社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－DPS－122 (28)
2005－CSEC－ 28 (28)
2005／3／22

# 安全性を保証する構造要件に基づく セキュリティプロトコルの自動生成法

佐藤 直人　萩原 茂樹　米崎 直樹

東京工業大学 大学院 情報理工学研究科 計算工学専攻

近年のインターネット上での重要な取引の増加に伴い、安全なプロトコルが必要不可欠となっているが、検証による安全性保証の限界を考えると、安全なプロトコルを自動生成することが望ましい。本稿では二者間データ通信のためのセキュリティプロトコル自動生成法を提案する。これは、プロトコルの構造から安全性を導出する推論規則を利用し、安全なプロトコルのみを生成する手法である。また、プロトコルに対するコスト評価を行うことで、使用環境に適したより良いプロトコルを生成する。さらに、他の自動生成方式と比較することで、本稿で提案する自動生成法の利点についても議論する。

# Automatic Generation of Security Protocols by Verified Constructions

Naoto Sato　Shigeki Hagihara　Naoki Yonezaki

Department of Computer Science, Graduate School of Information Science and Engineering,
Tokyo Institute of Technology.

As the Internet has grown rapidly, it becomes a significant issue to design a security protocol to transmit personal or commercial data securely. Although many formal methods for verifying security protocol have been suggested, automatic generation of security protocol is more promising than finding secure protocol by verification. In this paper, we propose a method for automatic generation of security protocols for transimitting data between two principals. We give inferrence rules which deduce security properties from structure of a protocol. By using these rules, we can generate a protocol suitable for real environment, introducing cost evaluation of a protocol. Finally we discuss advantage of our method compared with other generation methods.

## 1　はじめに

個人情報や商取引に関する重要データがネットワークを通じて伝達されている。これらの通信上のセキュリティを守るためには、個々のメッセージに対する暗号化技術が必須であるが、それだけでは十分ではない。暗号化技術が完全であっても、伝送手順であるプロトコルに欠陥があればメッセージのすり替えや成り済まし攻撃を受ける可能性がある。よって、プロトコルの安全性を形式的に検証する技術が必要とされる。

しかし、検証による安全性保証の計算量的限界を考えると、これまでは非形式的に行われてきたプロトコルの設計を形式的に行うアプローチの方が現実的である。

そこで本稿では、二者間でデータ通信を行うためのセキュリティプロトコルの自動生成法を提案する。この生成法では安全性が保証されているメッセージ構造を元に、安全なプロトコルのみを生成する。また、プロトコルに対する静的なコスト評価を行うことで、使用環境に適したセキュリティプロトコルを選別する。

以下ではまず前提条件を定義した後、プロトコルの構造から安全性を推論するための推論規則を導入する。そしてこれを利用した自動生成手続きを示す。さらにこの推論規則の健全性をストランド空間 [4] における意味に基づいて証明する。最後に、他の自動生成法との比較も行う。

## 2　準備

まず、本稿で自動生成の対象としているプロトコルの条件や、想定する攻撃者の振る舞い、プロトコルが満たすべきセキュリティプロパティ等を定義していく。

### 2.1　生成対象プロトコル

- 2者間でデータをやり取りするためのプロトコルを生成する。データとは、ナンスや参加者の名前とは異なるメッセージで、これをお互いに通達することをプロトコルの目的とする。
- 参加者は受信したデータを返送することはない。
- データは必ず交互にやり取りする。
- 1つの送受信メッセージに対して、同じ鍵で二度以上の暗号化、署名操作は行わない。
- ナンスは必ずセッションに対応付けて使用する。(自分の作成したナンスとセッションとの対応を、相手に認識させる)
- レスポンダは、ステップ1のメッセージからイニシエータの名前を知ることが出来る。
- 正当な参加者がセッションごとに作成するナンスは新規なものであり、以前に使用されたものを使用することはない。(ナンスの新規性)
- 参加者は一度に複数のセッションで通信を行うことが出来る。

### 2.2　攻撃者のモデル

- 攻撃者は、通信される全てのメッセージを盗聴することが出来る。
- 攻撃者は、通信される全てのメッセージを他のメッセージとすり替えることが出来る。
- 攻撃者は対象とするネットワークにおいて、参加者として通信を行うことが出来る。
- 攻撃者は全ての参加者の公開鍵を知っている。
- セッション開始前に、攻撃者がそのセッション参加者の秘密鍵を知っていることはない。
- 攻撃者は、元々知っている知識や攻撃によって得た知識を元にメッセージを偽造し、正しいメッセージとすり替えることが出来る。

### 2.3　セキュリティプロパティ

生成するプロトコルは以下の3つのセキュリティプロパティを満たすものとする。

**データの秘密性**　ある特定のデータについて、データ作成者の意図した相手以外にそのデータは知られることのない性質。

**データ認証**　受信したデータが意図した相手からのものであることを受信者が確認できる性質。

**受信確認**　セッションで送信した全てのデータが、意図した相手に正しく受信されたことを送信者が確認できる性質。

### 2.4　プロトコルの仕様

生成するプロトコルの仕様は以下の項目からなる。これらの仕様は自動生成器に対する入力として与えられる。

- やり取りするデータの数。

- 秘密にしたいデータ（何番目のデータを秘密にするか）。
- イニシエータとレスポンダのどちらからデータを送信し始めるか。

## 3 生成に利用する論理

本節ではプロトコルの自動生成に用いる論理を与える。この論理を用いることで、前節で挙げたセキュリティプロパティの検証を行うことが出来る。

### 3.1 用語

論理で用いる変数や定数、その他の用語を定義する。

$A, N_A$ ：それぞれイニシエータの名前とイニシエータの作成したナンスを表す定数

$B, N_B$ ：それぞれレスポンダの名前とレスポンダの作成したナンスを表す定数

$P_i, N_{P_i}$ ：それぞれステップ $i$ での送信者の名前と、その送信者が作成したナンスを表す変数

$Q_i, N_{Q_i}$ ：それぞれステップ $i$ での受信者の名前と、その受信者が作成したナンスを表す変数。$i$ を省略した場合、任意のステップの受信者を表す

$X$ ：参加者の名前を表すメタ変数

$K_X$ ：X の公開鍵を表す定数

$K_X^{-1}$ ：X の秘密鍵を表す定数

$M_i$ ：プロトコルにおいてステップ $i$ で送受信されるメッセージを表す定数

$D_i$ ：プロトコルにおいてステップ $i$ でやり取りされるデータを表す定数

$M$ ：メッセージを表す変数

$N$ ：ナンスを表す変数

$K$ ：鍵を表す変数

$I$ ：イントルーダ（攻撃者）を表す定数

また、ナンス、データ、参加者の名前をアトミックメッセージと呼ぶ。

### 3.2 論理式

本論理で用いる論理式は、プロトコルの構造を表す論理式と、プロトコルの性質を表す論理式の 2 種類に大別することが出来る。

#### 3.2.1 プロトコルの構造を表す論理式

$M_1 \sqsubseteq M_2$ ：$M_1$ が $M_2$ のサブメッセージであることを表す。ただしメッセージ中で鍵として使われているものは、サブメッセージではない。サブメッセージは以下のように帰納的に定義される。

- $M \sqsubseteq M$
- $M \sqsubseteq M'$ ならば、$M \sqsubseteq \{M'\}_K$
- $M \sqsubseteq M_1$ か $M \sqsubseteq M_2$ ならば、$M \sqsubseteq M_1 M_2$

さらに、$M_1 \sqsubseteq_N M_2$ のように表記することで、$M_1$ は $M_2$ のサブメッセージで、かつ $N$ が $M_2$ のサブメッセージであるならば $M_1$ のみにサブメッセージとして含まれることを表す。

- $M \sqsubseteq_N M$
- $M \sqsubseteq_N M'$ ならば、$M \sqsubseteq_N \{M'\}_K$
- $M \sqsubseteq_N M_1$ かつ $N \not\sqsubseteq M_2$、もしくは $N \not\sqsubseteq M_1$ かつ $M \sqsubseteq_N M_2$ ならば、$M \sqsubseteq_N M_1 M_2$

$i \vdash M_1 \downarrow_Q M_2$ ：ステップ $i$ で、主体 $Q$ がメッセージ $M_1$ を受け取ったとき、分解や復号の操作により $M_1$ から $M_2$ を（知識として）取り出すことができる。

$i \vdash Q$ ***knows*** $N$ ：ステップ $i$ 以前のメッセージから、$Q$ は $N$ を知ることが出来る。

$i \vdash N_P \leftrightarrow P$ ：$N_P$ の作成者 $P$ は、ステップ $i$ で自分の名前 $P$ と $N_P$ を一緒に暗号化するか、$P$ を $N_P$ で暗号化して相手に送信する。

#### 3.2.2 プロトコルの性質を表す論理式

$i \vdash N$ ***is not extracted*** ：イントルーダはステップ $i$ までに送受信されたメッセージから $N$ を得られない。

$i \vdash N$ ***is not known*** ：$N$ の作成者がステップ $i$ まで実行した時点で、$N$ はイントルーダに知られておらず、$N$ の作成者とその意図した相手のみが知っている。

$i \vdash M$ ***is addressed to*** $N_Q$ ：$M_i$ のサブメッセージ $M$ は、$N_Q$ の作成者への宛先情報を含んでおり、かつイントルーダによって偽造されることはない。

$i \vdash Q$ ***recognize*** $(N_P, N_Q)$ ：$N_Q$ を作成した受信者 $Q$ は、ナンス $N_P$ が意図した通信相手の作成したもので、セッションに対応したナンスであることをステップ $i$ で認識する。（$Q$ は $P$ を認証する）

$i \vdash M$ ***is guaranteed*** ：$M_i$ のサブメッセージ $M$ は、イントルーダによる攻撃に対して耐性がある。つまりステップ $i$ の受信者は、$M$ は意図した通信相手が意図したステップで作成したものであることを確認出来る。

$D_i$ ***is warranted*** ：ステップ $i$ の受信者は、$D_i$ は意図した通信相手が意図したステップで送信したものであることを確認できる。（データ認証）

$D_i$ ***is secret*** ：データ $D_i$ はその作成者の意図する相手以外に知られることはない。（データの秘密性）

### 3.3 推論規則

#### 3.3.1 プロトコルの構造を導出する推論規則

プロトコルの構造を表す論理式は以下の規則によって導出される。これらの規則は [1] で導入されている。

**(DA1)** $M$ から $M$ を取り出すことが出来る。

$$\frac{}{i \vdash M \downarrow_Q M}$$

**(DA2)** $M_1$ から $M_2$ を取り出すことができ、$M_2$ から $M_3$ を取り出すことが出来るならば、$M_1$ から $M_3$ を取り出すことが出来る。

$$\frac{i \vdash M_1 \downarrow_Q M_2 \quad i \vdash M_2 \downarrow_Q M_3}{i \vdash M_1 \downarrow_Q M_3}$$

**(DA3)** $M$ から $M_1$ と $M_2$ の連接メッセージが取り出すことが出来るならば、$M$ から $M_1$ を取り出すことが出来、かつ $M$ から $M_2$ を取り出すことが出来る。

$$\frac{i \vdash M \downarrow_Q (M_1, M_2)}{i \vdash M \downarrow_Q M_1 \ \wedge \ i \vdash M \downarrow_Q M_2}$$

**(DA4)** $Q$ は $K_Q^{-1}$ を知っているため、$\{M\}_{K_Q}$ から $Q$ と $M$ を取り出すことが出来る。

$$\frac{}{i \vdash \{M\}_{K_Q} \downarrow_Q Q \ \wedge \ i \vdash \{M\}_{K_Q} \downarrow_Q M}$$

**(DA5)** $Q$ は $K_P$ を知っているため、$\{M\}_{K_P^{-1}}$ から $P$ と $M$ を取り出すことが出来る。

$$\frac{}{i \vdash \{M\}_{K_P^{-1}} \downarrow_Q P \ \wedge \ i \vdash \{M\}_{K_P^{-1}} \downarrow_Q M}$$

**(DA6)** $Q$ が $N$ を知っていれば、$\{M\}_N$ から $N$ と $M$ を取り出すことが出来る。

$$\frac{i \vdash Q \ knows \ N}{i \vdash \{M\}_N \downarrow_Q N \ \wedge \ i \vdash \{M\}_N \downarrow_Q M}$$

**(Knows1)** イニシエータ $A$ は $N_A$ を初めから知っている。同様にレスポンダ $B$ は $N_B$ を初めから知っている。

$$\frac{}{i \vdash A \ knows \ N_A \ \wedge \ i \vdash B \ knows \ N_B}$$

**(Knows2)** $i$ 以前のステップ $j$ のメッセージから、$Q_i$ は $N$ を取り出すことが出来る。

$$\frac{j < i. \ j \vdash M_j \downarrow_{Q_i} N}{i \vdash Q_i \ knows \ N}$$

**(B1)** 送信者 $P_i$ は、ナンス $N_{P_i}$ と名前 $P_i$ をメッセージ $\{M\}_{K_{Q_i}}$ に含めて相手に送信する。
$tag_i$ はタグを表しており、メッセージがステップ $i$ で送信されたという情報を持つ。仮にプロトコルの中で $\{M\}_{K_{Q_i}}$ という形のメッセージが一意的であるならば、$tag_i$ は必要ない。他の規則でも同様である。

$$\frac{i \vdash M_i \downarrow_{Q_i} \{M\}_{K_{Q_i}} \quad i \vdash M \downarrow_{Q_i} N_{P_i}}{i \vdash M \downarrow_{Q_i} P_i \quad i \vdash M \downarrow_{Q_i} tag_i}$$
$$i \vdash N_{P_i} \leftrightarrow P_i$$

**(B2)** 送信者 $P_i$ は、名前 $P_i$ を含むメッセージをナンス $N_{P_i}$ で暗号化して相手に送信する。

$$\frac{i \vdash M_i \downarrow_{Q_i} \{M\}_{N_{P_i}} \quad i \vdash \{M\}_{N_{P_i}} \downarrow_{Q_i} N_{P_i}}{i \vdash M \downarrow_{Q_i} P_i \quad i \vdash M \downarrow_{Q_i} tag_i}$$
$$i \vdash N_{P_i} \leftrightarrow P_i$$

### 3.3.2 プロトコルの性質を導出する推論規則

プロトコルの性質を表す論理式は、以下の規則によって導出される。

**(NN)** ナンス $N$ がサブメッセージとして含まれなければ、イントルーダに漏れることはない。

$$\frac{i-1 \vdash N \text{ is not extracted} \quad N \not\sqsubseteq M_i}{i \vdash N \text{ is not extracted}}$$

**(NP)** ナンス $N_{P_i}$ を相手の公開鍵 $K_{Q_i}$ で暗号化すれば、相手 $Q_i$ 以外に漏れることはない。

$$\frac{i-1 \vdash N_{P_i} \text{ is not extracted} \quad \{M\}_{K_{Q_i}} \sqsubseteq_{N_{P_i}} M_i \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash N_{P_i} \text{ is not extracted}}$$

**(NQ)** ナンス $N_{Q_i}$ をその作成者の公開鍵 $K_{Q_i}$ で暗号化すれば、$Q_i$ 以外に漏れることはない。

$$\frac{i-1 \vdash N_{Q_i} \text{ is not extracted} \quad j<i.\ j \vdash N_{Q_i} \leftrightarrow Q_i \quad \{M\}_{K_{Q_i}} \sqsubseteq_{N_{Q_i}} M_i \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash N_{Q_i} \text{ is not extracted}}$$

**(K0)** メッセージを送受信していなければ、ナンスは漏れない。

$$\frac{}{0 \vdash N \text{ is not extracted}}$$

**(KL)** 全てのメッセージから $N$ が漏れることがなければ、セッションを通して $N$ は秘密である。

$$\frac{k \vdash N \text{ is not extracted} \quad k \text{ は最終ステップ}}{k \vdash N \text{ is not known}}$$

**(K1)** ステップ $i$ まで実行した時点で $N_{Q_i}$ が秘密であり、ステップ $i+1$ のメッセージが $N_{Q_i}$ とタグを含む構造であれば、ステップ $i+1$ のメッセージがイントルーダによって偽造されることはない。よって $Q_i$ がステップ $i$ までしか実行しなければ、相手 $P_i$ もステップ $i$ までしか実行することが出来ない。$i$ までのメッセージから $N_{Q_i}$ は漏れないため、$Q_i$ がステップ $i$ まで実行した時点では $N_{Q_i}$ は秘密となる。

$$\frac{i \vdash N_{Q_i} \text{ is not extracted} \quad i+1 \vdash M_{i+1} \downarrow_{Q_{i+1}} \{M\}_{K_{Q_{i+1}}} \quad i+1 \vdash M \downarrow_{Q_{i+1}} N_{Q_i} \quad i+1 \vdash M \downarrow_{Q_{i+1}} tag_{i+1}}{i \vdash N_{Q_i} \text{ is not known}}$$

**(K2)**(K1) と同様

$$\frac{i \vdash N_{Q_i} \text{ is not extracted} \quad i+1 \vdash M_{i+1} \downarrow_{Q_{i+1}} \{M\}_{K^{-1}_{P_{i+1}}} \quad i+1 \vdash M \downarrow_{Q_{i+1}} N_{Q_i} \quad i+1 \vdash M \downarrow_{Q_{i+1}} tag_{i+1}}{i \vdash N_{Q_i} \text{ is not known}}$$

**(K3)**(K1) と同様

$$\frac{i \vdash N_{Q_i} \text{ is not extracted} \quad i+1 \vdash M_{i+1} \downarrow_{Q_{i+1}} \{M\}_{N_{Q_i}} \quad i+1 \vdash \{M\}_{N_{Q_i}} \downarrow_{Q_{i+1}} N_{Q_i} \quad i+1 \vdash M \downarrow_{Q_{i+1}} tag_{i+1}}{i \vdash N_{Q_i} \text{ is not known}}$$

**(KK)** ナンス $N$ が、ステップ $i+1$ まで実行した時点で秘密ならば、ステップ $i$ の時点でも秘密である。

$$\frac{i+1 \vdash N \text{ is not known}}{i \vdash N \text{ is not known}}$$

**(AD1)** 秘密の $N_{Q_i}$ を含み $Q_i$ の公開鍵で暗号化された $M$ を作成できるのは、通信相手は $N_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみ。

$$\frac{i \vdash N_{Q_i} \text{ is not known} \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{K_{Q_i}} \quad i \vdash M \downarrow_{Q_i} N_{Q_i} \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash \{M\}_{K_{Q_i}} \text{ is addressed to } N_{Q_i}}$$

**(AD2)** $N_{Q_i}$ と $Q_i$ を含み $P_i$ の署名がなされた $M$ を作成できるのは、通信相手は $N_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみ。

$$\frac{i \vdash M_i \downarrow_{Q_i} \{M\}_{K^{-1}_{P_i}} \quad i \vdash M \downarrow_{Q_i} N_{Q_i} \quad i \vdash M \downarrow_{Q_i} Q_i \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash \{M\}_{K^{-1}_{P_i}} \text{ is addressed to } N_{Q_i}}$$

**(AD3)** $Q_i$ を含み秘密の $N_{Q_i}$ で暗号化された $M$ を作成できるのは、通信相手は $N_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみ。

$$\frac{i \vdash N_{Q_i} \text{ is not known} \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{N_{Q_i}} \quad i \vdash M \downarrow_{Q_i} Q_i \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash \{M\}_{N_{Q_i}} \text{ is addressed to } N_{Q_i}}$$

**(AD4)** 既に $P_i$ が $N_{Q_i}$ とその持ち主の名前 $Q_i$ の対応を認識しているのであれば、秘密の $N_{Q_i}$ で暗号化された $M$ を作成できるのは、通信相手は $N_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみ。

$$\frac{i \vdash N_{Q_i} \text{ is not known} \quad j<i.\ j \vdash N_{Q_i} \leftrightarrow Q_i \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{N_{Q_i}} \quad i \vdash M \downarrow_{Q_i} tag_i}{i \vdash \{M\}_{N_{Q_i}} \text{ is addressed to } N_{Q_i}}$$

**(AD5)** $M$ を作成したのは、通信相手は $N^1_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ であり、$M$ に $N^2_{Q_i}$ が含まれるならば、$M$ を作成できるのは通信相手は $N^2_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみである。

$$\frac{i \vdash M \text{ is addressed to } N^1_{Q_i} \quad i \vdash M \downarrow_{Q_i} N^2_{Q_i}}{i \vdash M \text{ is addressed to } N^2_{Q_i}}$$

**(REC1)** $M$ を作成できるのは、通信相手を $N_{Q_i}$ を作成した $Q_i$ であると信じている $P_i$ のみであるため、受信者 $Q_i$ は、$N_{P_i}$ はそのような $P_i$ の作成したナンスであることを認識する。

$$\frac{\text{(ステップ } i \text{ 以前に } Q_i \text{が送信した全ての } N_{Q_i} \text{について) } i \vdash M \text{ is addressed to } N_{Q_i} \quad i \vdash M \downarrow_{Q_i} N_{P_i}}{i \vdash Q_i \text{ recognize } (N_{P_i}, N_{Q_i}) \wedge i \vdash M \text{ is guaranteed}}$$

**(REC2)**(REC1) によって $P_i$ が $N_{Q_i}$ が意図した相手のナンスであることを確認した後、$N_{Q_i}$ を返送することによって、$Q_i$ は $N_{P_i}$ が $P_i$ の作成したナンスであることを確認できる。

$$\frac{i \vdash M \text{ is addressed to } N_{Q_i} \quad j<i.\ j \vdash P_i \text{ recognize } (N_{Q_i}, N_{P_i})}{i \vdash Q_i \text{ recognize } (N_{P_i}, N_{Q_i}) \wedge i \vdash M \text{ is guaranteed}}$$

**(GP1)** 受信者がセッションとの対応を認識する送信者の秘密のナンス $N_{P_i}$ と、タグを含む暗号化メッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。i が最終ステップの場合は $i+1 \vdash M'$ *is guaranteed* の条件は必要ない。

$$\frac{i \vdash N_{P_i} \text{ is not known} \quad j \vdash Q_i \text{ recognize } (N_{P_i}, N_{Q_i}) \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{K_{Q_i}} \quad i \vdash M \downarrow_{Q_i} N_{P_i} \quad i \vdash M \downarrow_{Q_i} tag_i \quad i+1 \vdash M' \text{ is guaranteed}}{i \vdash \{M\}_{K_{Q_i}} \text{ is guaranteed}}$$

**(GP2)** 受信者がセッションとの対応を認識する送信者のナンス $N_{P_i}$ と、タグを含む署名メッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。

$$\frac{i \vdash M_i \downarrow_{Q_i} \{M\}_{K^{-1}_{P_i}} \quad i \vdash M \downarrow_{Q_i} N_{P_i} \quad i \vdash M \downarrow_{Q_i} tag_i \quad j \vdash Q_i \text{ recognize } (N_{P_i}, N_{Q_i})}{i \vdash \{M\}_{K^{-1}_{P_i}} \text{ is guaranteed}}$$

**(GP3)** タグを含み、受信者がセッションとの対応を認識する送信者のナンス $N_{P_i}$ で暗号化されたメッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。i が最終ステップの場合は $i+1 \vdash M'$ *is guaranteed* の条件は必要ない。

$$\frac{i \vdash N_{P_i} \text{ is not known} \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{N_{P_i}} \quad i \vdash M \downarrow_{Q_i} tag_i \quad i \vdash \{M\}_{N_{P_i}} \downarrow_{Q_i} N_{P_i} \quad j \vdash Q_i \text{ recognize } (N_{P_i}, N_{Q_i}) \quad i+1 \vdash M' \text{ is guaranteed}}{i \vdash \{M\}_{N_{P_i}} \text{ is guaranteed}}$$

**(GQ1)** 送信者がセッションとの対応を認識した、受信者の秘密のナンス $N_{Q_i}$ と、タグを含む暗号メッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。

$$\frac{i \vdash N_{Q_i}\ \textit{is not known} \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{K_{Q_i}} \quad i \vdash M \downarrow_{Q_i} N_{Q_i} \quad i \vdash M \downarrow_{Q_i} tag_i \quad j < i.\ j \vdash P_i\ recognize\ (N_{Q_i}, N_{P_i})}{i \vdash \{M\}_{K_{Q_i}}\ \textit{is guaranteed}}$$

**(GQ2)** 送信者がセッションとの対応を認識した、受信者のナンス $N_{Q_i}$、受信者の名前 $Q_i$、タグを含む署名メッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。

$$\frac{i \vdash M_i \downarrow_{Q_i} \{M\}_{K_{P_i}^{-1}} \quad i \vdash M \downarrow_{Q_i} N_{Q_i} \quad i \vdash M \downarrow_{Q_i} tag_i \quad i \vdash M \downarrow_{Q_i} Q_i \quad j < i.\ j \vdash P_i\ recognize\ (N_{Q_i}, N_{P_i})}{i \vdash \{M\}_{K_{P_i}^{-1}}\ \textit{is guaranteed}}$$

**(GQ3)** タグを含み、送信者がセッションとの対応を認識した受信者のナンス $N_{Q_i}$ で暗号化されたメッセージは、意図した通信相手が意図したステップで送信したものであることを確認できる。

$$\frac{i \vdash N_{Q_i}\ \textit{is not known} \quad i \vdash M_i \downarrow_{Q_i} \{M\}_{N_{Q_i}} \quad i \vdash M \downarrow_{Q_i} tag_i \quad j < i.\ j \vdash P_i\ recognize\ (N_{Q_i}, N_{P_i})}{i \vdash \{M\}_{N_{Q_i}}\ \textit{is guaranteed}}$$

**(WAR)** $M_i$ のサブメッセージである $M$ は、意図した通信相手が意図したステップで作成したものであることを確認できるため、$M_i$ に含まれる $D_i$ も意図した通信相手が意図したステップで作成したデータであることを受信者は確認できる。

$$\frac{i \vdash M\ \textit{is guaranteed} \quad i \vdash M \downarrow_{Q_i} D_i}{D_i\ \textit{is is warranted}}$$

**(SEC1)** $D_i$ は一度しか送受信されないため、相手の公開鍵で暗号化すればイントルーダに漏れることはない。

$$\frac{\{M\}_{K_{Q_i}} \sqsubseteq_{D_i} M_i}{D_i\ \textit{is secret}}$$

**(SEC2)** $D_i$ が送信者の作成した秘密のナンス $N_{P_i}$ で暗号化されていれば、イントルーダに漏れることはない。

$$\frac{k \vdash N_{P_i}\ \textit{is not known} \quad k \text{ は最終ステップ} \quad \{M\}_{N_{P_i}} \sqsubseteq_{D_i} M_i}{D_i\ \textit{is secret}}$$

**(SEC3)** $D_i$ が認証した相手の秘密のナンス $N_{Q_i}$ で暗号化されていれば、イントルーダに漏れることはない。

$$\frac{k \vdash N_{Q_i}\ \textit{is not known} \quad k \text{ は最終ステップ} \quad \{M\}_{N_{Q_i}} \sqsubseteq_{D_i} M_i \quad j < i.\ j \vdash P_i\ recognize\ (N_{Q_i}, N_{P_i})}{D_i\ \textit{is secret}}$$

### 3.4 セキュリティプロパティを表す式

本節では、セキュリティプロパティと論理式の対応を示す。つまり本論理において、どのように2.3節のセキュリティプロパティを表現しているかを述べる。

**データの秘密性** $D_i$ *is secret* が導出されればステップ $i$ で送受信されるデータ $D_i$ の秘密性が保証される。

**データ認証** やり取りされる全てのデータ $D_i$ について $D_i$ *is warranted* が導出されれば、データ認証が満たされる。

**受信確認** 受信確認を満たすためには、データのやり取りが終わった後（データを含まない）確認メッセージを送受信すればよい[2]。よってやり取りしたいデータが $D_k$ まで存在する場合、$k+1 \vdash M$ *is guaranteed* が導出されれば受信確認が満たされる。

## 4 プロトコルに対するコスト評価

本稿では、以下のコスト評価基準に基づきプロトコルを評価し、使用環境に適したプロトコルを選別する。

- メッセージの暗号化コスト
- メッセージの署名操作コスト
- メッセージの量的コスト
- 攻撃の検出時期コスト

やり取りされるメッセージの暗号化コストや量的コストはプロトコルの評価基準として一般的であるが、本稿ではこれに加え、攻撃が検出される時期に対してコスト基準を導入する。

本稿で生成するプロトコルは2.3節のセキュリティプロパティを満たしているため、攻撃者によってメッセージすり替え等の攻撃が行われたとしても、セッション参加者は必ずこれを検出することが出来る。しかし、その検出時期はプロトコルごとに異なり、全ての攻撃を即座に検出できるプロトコルもあれば、ある特定の攻撃に関しては後のステップでしか検出できないプロトコルもある。このような違いをコスト基準として導入した。本稿では、より早く攻撃を検出できるプロトコルの方が良いプロトコルであるという立場をとる。

## 5 自動生成法

本節では3節で示した論理を利用した、プロトコルの自動生成法を示す。まず初めに生成に必要な入力について述べ、その後プロトコルの生成手続きを示す。

### 5.1 入力

まず、2.4節で挙げたプロトコルの仕様を以下のように形式化し、入力として与える。

やり取りするデータの数とデータを送信し始める側が決定されれば、そのために必要なプロトコルの最小ステップ数が決定されるため、これをプロトコルのステップ数とする。これによりデータが送受信されるステップも決定される。

⟨step, Data, Secret⟩ : プロトコルの仕様

step: プロトコルのステップ数

Data: データを送受信するステップ番号の集合

Secret: 秘密にしたいデータが送受信されるステップ番号の集合。Data の部分集合

さらに4節のコスト基準に対するコスト重みを入力として与えることで、4の基準のうちどの基準を重視するかを指定することが出来る。つまり、コスト重みによってプロトコルが使用される環境が表現される。

### 5.2 生成手続き

プロトコルの生成手続きについて述べる。この生成法では、仕様とセキュリティプロパティを両方とも満たす構造を持つメッセージを元に、安全なプロトコルのみを生成する。

(1) メッセージ生成器によりステップ $i$ のメッセージ候補をコスト昇順に自動生成する。このメッセージ候補はプロトコルの仕様を満たしている。

(2) 生成されたメッセージ候補から、ステップ $i$ に関するセキュリティプロパティの式が導出されるかを、推論規則を用いて調べる。候補メッセージをコスト昇順に調べることで、セキュリティプロパティの式が導出される最小コストメッセージをステップ $i$ の送受信メッセージとして採用する。

(3) 全てのステップの送受信メッセージを決定し、プロトコルが生成される

以上の手続きにより、一つのセキュリティプロトコルが生成されるが、相手を認証するステップ等、仕様として与えられていない条件を変更することで、一つの入力から複数のセキュリティプロトコルが生成される。本生成法では、得られる可能性のあるプロトコルを全て生成する。その後プロトコル全体に対するコスト評価を行い、最小コストプロトコルを出力する。

## 6 推論規則の健全性

3.3節で示した推論規則の健全性を、ストランド空間における意味に基いて証明する。

### 6.1 ストランド空間

本稿で扱うストランド空間の定義は[4]に従う。ストランドとは符号付メッセージの列である。ストランドはスレッドのようなもので、参加者によって順番に送受信されるメッセージの列を表す。符号付メッセージは $+M$ と $-M$ のどちらかであり、$+M$ によってメッセージ $M$ の送信を表し、$-M$ によってメッセージ $M$ の受信を表す。ストランドに現れるこれらの符号付メッセージをノードと呼ぶ。例えば $\langle +M_1, -M_2 \rangle$ はストランドであり、初めのノードで $M_1$ を送信し、二番目のノードで $M_2$ を受信する。また、ストランドの持つノードの総数をそのストランドの高さと呼ぶ。

また、攻撃者でない正当な参加者の振る舞いを表すストランドをレギュラーストランド、攻撃者の振る舞いを表すストランドをイントルーダストランドと呼ぶ。また、イニシエータの振る舞いを表すストランドをイニシエータストランド、レスポンダの振る舞いを表すストランドをレスポンダストランドと呼ぶ。

イントルーダストランドは、攻撃者が行える全ての可能性を考慮する必要があるため、攻撃者の行える基本動作の組み合わせで定義する。基本動作はアトミックメッセージの生成、メッセージの消失、メッセージの二重化、連結、分解、鍵使用、暗号化、復号化（署名）であり、これらを組み合わせることにより、2.2節で定義した攻撃者の可能な行為を全て表現できる。

生成されるプロトコルのストランド空間について、以下の変数を定義する。

$a, b, \cdots$ ： 参加者の名前を表す変数
$n_a, n_b, \cdots$ ： $a, b, \cdots$ の作成したナンスを表す変数
$d_1, d_2, \cdots$ ： やり取りされるデータを表す変数

また、パラメータを利用してストランドを表記することも出来る。例えば、イニシエータの名前 $a$、レスポンダの名前 $b$、イニシエータのナンス $n_a$、レスポンダのナンス $n_b$、やり取りするデータ $d_1, d_2$ であるセッションにおける、高さ $i$ のイニシエータストランドを $INI(a, b. n_a, n_b, d_1, d_2)_i$、レスポンダストランドを $RES(a, b. n_a, n_b, d_1, d_2)_i$ と表記する。さらに $*$ を利用した省略形を定義する。

$$INI(a, b, n_a, *) = \bigcup_{n_b} INI(a, b, n_a, n_b)$$

$*$ は任意の値を表しており、どのような値が入っても構わない。本稿ではしばしば、やり取りするデータを省略したストランド表記を用いるが、その場合もデータの値は任意である。

### 6.2 論理式に与えられるストランド空間における意味

各論理式に対して、ストランド空間における意味を与える。プロトコルの構造を表す論理式の持つ意味は、そのような構造を持つプロトコルのストランド空間を考慮しているという意味でしかないため、ここではプロトコルの性質を表す論理式に対してのみ意味を定義していく。

$i \vdash N$ ***is not extracted*** ：$N$ がイニシエータのナンスの場合を考える。高さ $i+1$ 以上の $INI(a, b, n_a, *)$ が存在せず、かつ高さ $i+1$ 以上の $RES(*, b, n_a, *)$ も存在しなければ、$n_a$ を知っている可能性のあるレギュラーストランドは $INI(a, b, n_a, *)$ と $RES(*, b, n_a, *)$ のみであり、かつ $+n_a$ をノードとして持つイントルーダストランドは存在しない

$i \vdash N$ ***is not known*** ：$N$ がイニシエータのナンスの場合を考える。高さ $i+1$ 以上の $INI(a, b, n_a, *)$ が存在しないならば、$n_a$ を知っている可能性のあるレギュラーストランドは $INI(a, b, n_a, *)$ と $RES(*, b, n_a, *)$ のみであり、かつ $+n_a$ をノードとして持つイントルーダストランドは存在しない。

$i \vdash M$ ***is addressed to*** $N_Q$ ：ステップ $i$ ではイニシエータが受信側とする。$i$ 番目のノードに、$M$ を含むメッセージの受信ノードを持つ $INI(a, b, n_a, *)_i$ が存在するならば、$i$ 番目のノードに $M$ を含むメッセージの送信ノードを持つ、高さ $i$ 以上の $RES(a, b, n_a, *)$ が存在する。

$i \vdash Q$ ***recognize*** $(N_P, N_Q)$ ：ステップ $i$ ではイニシエータが受信側とする。$INI(a, b, n_a, n_b)_i$ が存在するならば、高さ $i$ 以上の $RES(a, b, n_a, n_b)$ が存在する

$i \vdash M$ ***is guaranteed*** ：ステップ $i$ ではイニシエータが受信側とする。$i$ 番目のノードに、$M$ を含むメッセージの受信ノードを持つ $INI(a, b, n_a, n_b)_i$ が存在するならば、$i$ 番目のノードに $M$ を含むメッセージの送信ノードを持つ、高さ $i$ 以上の $RES(a, b, n_a, n_b)$ が存在する。
※ただし、$a$ による $b$ の認証が $i$ より後のステップ $j$ で行われる場合、この式の意味は以下のようになる
ステップ $i$ ではイニシエータが受信側とする。$i$ 番目のノードに、$M$ を含むメッセージの受信ノードを持つ $INI(a, b, n_a, n_b)_j$ が存在するならば、$i$ 番目のノードに $M$ を含むメッセージの送信ノードを持つ、高さ $j$ 以上の $RES(a, b, n_a, n_b)$ が存在する。

$D_i$ ***is warranted*** ：ステップ $i$ ではイニシエータが受信側とする。$i$ 番目のノードに、$d_i$ を含むメッセージの受信ノードを持つ $INI(a, b, n_a, n_b, d_i)_i$ が存在するならば、$i$ 番目のノードに $d_i$ を含むメッセージの送信ノードを持つ、高さ $i$ 以上の $RES(a, b, n_a, n_b, d_i)$ が存在する。加えて、受信者はステップ $i$ の送受信メッセージから $d_i$ を取り出すことが出来る。

$D_i$ ***is secret*** ：ステップ $i$ ではイニシエータが送信側とする。$i$ 番目のノードに $d_i$ を含むメッセージの送信ノードを持つ $INI(a, b, n_a, n_b, d_i)$ が存在するならば、$d_i$ を知る可能性のあるレギュラーストランドは $RES(*, b, *, *, d_i)$ のみで、かつ $+d_i$ をノードとして持つイントルーダストランドは存在しない。

### 6.3 推論規則の健全性の証明

3.3節の推論規則が、式に与えられたストランド空間における意味を保存することを証明する。これにより、定理1が示され推論規則の健全性が言える。

対象プロトコルの条件より、考慮するストランド空間では以下の性質が成立すると仮定する。これは、正当な参加者がセッションで作成するナンスは新規なもので、以前に使用されたものを使用することはないという意味である。

**仮定1（ナンスの新規性）** あるストランド空間にレギュラーイニシエータストランド $INI(a, b, n_a, n_b, d_1, \cdots)$ と、レギュラーイニシエータストランド $INI(a', b', n_a, n'_b, d'_1, \cdots)$ が存在するならば、$a = a', b = b'. n_b = n'_b, d_1 = d'_1, \cdots$ が成立する。同様に、あるストランド空間にレギュラーレスポンダストランド $RES(a, b, n_a, n_b. d_1, \cdots)$ と、レギュラーレスポンダストランド $RES(a', b', n'_a, n_b, d'_1, \cdots)$ が存在するならば、$a = a', b = b', n_a = n'_a, d_1 = d'_1, \cdots$ が成立する。

**(K1) の証明**：$N_{Q_i}$ はイニシエータが作成したナンスであるとすると、ステップ $i$ ではイニシエータが受信側となる。今、$INI(a, b, n_a, *)_i$ が存在するとする。仮にまだ $RES(*, b, n_a, *)$ の高さが $i$ 以下であるとすると、$i \vdash N_{Q_i}$ *is is not extracted* より、$n_a$ を知っている可能性のある

レギュラーストランドは$INI(a,b,n_a,*)_i$と$RES(*,b,n_a,*)$のみであり、かつ$+n_a$をノードとして持つイントルーダストランドは存在しない。ここで、ステップ$i+1$ではイニシエータが送信側であり、ステップ$i+1$で送受信されるメッセージにはタグと秘密の$n_a$が含まれるため、イントルーダがこのメッセージの送信ノードを持つことはない。よって$RES(*,b,n_a,*)$の高さが$i+1$以上になるためには、高さ$i+1$以上の$INI(a,b,n_a,*)$が存在しなければならない。つまり、高さが$i+1$以上の$INI(a,b,n_a,*)$が存在しないならば、高さ$i+1$以上の$RES(*,b,n_a,*)$は存在しないことになる。今、$INI(a,b,n_a,*)$の高さは$i$であることと$i \vdash N_{Q_i}$ *is not extracted*より、$n_a$を知っている可能性のあるレギュラーストランドは$INI(a,b,n_a,*)$と$RES(*,b,n_a,*)$のみであり、かつ$+n_a$をノードとして持つイントルーダストランドは存在しない。

**(GQ1) の証明**：ステップ$i$ではイニシエータが受信側とする。$i$番目のノードに、$\{M\}_{K_a}$を含むメッセージの受信ノードを持つ$INI(a,b,n_a,n_b)_i$が存在するとする。この時、$i \vdash N_{Q_i}$ *is not known*より$INI(a,b,n_a,n_b)_i$以外に$n_a$を知っている可能性があるのは$RES(*,b,n_a,*)$のみであり、$i \vdash M \downarrow_{Q_i} N_{Q_i}$より$RES(*,b,n_a,*)$が存在する。さらに$M$が$a$の公開鍵で暗号化されていることから$RES(a,b,n_a,*)$が存在し、$i \vdash M \downarrow_{Q_i} tag_i$より高さ$i$以上の$RES(a,b,n_a,*)$が存在することになる。ここで$j<i$. $j \vdash P_i$ *recognize* $(N_{Q_i}, N_{P_i})$より、高さ$j$以上の$INI(a,b,n_a,*)$が存在することになるが、ナンスの新規性より$INI(a,b,n_a,*)$は$INI(a,b,n_a,n_b)$であるため、$RES(a,b,n_a,n_b)$が存在しなければならない。以上より$i$番目のノードに$\{M\}_{K_a}$を含むメッセージの送信ノードを持つ、高さ$i$以上の$RES(a,b,n_a,n_b)$が存在する。これ以外の規則に対する証明は、本稿では省略する。

**定理 1** 全ての$D_i$について$D_i$ *is warranted*が導出されるとする。また、やり取りするデータが$D_k$まで存在し、ステップ$k$ではイニシエータが受信側であるとする。この時$INI(a,b,n_a,n_b,d_1,d_2,\cdots,d_k)_k$が存在するならば、高さ$k$以上の$RES(a,b,n_a,n_b,d_1,d_2,\cdots,d_k)$が存在する。逆にステップ$k$ではレスポンダが受信側であるとすると、$RES(a,b,n_a,n_b,d_1,d_2,\cdots,d_k)_k$が存在するならば、高さ$k$以上の$INI(a,b,n_a,n_b,d_1,d_2,\cdots,d_k)$が存在する。

これはストランド空間におけるagreementプロパティである[4]。agreementプロパティは、あるストランドに対してそのパラメータと高さが一致した相手ストランドの存在を保証するものである。

## 7 関連研究との比較

本節では関連研究との比較を行う。関連研究の特徴に触れながら、本稿で提案した自動生成法の利点について言及する。

### 7.1 Automatic Protocol Generator

[3]にて提案された自動生成法で、2者間で認証を行いセッション鍵を秘密共有するためのプロトコルを自動生成する。まず、プロトコル生成器により、入力として与えられた仕様を満たすプロトコルを大量に生成し出力する。次に、得られたプロトコル群をプロトコル自動検証器にて検証し、セキュリティプロパティを満たすプロトコルを選別する。自動検証器にはAthena[5]を使用する。そして、セキュリティプロパティを満たしたプロトコルのみを出力する。この生成法では、大量のプロトコルに対して検証を行わなければならないため、大きな計算コストがかかってしまうという欠点があった。これに対し本稿で提案した自動生成法では、安全性が保証されているメッセージ構造を元に設計を行うため、安全なプロトコルのみを効率的に生成することが出来る。

### 7.2 束縛関係に基く検証法

これは[1]にて提案された。推論規則によって、プロトコルの構造から直接的にプロトコルの秘密性や認証性を検証する。束縛関係とは、ナンス等のメッセージ間の対応関係のことである。ある参加者が相手を認証するということは、相手の名前やナンスを自分の名前やナンスと対応付けることと考えられるため、この対応関係に基づいて認証性を検証する。束縛関係に関する推論規則では、本稿で提案した推論規則と同様、プロトコルの静的な構造から安全性を検証することができる。よって、これを利用してプロトコルの自動生成を行うことも可能である。しかし束縛関係に基く推論規則を用いた場合では、生成されないようなプロトコルが存在する。束縛関係に基く推論規則では、ナンスの秘密性に関する導出を行った後にこれを利用して認証性を導くことや、ナンスの秘密性に対する要件が厳しすぎるという理由から、ナンスを秘密共有しないプロトコルやセッション途中までしかナンスを秘密保持しないようなプロトコルは生成されない。しかし、本稿では提案する推論規則では、より厳密にナンスの秘密性について調べており、これらのプロトコルも生成することが出来る。

## 8 まとめ

本稿では、二者間データ通信のためのプロトコルの自動生成法を提案した。これにより、使用環境に適した安全なプロトコルを効率的に自動生成出来るようになった。また、関連研究との比較を行い、本生成法の特徴について述べた。また、近年ではインターネット上において様々な種類の取引が行われており、認証や秘密性以外のセキュリティプロパティを満たすプロトコルが提案されてきている。これらのプロトコルの自動生成が今後の課題として挙げられる。

## 参考文献


[1] 萩谷昌己, 竹村亮, 高橋孝一, 斉藤孝道:束縛関係に基づく認証プロトコルの検証. コンピュータソフトウェア,vol.20,No.3(2003),pp.17-29

[2] 根岸和義: 平行セッションを利用した攻撃を考慮するセキュリティプロトコルの安全性検証に関する研究. 東京工業大学大学院, 博士論文, 2000 年 9 月.

[3] Adrian Perrig and Dawn Song: Looking for diamonds in the dessert — extending automatic protocol generation to three-party authentication and key distribution. In Proc. of IEEE Computer Security Foundation Workshop, July 2000.

[4] F. Javier Thayer Fabrega, Jonathan C. Herzog, Joshua D. Guttman. Strand spaces: proving security protocols correct? J. Computer Security, 7(1) 1999.

[5] D.Song, S.Berezin, and A.Perrig: Athena: a novel approach to efficient automatic security protocol analysis. Journal of Computer Security.9(1):47-74,2001.