社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

暗号プロトコル記述からカラーペトリネットへの変換による機密性検証

奥谷　大介† 　坂部　俊樹†† 　酒井　正彦†† 　草刈　圭一朗†† 　西田　直樹††

†, †† 名古屋大学大学院情報科学研究科
〒 464-8603 名古屋市千種区不老町
E-mail: †okuya@sakabe.i.is.nagoya-u.ac.jp, ††{sakabe,sakai,nishida,kusakari}@is.nagoya-u.ac.jp

あらまし　カラーペトリネット（CPN）の可達性判定は決定可能であるので，暗号プロトコルの安全性をカラーペトリネットの可達性判定に帰着することにより，機械的な安全性検証が可能である．本稿では，プロトコル記述言語によって記述された暗号プロトコルを CPN に変換するアルゴリズムを提案し，安全性の一つである機密性が，変換によって得られる CPN の可達性判定問題に帰着させて検証できることを示す．

キーワード　カラーペトリネット，機密性検証，可達性

Secrecy Verification by Transforming Cryptographic Protocol Descriptions to Coloured Petri Nets

Daisuke OKUYA†, Toshiki SAKABE††, Masahiko SAKAI††,

Keiichirou KUSAKARI††, and Naoki NISHIDA††

†, †† Graduate School of Information Science, Nagoya University
Furou-chou, Chikusa-ku, Nagoya, 464-8603, Japan
E-mail: †okuya@sakabe.i.is.nagoya-u.ac.jp, ††{sakabe,sakai,nishida,kusakari}@is.nagoya-u.ac.jp

Abstract　Verification of the safety of cryptographic protocols can be mechanized by reducing the safety to the reachability of Coloured Petri Nets (CPNs) since the reachability of CPNs is known to be decidable. In this paper, we propose an algorithm that transforms cryptographic protocol descriptions to CPNs, and show that the protocol is secure if no critical states are reachable from the initial states in the CPN obtained by the transformation from a given cryptographic protocol.

Key words　coloured petri net, secrecy verification, reachability

1. はじめに

ネットワークの処理を保護するための暗号プロトコルにおいて，通信の侵入者の攻撃や不正行為に対して安全であることを検証する研究がさまざまな方法で行われている．その検証手法の一つとして，spi 計算 [3] やプロトコル記述言語 [1] などのプロセス計算言語を用いた手法が挙げられる．プロセス計算言語を用いた手法は，暗号プロトコルの通信手順からのモデル化を容易に行うことができる反面，意味論に従った検証に手間がかかる．別の検証手法として，カラーペトリネットを用いた手法がある [4]．カラーペトリネットでは，プロセス計算言語と比較してモデル化が難しいが，モデルが図的に表現されるため，システムの動的挙動の表現と解析に適しており，可達性判定が決定可能であること [2] を利用した，検証の自動化を支援するツールも開発されている [5]．

本稿では，上記の二種類の検証手法の利点を利用できるように，つまり暗号プロトコルのモデル化はプロトコル記述言語を用いて行い，安全性の検証はカラーペトリネットの可達性判定問題として検証できるように，プロトコル記述をカラーペトリネットに変換するアルゴリズムを提案する．そして，変換されたカラーペトリネットにおいて，機密性が正しく検証できることを証明する．なお，機密性とは安全性の一つであり，暗号化された情報が侵入者に知られることがないことを保証する性質である．

2. 準　　備

本稿で扱うプロトコル記述言語の構文と意味論 [1]，およびカラーペトリネットの基本的な概念について述べる [4].

2.1 プロトコル記述言語

最初に，言語の構文を定義する．まず以下の集合を与える．

- Names：名前の集合（$n, m, A, B \in$ Names）
- Indexes：インデックスの集合（$i \in$ Indexes）
- Nvar：名前の変数集合（$x, y, z \in$ Nvar）
- Mvar：メッセージの変数集合（$X, Y \in$ Mvar）

上記の集合を用いて，その他の集合を BNF 記法により図 1 のように定める．

Values	v, v'	$::=$	$n \mid x \mid X$
Keys	k	$::=$	$Pub(v) \mid Priv(v) \mid Key(v, v')$
Messages	M, N	$::=$	$v \mid (M, N) \mid \{M\}_k$
Patterns	Π, Π'	$::=$	$v \mid (\Pi, \Pi')$
Process	P	$::=$	nil
		$\mid$	$new(x).P$ new name
		$\mid$	$out\ M.P$ output
		$\mid$	$in\ \Pi.P$ input
		$\mid$	$[M = N].P$ match
		$\mid$	$[M > \Pi].P$ case
System	Q	$::=$	$\|\|_{i \in I}\ P_i$ parallel

図 1　プロトコル記述言語の構文

この言語を用いて，System でプロトコル全体，Process で送信者，受信者，侵入者といった各関係者の動作を表現する．

$\{M\}_k$ はメッセージ M を鍵 k で暗号化したメッセージを表し，$[M > \Pi]$ はメッセージ M を復号化したときパターン Π に適合するかを調べる役割を持つ．$out\ M$ と $in\ \Pi$ は各々メッセージ M の出力とパターン Π の入力を表し，$[M = N]$ ではメッセージ M と N を照合することを表す．また，$\|_{i \in I}\ P_i$ はインデックス i を持つプロセスが並行動作することを表現し，特に $I = \mathbb{N}$ のときはこれを $!P$ と略記する．

次に，言語の意味論を遷移系 $(S, L, Tran)$ として与える．

- S は P，s，t の三項組からなり，P は Process，s は Values の部分集合，t は Messages の部分集合である．
- L は次で定義するアクション α の集合である．

$$\alpha \ ::= \ out(M) \mid in(M) \mid i : \alpha$$

- $Tran$ は図 2 の規則を再帰的に適用して得られる最小の関係である（$Tran \subseteqq S \times L \times S$）．

$$out(M).P, s, t \xrightarrow{out(red(M))} P, s, t \cup \{red(M)\} \quad (send)$$

$$\frac{\Pi[\sigma] \in t}{in(\Pi).P, s, t \xrightarrow{in(\Pi[\sigma])} P[\sigma], s, t} \quad (receive)$$

$$\frac{P[n/x], s \cup \{n\}, t \xrightarrow{\alpha} P', s', t'}{new(x).P, s, t \xrightarrow{\alpha} P', s', t'} \ n \notin s \quad (new)$$

$$\frac{P, s, t \xrightarrow{\alpha} P', s', t'}{[M = M].P, s, t \xrightarrow{\alpha} P', s', t'} \quad (match)$$

$$\frac{M = \Pi[\sigma] \ \ P[\sigma], s, t \xrightarrow{\alpha} P', s', t'}{[M > \Pi].P, s, t \xrightarrow{\alpha} P', s', t'} \quad (case)$$

$$\frac{P_j, s, t \xrightarrow{\alpha} P'_j, s', t'}{\|_{i \in I}\ P_i, s, t \xrightarrow{j:\alpha} \|_{i \in I}\ P_i[P'_j/j], s', t'} \quad (par)$$

図 2　プロトコル記述言語の意味論

図 2 において，$\Pi[\sigma]$ はパターン Π の変数に代入 σ を適用したものを表し，$\|_{i \in I}\ P_i[P'_j/j]$ は $\|_{i \in I}\ P_i$ 中の P_j だけを P'_j に置換したものを表す．

また $((P_i, s_i, t_i), \alpha_i, (P_{i+1}, s_{i+1}, t_{i+1})) \in Tran$，$(0 \geqq i \geqq n)$ のとき，遷移系列

$$(P_0, s_0, t_0) \xrightarrow{\alpha_0} (P_1, s_1, t_1) \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_{n-1}} (P_n, s_n, t_n) \xrightarrow{\alpha_n} (P_{n+1}, s_{n+1}, t_{n+1})$$

をプロセス P_0 の計算といい，アクションの系列 $\alpha_0 \cdots \alpha_n$ を P_0 のトレースという．各時点において，t_i は Process によって既に出力されたメッセージを表している．

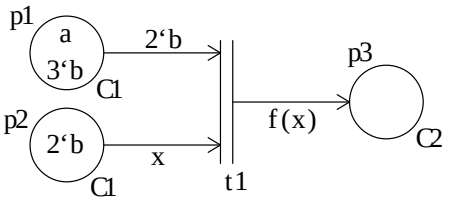
2.2 カラーペトリネット

カラーペトリネットは $CPN = (\Sigma, P, T, A, N, C, G, E, I)$ の 9 項組である．ここで Σ はトークンの型を定めるカラーの有限集合，P はプレイスの有限集合，T はトランジションの有限集合，A はアークの有限集合，N はプレイスとトランジションの接続を表す写像であるノード関数，C はプレイスに属するカラーの集合を定めるカラー関数，G はトランジションの発火を制御するためのガード関数．E はアークの発火条件を表現するアーク関数，I は CPN の最初の状態（マーキング）を定める初期関数をそれぞれ表す．

カラーペトリネットはプレイスとトランジションの 2 種類の接点を持つ有向 2 部グラフとしてみることができ，図 3 上のように図示する．◯ はプレイス，‖ はトランジション，→ はアークを表す．また，図 3 下のように，プレイスに属するカラー集

合の要素と，アークに与える発火の条件であるアーク関数の記述も同時に与える．なお，トランジションにガード関数を与えることにより，ガード関数の条件が真であるときのみ発火させることができるが，本稿で扱う CPN ではガード関数による発火の制御をさせることがないので，以下ではガード関数の記述を省略する．

図 3 では，プレイス p1 にトークン a が 1 個，b が 3 個，プレイス p2 にトークン b が 2 個，そしてプレイス p3 にはトークンがないことを表している．




```
colour C1 = with a|b;
colour C2 = with d|e;
var x : C1;
fun f(x) = if x=a then 2‘d else 3‘e;
```

図 3　カラーペトリネットの図表現

あるトランジションにおいて，トランジションに向かって接続している全てのプレイスの持つトークンがアーク関数の条件を満たしているとき，そのトランジションは発火可能であるという．トランジションが発火すると，トランジションに向かって接続している全てのプレイスのトークンがアーク条件に従って失われると共に，トランジションから出て接続している全てのプレイスのトークンがアーク条件に従って加えられる．

図 3 のカラーペトリネットにおいて，トランジション t1 は発火可能であり，t1 を発火させると，プレイス p1 から 2 個の b とプレイス p2 から 1 個の b が失われ，プレイス p3 に 3 個の e が加えられ，図 4 のようにマーキングが変化する．

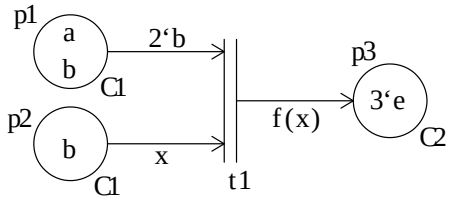


図 4　発火によるマーキングの遷移

マーキング M' からマーキング M へ発火の系列が存在するとき，M は M' から可達であるという．

定理 1　カラーペトリネットの可達性判定は決定可能である [2]. □

3. 変換アルゴリズム

プロトコルを表す System から，機密性判定を可達性問題として解けるような CPN へ変換するアルゴリズムを与える．

まず，通信の関係者，つまりプロトコルを通じて通信を行う送信者と受信者，また通信を妨害する侵入者に対応するそれぞれのプレイスと，ネットワークを表すプレイスを，変数 w でつなげたトランジションをもつ図 5 のようなカラーペトリネット CPN_0 を用意する．この CPN_0 の各トランジションは，各関係者がネットワーク上の情報を自由に知ることができることを表現している．

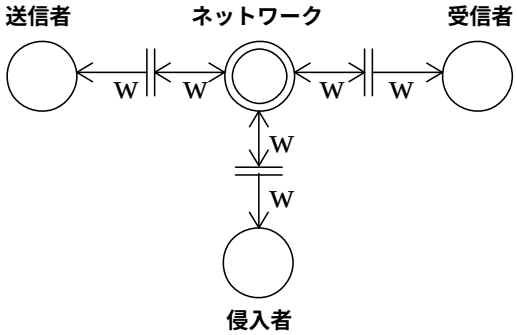


図 5　基本となるカラーペトリネット CPN_0

この CPN_0 を基本とし，プロトコルを用いた送受信者の動作と，侵入者の持つ能力に応じてトランジションを追加していき，目的とする CPN_k を構成する．

入力：　$Q = P_1 \parallel P_2 \parallel \cdots \parallel P_k$

出力：　CPN_k

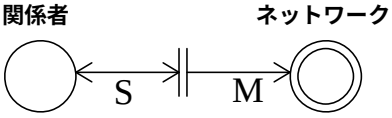
手順：　For $i = 1, 2, \cdots, k$　$\mathrm{CPN}_i = \mathrm{Trans}(P_i, \emptyset, \mathrm{CPN}_{i-1})$;

ここで関数 Trans(P, S, C) は次のように与えられる．下図において関係者とは，Trans が適用される P を行う関係者のことで，k’ は k と対になる鍵を表す．公開鍵 $Pub(v)$ は秘密鍵 $Priv(v)$ が対となり，共通鍵 $Key(v, v')$ は自身が対となる．

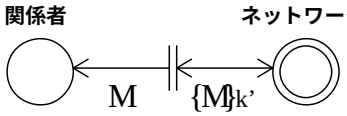
```
fun Trans (P, S, C) =
  case P of nil  => C
   | new(x).P’   => Trans(P’, S∪{x}, *1)
   | out M.P’    => Trans(P’, ∅, *2)
   | in Π.P’     => Trans(P’, S∪{Π}, C)
   | [M=N].P’    => Trans(P’, S, C)
   | [{M}k>Π].P’ => Trans(P’,S\{M} ∪{Π}, *3);
```

*1 : 関係者のプレイスにトークン x を追加する

*2 : 次のトランジションを追加する



*3 ： 次のトランジションを追加する



第一引数 P は，入力のプロトコル記述であり，その先頭の Process によって，関数 Trans が再帰されていく．第二引数の S は出力するカラーペトリネットの，動作に該当するトランジションの発火条件の候補となるものを保持しておく役割をし，メッセージの生成（new name）や入力（input）の Process である場合，そのメッセージが発火条件となる．また，メッセージの復号化（case）の Process である場合は，それ以前に保持していたメッセージの代わりに，そのメッセージを復号化したものが発火条件に加わることになる．第三引数の C は出力するカラーペトリネット自体を保持しておくもので，メッセージの出力（output）や復号化（case）の Process である場合に，それまで第二引数 S で保持しておいた発火条件をトランジションの入力とし，出力，復号されるメッセージをトランジションの出力とするようなトランジションを，関係者とネットワークのプレイス間に新しく追加する．第一引数 P の Process が nil となったら，その Process を持っている関係者の動作がカラーペトリネットに追加され終えたことになり，保持しておいた第三引数 C を出力する．

また，各プレイスごとにトークンの属するカラーを決めるカラー集合については，全てのプレイスに MESSAGE という共通のカラー集合を割り当てる．MESSAGE は入力 Q 中の Values をカラーとして持つ VALUE から図 6 のようにして構成する．

```
colour VALUE = with (* 入力 Q 中の Values *);
colour PAIR = list VALUE;
colour PUB = product VALUE * PAIR;
colour PRIV = product VALUE * PAIR;
colour KEY = product VALUE * VALUE * PAIR;
colour MESSAGE = union value:VALUE + pair:PAIR
                 + pub:PUB + priv:PRIV + key:KEY;
```

図 6　変換後のカラーペトリネットのカラー

上記のカラー集合を用いることで，プロトコル記述言語における Messages が，CPN におけるカラー集合 MESSAGE の要素として表すことができる．表 1 に各々の対応を示す．

表 1　Mesages と MESSAGE の対応

プロトコル記述言語	カラーペトリネット
v	value (v)
(M, N)	pair ($[M, N]$)
$\{M\}_{Pub(v)}$	pub (v, $[M]$)
$\{M\}_{Priv(v)}$	priv (v, $[M]$)
$\{M\}_{Key(v,v')}$	key (v, v', $[M]$)

4. 暗号プロトコルの検証手法

本稿における，暗号プロトコルの機密性検証の手順を示す．

最初に，暗号プロトコルを通信手順を元にしてプロトコル記述言語によってモデル化する．次にそのプロトコル記述を，変換アルゴリズムを用いてカラーペトリネットに変換する．そして，変換されたカラーペトリネットの可達性判定問題を解くことにより，暗号プロトコルの機密性を示す．暗号プロトコルをカラーペトリネットで検証する理由は，カラーペトリネットの可達性判定が決定可能であることにより，ツールを用いての機械的に検証が可能であるからであり，一旦プロトコル記述言語で表現するのは．プロトコル記述言語がカラーペトリネットに比べて，通信手順からのモデル化が容易であるからである．

以下では，暗号プロトコルとして，Needham-Schroeder Public Key プロトコル（NSPK プロトコル）を検証する場合を例として示す．

4.1 プロトコル記述言語によるモデル化

NSPK プロトコルは，送信者と受信者がお互いに生成した乱数を安全な方法で受け渡しを行うことを目的とする，公開鍵方式暗号プロトコルである．公開鍵方式のプロトコルでは，公開鍵と秘密鍵という二つの対になった鍵を用いて通信を行う．ある公開鍵を使って暗号化された情報は，その公開鍵と対になった秘密鍵でのみ復号可能であり，逆にある秘密鍵を使って暗号化された情報は，その秘密鍵と対になった公開鍵でのみ復号可能である．関係者は自分の公開鍵と秘密鍵を用意し，公開鍵は他者が使用できるように公開しておき，秘密鍵は自分以外に知られないように保持しておく．

以下に送信者を A，受信者を B として NSPK プロトコルの通信手順を示す．

（1） $A \rightarrow B$ ： $\{N_A,\ A\}_{PK(B)}$

（2） $B \rightarrow A$ ： $\{N_A,\ N_B,\ B\}_{PK(A)}$

（3） $A \rightarrow B$ ： $\{N_B\}_{PK(B)}$

NSPK プロトコルは上のような 3 ステップで表される．ここで、$A \rightarrow B : \alpha$ は A から B へ α を送信することを，$\{\alpha\}_\beta$ は，α を公開鍵 β で暗号化することを表している．また，$PK(A)$ は A の公開鍵，N_A は A が生成する乱数を表す。

最初に，A は自分が生成した乱数と，自分の名前を B の公開鍵で暗号化して B に送信する．それを受信した B は自分の持っている秘密鍵により復号化を行う．次に，取り出した A が生成した乱数に，自分が生成した乱数と自分の名前を一緒にして，A の公開鍵で暗号化をして A に送信する．A はそれを受信，自分の秘密鍵で復号化して，先ほど送信した自分が生成した乱数が正しく戻ってきたかを確認する．その後，再び B が生成した乱数を B の公開鍵で暗号化したものを B へ送信し，それを受け取った B は自分の生成した乱数であるかを照合して確認する．

これをプロトコル記述言語を用いてモデル化する．プロトコ

ル記述言語では，各関係者ごとに分けてモデル化を行い，上記の通信手段では見られなかった，メッセージの生成や，復号化，照合といった動作もそれに対応する Process を用いて表現する．NSPK プロトコルの送信者を表す Process を $Init(A,B)$，受信者を表す Process を $Resp(B)$ として図 7 に示す．

$Init(A,B) \equiv$
$new(x).$
$out\ \{(x,A)\}_{Pub(B)}.$
$in\ X.$
$[\{X\}_{Priv(A)} > (z,(y,B))].$
$[x = z].$
$out\ \{y\}_{Pub(B)}.$
nil

$Resp(B) \equiv$
$inX.$
$[\{X\}_{Priv(B)} > (x,u)].$
$new(y).$
$out\ \{(x,(y,B))\}_{Pub(u)}.$
$in\ X'.$
$[\{X'\}_{Priv(B)} > z].$
$[y = z].$
nil

図 7　送受信者の Process

プロトコルの通信手順と同様に，通信を妨害する役割である，侵入者の能力もモデル化する必要がある．本稿では，侵入者の能力を次のように定める．

- 情報を合成することができる（composing）.
- 情報を分解することができる（decomposing）.
- 任意の公開鍵で情報を暗号化できる（encrription）.
- 自分の秘密鍵による暗号を解読できる（decription）.

これらを図 8 のように個別にプロトコル記述言語で表し，parallel で結合したものが，侵入者の Process となる．

$Spy_1 \equiv in\ X.in\ X'.out(X,X').nil$ $(composing)$
$Spy_2 \equiv in(X,X').out\ X.out\ X'.nil$ $(decomposing)$
$Spy_3 \equiv in\ x.in\ X.out\{X\}_{Pub(x)}.nil$ $(encription)$
$Spy_4 \equiv in\ Priv(x).in\ X.out\{X\}_{Priv(x)}.nil$ $(decription)$
$Spy \equiv \|_{i\in\{1,\cdots,4\}}\ Spy_i$

図 8　侵入者の Process

通信における送信者，受信者，侵入者を別々にモデル化した $Init(A,B)$，$Resp(B)$，Spy を，図 9 のようにさらに parallel で一つの System としたものが，プロトコル記述言語による NSPK プロトコルのモデルとなる．

$P_1 \equiv \|_A\|_B!Init(A,B)$
$P_2 \equiv \|_B!Resp(B)$
$P_3 \equiv !Spy$
$NSPK \equiv P_1 \parallel P_2 \parallel P_3$

図 9　NSPK プロトコルのモデル

4.2　カラーペトリネットへの変換と検証

プロトコル記述言語でモデル化した暗号プロトコルを，変換アルゴリズムに従ってカラーペトリネットへ変換する．変換アルゴリズムでは，各関係者を表す Process ごとに関数 Trans が呼び出され，output と case の動作があると，ペトリネット上の関係者のプレイスとネットワークのプレイスの間にトランジションが追加されていく．

先に記したプロトコル記述言語で表現した NSPK プロトコルを，カラーペトリネットに変換したものを図 10 に示す．

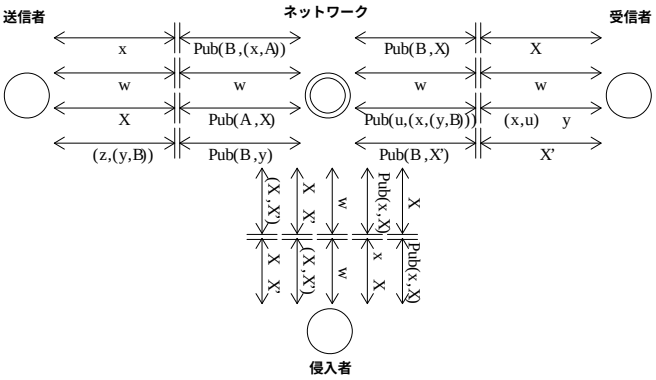


図 10　NSPK プロトコルの CPN 表現

このカラーペトリネットを使って安全性の検証を行うが，本稿では，安全性の一つである機密性の検証について述べる．機密性とは，暗号化された情報が侵入者に知られないことを保証する性質であり，プロトコル記述言語での機密性は次のように定義される．

定義 1（プロトコル記述言語における機密性） 言語で表現されたプロトコル Q が以下を満たすとき，Q は機密性を持つという．

$\forall\tilde{\alpha}\forall Q'\forall s'\forall t'\forall n\forall i.$
$(Q,\emptyset,\emptyset) \xrightarrow{\tilde{\alpha}} (Q',s',t')$ かつ $\texttt{new(n)} \sqsubset P_i$ ならば $\texttt{n} \notin t'$

ただし P_i は Q 中の送受信者を表すプロセスであり，$\tilde{\alpha}$ はアクションの系列，new(n)$\sqsubset P_i$ は P_i 中に new(n) が含まれることを表す．□

プロトコル記述言語で機密性を持っているということは，意味論で定めたアクションによるどのような計算においても，送受信者が生成したメッセージを出力することがないことである．

次に，構成されたカラーペトリネットでの機密性の定義を与える．

定義 2（CPN における機密性） カラーペトリネットの初期マーキングで送受信者のプレイスに，new name によって配置されたトークンが，初期マーキングから到達できる任意のマーキングで，ネットワークを表すプレイスに含まれないとき，カラーペトリネットは機密性を持つという．□

これは構成されたカラーペトリネットにおいて，ネットワークを表すプレイスに特定のトークンがあるようなマーキングに遷移することがあるかを調べる可達性判定問題として扱うことができ，ツールを用いて機械的に検証を行うことが可能となる．

ツールを利用して検証を行うと，NSPK プロトコルから変換して得られたカラーペトリネットは機密性を持っていることが示された．しかし，これは変換前のプロトコル記述言語においても機密性を持っていることを保証していない．これを解決するために，以下の定理を与える．

定理 2 暗号プロトコルを表現する CPN が機密性を持つならば，変換前の Q が機密性を持つ．

[証明] 対偶を示す．プロトコル Q が機密性を持っていないと仮定すると，

$\exists\tilde{\alpha}\exists Q'\exists s'\exists t'\exists n\exists i.$

$(Q,\emptyset,\emptyset) \xrightarrow{\tilde{\alpha}} (Q',s',t')$ かつ $\texttt{new(n)} \sqsubset P_i$かつ $\texttt{n} \in t'$

ここで系列 $\tilde{\alpha}$ は以下のような系列 $\tilde{\alpha}_1$，$\tilde{\alpha}_2$，$\tilde{\alpha}_3$ に分割できる．

$$\begin{aligned}(Q,\emptyset,\emptyset) &\xrightarrow{\tilde{\alpha}_1} (new(n).P_A \parallel P_B, s_1, t_1)\\ &\xrightarrow{\tilde{\alpha}_2} (out\ n.P_B \parallel P_D, s_2, t_2)\\ &\xrightarrow{\tilde{\alpha}_3} (Q', s', t')\end{aligned}$$

このとき，以下の場合のいずれかになる．

- $new(n)$ と $out\ n$ が同一のプロセス P_m に含まれる場合．

$P_m \equiv \cdots .new(n).\cdots .out\ n.\cdots .nil$

変換アルゴリズムに従うと，CPN に図 11 のようなトークンとトランジションが追加される．

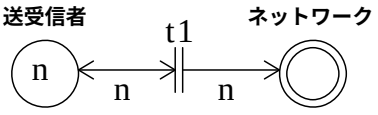


図 11 P_m によって追加される要素

この CPN ではトランジション t1 が発火し，送受信者のプレイスに配置されたトークン n が，ネットワークを表すプレイスに加えられる．

- $new(n)$ がプロセス P_n，$out\ n$ がプロセス P_o に含まれる場合．

$P_n \equiv \cdots .new(n).\cdots .out\ M.\cdots .nil$

$P_o \equiv \cdots .in\ \Pi.\cdots .out\ M'.\cdots .nil$

ただし $\exists\sigma.(\Pi[\sigma] = M$ かつ $M'[\sigma] = n)$

変換アルゴリズムに従うと，CPN に図 12 のようなトークンとトランジションが追加される．ここでトランジション t3，t4 は基本となる CPN に既に存在しているトランジションである．

この CPN では，まずトランジション t1 が発火し，続いて t4 が発火することで，関係者に M が加えられる．ここで $\Pi[\sigma] = M$ よりある代入 σ によってトランジション t2 の発火条件が満たされ，$M'[\sigma] = n$ であるので，ネットワークに n が加えられる．

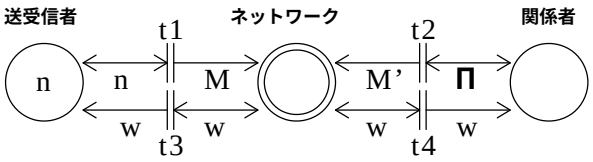


図 12 P_n，P_o によって追加される要素

よって，いずれの場合においても変換された CPN が機密性を持たないことが示された． □

上の定理により，元となる暗号プロトコルが機密性持っていることが示された．

5. ま と め

本稿では，暗号プロトコルの機密性を検証することを目的とし，プロトコル記述言語によって記述された暗号プロトコルをカラーペトリネットに変換するアルゴリズムを提案した．そして，変換で得られたカラーペトリネットが機密性を持っているならば，変換前の言語においても機密性を持っていることを示した．これらによって，プロトコル記述言語で表現された暗号プロトコルから，機械的に機密性の検証が行えるようになった．

また，機密性以外の安全性として認証性がある．認証性とは，実際の通信の相手が想定した相手に違いないこと，つまり通信でなりすましが行われていないことを保証する性質であり，本稿の手法が認証性の検証にも適用できるようにすることは，今後の課題として挙げられる．その他に，本手法ではカラーペトリネットの可達性判定が決定可能であることを利用したが，可達性判定は指数関数的な時間を要することが知られている．そこで，より多くの通信手順，複雑な暗号化を用いる暗号プロトコルに本手法を適用するためには，更に工夫を加えなければならない．

謝辞 本研究は一部，科研費#15500007，#16650005，#17700009 ならびに名古屋大学 21 世紀 COE プログラム（社会情報基盤のための音声・映像の知的統合）の補助を受けている．

文 献

[1] F. Crazzolara and G. Winskel. Language, Semantics, and Methods for Cryptographic Protocols. Technical Report, Basic Research in Computer Sciense RS–00–18, 2000.

[2] E. Mayr. An algorithm for the general Petri net reachability problem. SIAM J. Compu. 13, 3, pp. 441-46-, 1984.

[3] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. In Proceedings of the Fourth ACM Conference on Computer and Communications Security, pages 36–47, 1997.

[4] K. Jensen. Coloured Petri Nets : Basic Concepts, Analysis Methods and Practical Use Vol.1. Springer–Verlag, 1992.

[5] CPN Tools: Computer Tool for Coloured Petri Nets URL: http://wiki.daimi.au.dk/cpntools/