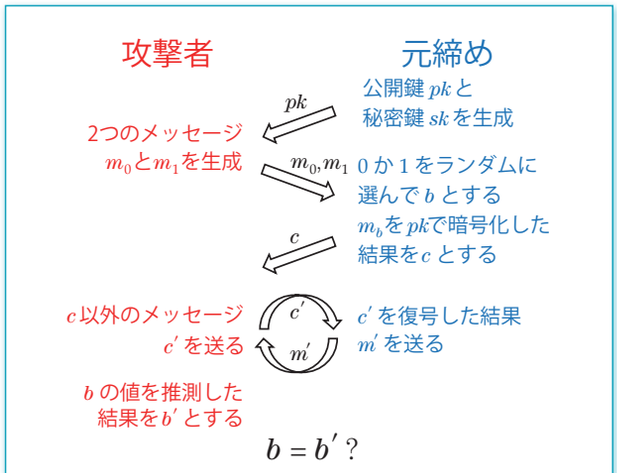

図 -5　IND-CCA2 ゲーム

ビット列を識別できる確率が無視できるほど小さい．したがって，復号オラクルがあっても大丈夫なので，たとえば先の PKCS#1 の million message attack の場合においても，ビット列の識別はできないことが分かる．すなわち，能動的な攻撃者が参加者をうまく利用するような場合でも，ビット列の識別ができないことを意味している．

次の段階として，より本格的に能動的な攻撃者，たとえば成り済ましをするような攻撃者を想定する場合においても，記号的解析によってプロトコルに対する攻撃が存在しないことから，計算論的にもプロトコルに対する攻撃が存在しなことを示したい．このためにはどうすればよいだろうか．まず，プロトコルへの計算論的な攻撃者，すなわち，PPT の攻撃者が存在したと仮定する．記号的解析によってプロトコルに対する攻撃が存在しないと分かっているので，この攻撃者の動きは記号的にはあり得ない．すると，計算論的な攻撃者はどこかで Dolev-Yao モデルを破っているはずである．この破れをうまく利用することができれば，暗号系の攻撃者を構成できるだろう．これは暗号系の強さの仮定に反する．

もちろん，Dolev-Yao モデルの破れから暗号系の攻撃者を構成することは容易ではない．しかし，一度このような議論を一般的に行っておけば，計算論的な攻撃者も Dolev-Yao モデルで許容される攻撃しかできないことが保証される．このような保証はマッピング補題（mapping lemma）と呼ばれている．より厳密にいうと，計算論的な攻撃者は，無視できるほど小さい確率を除いて，Dolev-Yao モデルで許容される攻撃しかできない．以上のようにして，能動的な攻撃者に対するマッピング補題を最初に示したのは，Micciancio と Warinschi である [9]．

本稿では紙数の都合で詳しく述べることができないが，セキュリティ・プロトコルの計算論的な正しさを定義す

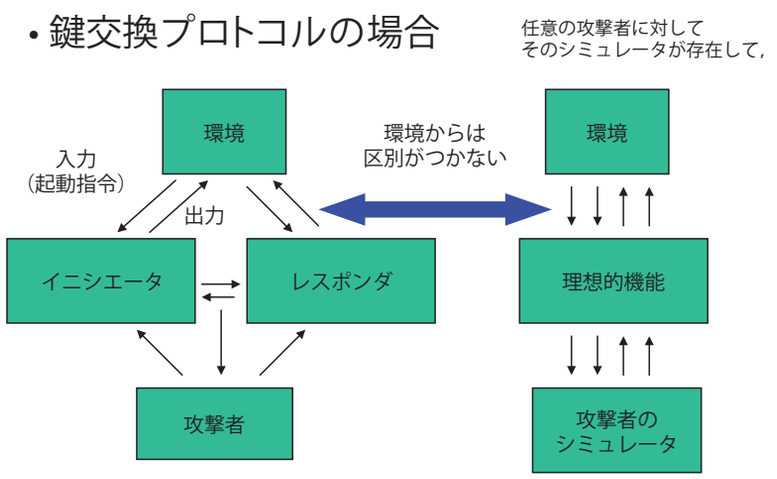

図 -6（a）汎用的結合可能性

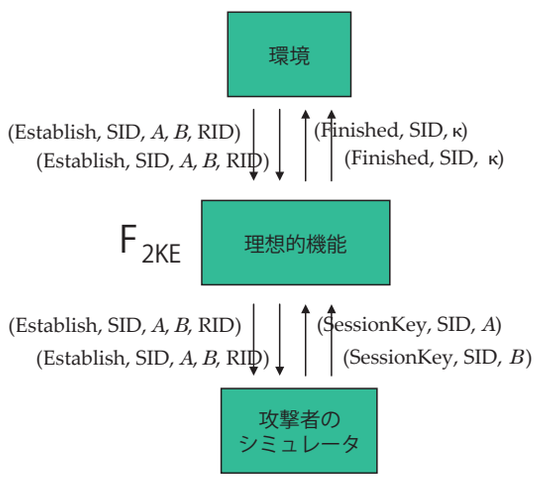

図 -6（b）鍵交換の理想的機能

る方法として，近年，汎用的結合可能性と呼ばれる概念が脚光を浴びている．この方法は，プロトコルが満たすべきあらゆる条件を理想的機能として定義し，プロトコルの実装と理想的機能が，実行環境から見て等価であることを示すことにより，プロトコルの絶対的な正しさを示す(図 -6)．最近になって，汎用的結合可能性を示すための間接的方法も提案されている [10]．

●直接的方法

直接的方法とは，暗号理論の分野で普通に行われている議論を忠実に，しかし形式的に，ときには自動的に，運用する方法である．現在のところ，主としてゲーム変換の形式化が試みられている．特に，ゲーム変換を用いて，IND-CCA2 などの暗号系の計算論的な強さを，より基本的なプリミティブの計算論的な強さから示すことが目標となっている．

ゲーム変換とは，変換前のゲームに攻撃者が勝つ確率の上限と，変換後のゲームに攻撃者が勝つ確率の上限とを関連付けながら，ゲームを少しずつ変換していくことにより，最終的にもとのゲームに勝てる攻撃者が存在しないことを示す証明方法である．暗号系の強さもプリミティブの強さもゲームによって定義されているので，ゲーム変換により暗号系のゲームとプリミティブのゲームが関連付けられれば，暗号系の強さを示すことができる．

ゲーム変換は，暗号理論の分野で頻繁に行われている議論を整理したもので，Shoup のチュートリアルに分かりやすく紹介されている [11]．直接的方法とは，このようなゲーム変換をフォーマルメソッドの技術を用いて形式化し，さらに自動化しようとする試みにほかならない．

確率ホーア論理は直接的方法の 1 つで，ゲーム変換で行われる確率的な議論を，ホーア論理を拡張した形式的体系のもとで行う [12]．また，プロセス代数に確率概念を導入し，プロセスの書き換え系によって，ある程度自動的にゲーム変換を行う方法もある [13]．実際にゲーム

変換を自動的に行うツールも開発されており，次第に広く用いられるようになっている．

おわりに

暗号理論の分野では，セキュリティ・プロトコルの安全性を証明によって厳密に保証すべきだという機運が高まっている．たとえば，暗号技術の安全性を厳密に証明しよう，という流れは電子情報通信学会誌「小特集：暗号技術の証明可能性安全性」にもまとめられている[14]．

しかし，暗号に関する議論はますます煩雑になっており，それに伴って厳密な議論は難しくなっている．そこで，フォーマルメソッドに対する期待は高まるばかりである．フォーマルメソッドの具体的な効用はいくつか考えられる．

- 人が証明を記述するための規範となる方法論や言語を提供する．
- 証明を(部分的に)自動化する．
- 汎用的結合可能性のようなきわめて複雑な概念を整理し再構築する．

今後のさらなる発展に期待したい．

謝辞　NTT コミュニケーション科学基礎研究所の櫻田英樹氏と塚田恭章氏に感謝します．

参考文献

1) Wagner, D. and Schneier, B. : Analysis of the SSL 3.0 Protocol, Proceedings of the Second USENIX Workshop on Electronic Commerce,Vol.2 (1996).
2) Mitchell, J., Shmatikov, V. and Stern, U. : Finite-State Analysis of SSL 3.0, USENIX Security '98 (1998). http://verify.stanford.edu/uli/secur/usenix/usenix.html
3) Bleichenbacher, D. : Chosen Ciphertext Attacks against Protocols Based on RSA Encryption Standard PKCS #1, Advances in Cryptology-CRYPTO '98, LNCS, Vol.1462, pp.1-12 (1998).
4) Abadi, M. and Rogaway, P. : Reconciling Two Views of Cryptography (the Computational Soundness of Formal Encryption), Journal of Cryptology, Vol.15, No.2, pp.103-127 (2002).
5) 萩谷昌己：数理的技法による情報セキュリティの検証，応用数理，Vol.17, No.4, pp.8-15 (2007).
6) Paulson, L. C. : Inductive Analysis of the Internet Protocol TLS, ACM Transactions on Information and System Security, Vol.2, No.3, pp.332-351 (1999).
7) Thayer, F. J., Herzog, J. C. and Guttman, J. D. : Strand Spaces : Proving Security Protocols Correct, Journal of Computer Security, Vol.7, No.2/3, pp.191-230 (1999).
8) Herzog, J. : A Computational Interpretation of Dolev-Yao Adversaries, Theoretical Computer Science, Vol.340, No.1, pp.57-81 (2005).
9) Micciancio, D. and Warinschi, B. : Soundness of Formal Encryption in the Presence of Active Adversaries, TCC 2004, LNCS, Vol.2951, pp.133-151 (2004).
10) Canetti, R. and Herzog, J. : Universally Composable Symbolic Analysis of Mutual Authentication and Key-Exchange Protocols, TCC 2006, LNCS, Vol.3876, pp.380-403 (2006).
11) Shoup, V. : Sequences of Games : A Tool for Taming Complexity in Security Proofs, http://eprint.iacr.org/2004/332 (2004).
12) Corin, R. and den Hartog, J. : A Probabilistic Hoare-style Logic for Game-Based Cryptographic Proofs, ICALP 2006, LNCS, Vol.4052, pp.252-263 (2006).
13) Blanchet, B. and Pointcheval, D. : Automated Security Proofs with Sequences of Games, CRYPTO 2006, LNCS, Vol.4117, pp.537-554 (2006).
14) 岡本　健他：小特集：暗号技術の証明可能性安全性，電子情報通信学会誌，2007 年 6 月号.

(平成 20 年 3 月 21 日受付)

萩谷昌己(正会員)
hagiya@is.s.u-tokyo.ac.jp
1982 年東京大学大学院理学系研究科情報科学専攻修士課程修了．京都大学数理解析研究所を経て，現在，東京大学大学院情報理工学系研究科コンピュータ科学専攻教授．