動的ネットワークにおける双方向匿名通信路構築手法の提案

田村 仁[†] 古原和邦[†] 今井秀樹[†]

アドホックネットワークに代表されるような動的なネットワーク上で何らかの双方向通信を行う際，送信時に利用した経路を返信時では利用できないケースは当然考慮されるべきであるが，とりわけ，返信時にその返信先を知ることができない匿名通信においては，これはそう単純な問題ではない．特に医療相談など，返信までのタイムラグが大きいアプリケーションほどそのような問題に陥る可能性は高い．しかしながら，こうした点について従来の匿名通信方式では十分に考慮されているとはいい難い．そこで本論文では，主な既存方式の特徴や問題点とその原因を整理したうえで，新たに高いデータ可用性を有した方式を提案する．また，これら提案方式を含め種々の組合せについて匿名性，データの可用性，および操作のコストという観点からの比較検証を行った．その結果，従来の代表的な双方向匿名通信方式であるオニオンルーティングを用いる場合に比べても総合的に性能が優った方式の組合せを示すことに成功した．

New Bi-directional Anonymous Routing Schemes over Dynamic Networks

Jin Tamura,[†] Kazukuni Kobara[†] and Hideki Imai[†]

In Bi-directional Communication over Dynamic Networks, the same route which had been used for sending massage might be unusable for the reply message. However, especially in Anonymous Routing, it is not easy to change the route, because even the destination of the reply message is hidden. There are many existing schemes, but no schemes are well considered about this important point. In this paper, therefore, we overview and analyze the existent schemes from this point of view, and propose new flexible anonymous routing schemes. Then we have a comparative study to find out the best combination for Bi-directional Anonymous Communication over Dynamic Networks from three points of view; anonymity, availability, and operational costs. As a result of this work, we succeed to show comprehensively good effect compare with Original Onion Routing Scheme which is the most popular anonymous routing scheme so far.

1. はじめに

1.1 背景

近年，P2P（Peer-to-Peer）をはじめ，アドホックネットワーク，ワイアレスネットワーク，モバイルネットワーク，センサネットワークなど，ネットワークのトポロジが変化する，いわゆる動的なネットワーク環境が増加している．一方で電子投票や社内告発など，匿名性を備えた通信を必要とするアプリケーションも数多く，なかでも医療相談などのように，双方向通信を必要とするものも少なくない．また動的ネットワークにおいては，中継ノードの移動や消滅などにより送信時で用いられた経路が返信時では使用できないケースは当然考慮する必要があるが，とりわけ送信元が秘匿されている匿名通信においては，送信先による柔軟な対応は困難である．

こうした点についてこれまで提案されてきた方式では十分に考慮されているとはいい難く，実際に代表的な方式である Crowds [10),11)] や Mix-net [2),3)] やオニオンルーティング[5),9)] においても，送信者が送信の際に用いた中継ノードのどの 1 つが離れたとしても，返信データが送信者までは戻ることは難しくなる．その場合，たとえばあらかじめ送信者から送信メッセージと一緒にテンポラリな公開鍵を加えておいてもらって，ブロードキャストする方法もあるが，この方法はシステムが大きくなると莫大な通信コストがかかってしまう．そこで，我々は経路変更の柔軟性に対する阻害要因であった多重暗号を用いずに，柔軟で可用性の高い返信方式として，低コストで高い柔軟性を有したスライスオニオンルーティング方式を提案した．

また一方でオニオンルーティングに比べて匿名性が

† 産業技術総合研究所情報セキュリティ研究センター
Research Center for Information Security, National Institute of Advanced Industrial Science and Technology

低くなってしまうので，まとまった数のノードのヘッダ情報に対してホップごとに共通鍵による多重暗号化を行う暗号化スライスオニオン方式を提案し，結果的に可用性を落とすことなく，匿名性を高めることに成功した．

さらに，それらの提案手法を含めた種々の組合せについて匿名性，柔軟性，およびコスト（時間，メモリ）の観点から検証を行うことで，その中での最適な組み合わせ法を示し，ひいては従来の代表的な双方向匿名通信方式であるオニオンルーティングだけを用いる場合に比べても，特に可用性とコストの面で格段に良い結果を得た．

本論文の構成は以下のとおりである．2 章では匿名通信の既存方式について述べ，各々の特徴や問題点を整理した．また 3 章では我々が提案するスライスオニオン方式および，暗号化スライスオニオン方式について述べ，4 章で比較研究を行い，最後に 5 章でまとめる．

1.2 用語の定義と仮定

- 匿名通信路
  メッセージの送信者（あるいは送受信者）を特定する情報（たとえば IP アドレスなど）を秘匿にしたまま通信することを目的とした通信路のことで，ネットワーク層やアプリケーション層によるルーティングの操作で実現される．また用途に応じて電子投票や匿名掲示板のような送信者から受信者の 1 方向の通信で秘匿性を保てばよいケースと，匿名医療相談や内部告発などのような，送受信者間の双方向の通信で秘匿性を保たなくてはならないケースとがあり，本論文では特に後者のケースにおける課題について扱う．
- 匿名性
  匿名通信路における匿名性については次の 3 つがあげられる[8]．すなわち送信者の匿名性（送信者が特定されにくいこと），受信者の匿名性（受信者が特定されにくいこと），および送受信者のつながりの匿名性（送信者と受信者とのつながりが特定されにくいこと）である．
- ノード
  本論文では匿名通信を行う特別なルータ（匿名通信ノード）のことを単にノードと呼ぶ．オニオンルータや Mix ノードなどもこれにあたる．各々が MAC アドレスや通信中の IP アドレスのようなユニークな ID を持つ．
- 送信元/送信先
  本論文では，送信者/受信者のノード（あるいはサーバ）に対して送信元/送信先と呼ぶことにする．また，送信元/送信先の ID が特定されてしまうことがすなわち送信者/受信者の匿名性が破られてしまうことと考える．
- 攻撃者
  攻撃者には外部攻撃者と内部攻撃者が存在し，その目的は匿名通信路の匿名性を破ることである．外部攻撃者は盗聴者ともいい，彼らにとって可能であることといえばせいぜいトラフィックに流れるデータの大きさやノードへの出入りのタイミングなどを計測することであり，それらの攻撃から防ぐ手だてはさほど難しくはない．内部攻撃者とは匿名通信ノードの所有者，または何らかの方法を用いてノードを流れるデータを収集することができる者であり，彼らはデータやヘッダ情報，そして前後のノードまで知ることができ，当然外部攻撃者よりも強力な攻撃者である．受信者が送信者の匿名性を破ろうとしている場合も内部攻撃者であり，この場合は送信者からのメッセージも知ることができる．本論文では，より強力な攻撃者である内部攻撃者に対する匿名性を目指しているため，以下で単に攻撃者という場合は内部攻撃者をさす．
- 送信者/受信者を特定できる環境
  近年増えてきている NAT 環境や，ファイアウォールを設置した環境においては，ネットワーク管理者にはネットワーク内のユーザのパケットの出入りや，使用されているポート番号（ひいては使用中のアプリケーション）などの観測が可能である．さらに，ほとんどの匿名通信方式ではノード間のホッピングにより匿名性を満たしており，その場合，ネットワーク管理者からはネットワーク内にいる送信者を特定できてしまう．また，匿名通信路のユーザは匿名通信ノードを立ち上げない場合，最初のノードは送信者を，最後のノードは受信者を特定することが可能である．本論文ではこうした環境を，送信者/受信者を特定できる環境と呼ぶ．

2. 匿名通信の従来研究

匿名通信路の研究は 1981 年に Chaum が提案した Mix-net[3] から始まる．彼はその提案方式の中で，メッセージを複数の中継ノード間を経由させ，そのルーティング情報を各中継ノードの公開鍵による多重暗号化を用いることによって隠蔽する方式を示し，その後 Mix-net をベースに双方向・実時間での通信が行えるよう

に変更されたオニオンルーティングなどが Reed らによって提案され[9]，現在でも多重暗号化が匿名通信方式の主流となっている．

また多重暗号化を用いない代表的な方式として，1998 年に Reiter らによって，Web 閲覧などへの利用を前提としている Crowds が提案されている[10]．また Chaum は 1988 年に DC-net という，情報量的に安全な匿名通信を行う方式も示しており[4]，現在もいくつかの研究が続けられているが，システムが大きくなるにつれ計算コストがかかりすぎてしまうため実現は難しい．そこで本章では Mix-net，オニオンルーティング，および Crowds について概説する．

2.1 Mix-net

Mix-net は，複数の入力メッセージをその順序を入れ替えて出力することにより，外部からは入力と出力の関連がつけられないようにした方式である．ある決められた数のメッセージを受け取ったノードは，それらを各自の秘密鍵により復号したうえで，その順番をランダムに並べ替えて，指定された次の Mix ノードに出力する．すなわち，Mix-net は以下のような処理により，攻撃者によるトラフィック分析攻撃に対抗する．

(1) メッセージは，匿名通信用ノード（以下，Mix ノード）$N_1, N_2, \ldots, N_m$ の順に送られるものとする．メッセージ送信者は，まずメッセージをノード N_m の公開鍵で暗号化し，次にこの暗号化したものをノード N_{m-1} の公開鍵で暗号化する．これを繰り返し，最終的にはノード N_1 の公開鍵で暗号化する．

(2) それぞれの Mix ノードは，このメッセージをある個数受け取ったときに，まずそれらを復号し，順番をランダムに並べ替え，次のノードに送信する．

ネットワーク上のそれぞれの Mix ノードは，あるメッセージが流れてきたときに，メッセージ転送経路の情報としては 1 つ前のノードとすぐ次のノードについての情報しか知ることができない．そのため，Mix ノードが信頼できる限りは，たとえ Mix-net の構成ノードが 1 つであっても，送信者と受信者のつながりを調べることは困難である．さらに一般的には，Mix-net として鎖状につながれた複数の Mix ノードを用いる．この場合には，個々の Mix ノードは最悪でも，メッセージの送信者もしくは受信者のどちらかを知ることしかできず，送信者と受信者のつながりは見破られない．

ここで**図 1** に，送信者 S が Mix ノード {N1, N2, N3} で構成された Mix-net を利用して受信者 R にメッセージを送信する様子を示しておく．同図では，まず S は N3→N2→N1 の順で，R へのメッセージ M を多重に暗号化する．そして，それをはじめの N1 に送る．N1 は決められた個数（同図では 3 個）のメッセージを受け取るまで待ち，各メッセージを自分の鍵で復号する．そしてさらにそれらを攪拌したうえで次のノード N2 に送信する．N2 も同様に，各メッセージを自分の鍵で復号した後に攪拌し，次ノード N3 に送信する．N3 もやはり自分の鍵で各メッセージを復号して，メッセージ M の送信先が R であることを読み取り，R に M を送信する．

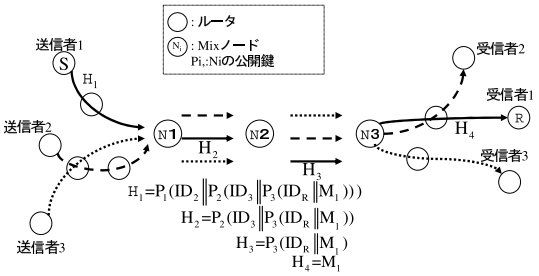

図 1 多重暗号を利用した Mix-net
Fig. 1 Mix-net scheme.

この結果，すべての Mix ノードが結託をしない限り送信者とメッセージを結びつけることは不可能となり，送信者や受信者の匿名性が保たれる．

2.2 オニオンルーティング

オニオンルーティングでは，Mix-net の多重暗号化の仕組みをベースに，実時間での双方向通信が行えるように，以下のような 2 点で変更が加えられ，効率性を高めている．

- *I*. 経路作成，*II*. データ送受信，*III*. 経路破棄という大きく 3 つのフェーズを持ち，データの送受信には共通鍵暗号を用いた多重暗号を利用する．
- 遅延を小さくするため，各ノードにおけるデータの送信順序の入れ替えは原則行わず，代わりにダミーのデータを送信することで，タイミングによるデータの入出力のリンクを防止する．

経路作成フェーズにおいて，送信元は，送信先までの各匿名通信用ノード（オニオンルータとも呼ばれる．以下，ノード）の送信用と返信用の共通鍵を 1 組ずつ生成し，多重暗号化を利用して各ノードと共有する．こうして経路が確立された後のデータ送受信フェーズでは，公開鍵暗号に比べて圧倒的に処理速度が速い共通鍵（送信用）を用いてメッセージを多重暗号化し，送信先まで各ノードで復号しながら届けられる．

また，返信では各ノードの返信用共通鍵を用いて暗

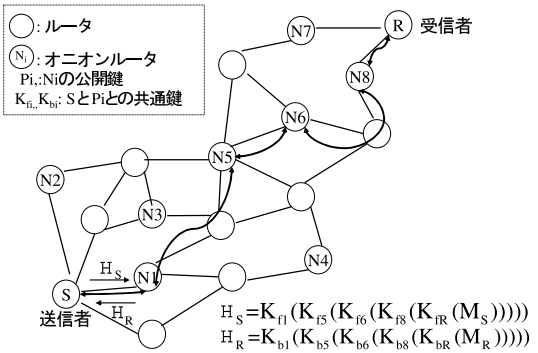

図 2 オニオンルーティング方式と送受信のヘッダの形
Fig. 2 Onion Routing scheme and its headers.

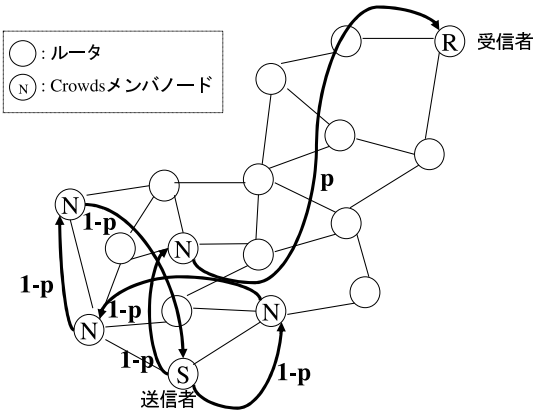

図 3 Crowds における確率 p のマルチホップ
Fig. 3 Multi-hops with probability p in Crowds.

号化されながら送信元まで届けられることにより，より高速な送受信を実現している．また，経路上のノードが抜ける際や，前後のノードと通信ができなくなった場合，そのノードは経路破棄用のコマンドをその経路の前後のノードに送信し，各ノードも自分のテーブルから対応する情報を削除し，経路破棄フェーズが完了する．

図 2 に，オニオンルーティングの送受信フェーズで用いられる多重暗号化の様子を示す．ここで，各中継ノード N_1，N_5，N_6，N_8 および送信先（受信者）R は経路作成フェーズにおいてすでに送信者 S と共通鍵の組 K_{fi}，K_{bi} の共有を完了している．

2.3 Crowds

Crowds では，ユーザはまず Crowds メンバ（この場合の匿名通信用ノードとなる）に加入しメンバリストを受け取る．データを送る際には送信先を記述したデータを Crowds メンバに送る．受け取ったメンバは確率 p で送信先に投げ，確率 1 – p で他の Crowds メンバに投げる．この処理を繰り返すことで，データは期待値 $1/p$ 人の Crowds メンバを経由して送信先に届けられることになる．送信先のアドレスは Crowds メンバには秘匿されないが，データの中身は送信先の公開鍵で暗号化することで秘匿できる．返信は，中継したデータの ID を各 Crowds メンバが記憶しておくことで行える．また，Crowds メンバが多くなければ送信者から送られてきたテンポラリの公開鍵によりデータを暗号化し，それをメンバ全員にブロードキャスト（あるいはマルチキャスト）することで返信を実現することも可能である．ここで**図 3** に確率 p で，Crowds メンバノードを数回ホップしながら送信先に届くまでを示す．

2.4 問 題 点

Mix-net やオニオンルーティングなど，データの多重暗号化を利用する方式は，中継ノードの 1 つでも正規ノードを含んでいれば匿名性を破られない高い匿名性を持つものの，送信フェーズ/返信フェーズともに柔軟性が低く，仮に送信フェーズであれば再度経路作成フェーズからやり直して再度送信する方法がとれるが，返信フェーズにおいては，ネットワークの動的な度合いや受信から返信まで要する時間（つまりアプリケーション）によっては送信経路が使用できない可能性が非常に高い．しかしながら，返信先が返信元（受信者）からは分からないため，この場合は Crowds の場合同様，送信者から送られてきたテンポラリの公開鍵を用いてブロードキャストをする手段しか残されていないが，この場合通信コストが非常に大きい．また暗号化処理を用いていない Crowds では，コストも少なく，送信時には可用性も高いものの，やはり返信時には上記と同様な問題が起こる．

3. 提 案 方 式

1.2 節であげた 3 通りの匿名性はそれぞれ独立した要件ではなく，送受信者のつながりの匿名性は，送信者の匿名性あるいは受信者の匿名性のどちらかを満たすならば満たされる，という関係にある（送信者の匿名性と受信者の匿名性は独立した要件）．この送受信者のつながりの匿名性は匿名通信方式が最低限満たすべき要件であるものの，Crows は，送信者の匿名性のみを満たすことによって送受信者のつながりの匿名性を満たす方式だが，NAT やファイアウォールなど，ネットワーク管理者が送信者を特定できる環境においては送受信者のつながりの匿名性を満たすことはできない．一方，Mix-net やオニオンルーティング方式は，送信者の匿名性と受信者の匿名性を同時に満たすが，多重暗号化を利用しているため可用性が低い．そこで

本章では，送信者の匿名性と受信者の匿名性を大きく損なうことなく，可用性を向上させた方式を提案する．

3.1 方式の要件

動的ネットワークにおける双方向通信のための匿名通信路として，以下の要件を満たすような方式を目標とする．

(1) データの可用性（=ルーティングの柔軟性）
動的ネットワークの中で送信データが受信者まで届き，また返信データが送信者まで返るためには，ルーティングの柔軟性が不可欠である．

(2) 匿名性
送信者/受信者を特定できる環境において，内部攻撃者の結託攻撃に対して，送受信者のつながりの匿名性を満たすことを目指す．ただし，攻撃者として中継ノードがすべて結託して匿名性を満たす方式などは存在せず，どこまでの結託に耐えられるかをすなわち匿名性の高さと考える．

(3) 低コスト（メモリサイズ/計算コスト）
動的ネットワークでは，デスクトップなどに比べて計算能力やメモリの小さい端末が用いられることが多い．そのため，通信で送受信者およびそれぞれのノードにかかる負荷はより小さいことが要求される．そのためにもヘッダサイズやデータサイズ，暗号/復号処理にかかるコストの削減が不可欠である．

3.2 記　　法

A ‖ B：文字列 A と文字列 B を結合したもの

N_S：送信元（送信者ノード）

N_R：送信先（受信者ノード）

N_i：ある経路上における i ホップ目☆のノード（ユニーク ID：ID_i，$1 \leq i \leq h$）

N_{ij}：ホップ数を固定したネットワーク環境☆☆における i ホップ目のグループ（以下，第 i ホップグループと呼ぶ）の j 番目のノード（ユニーク ID：ID_{ij}，$1 \leq j \leq k_i$；k_i は第 i ホップグループのノード数）

P_i，S_i：第 i ホップグループが共有する公開鍵ペア（N_S が生成）

H_i：ノード N_i の有するスライス形式のヘッダ情報（主に前後ノードの ID 情報）

$\mathbf{K}_i(X)$：文字列 X をノード N_i の共通鍵 K_i で暗号化したもの

$\mathbf{P}_i(X)$：文字列 X をノード N_i の公開鍵 P_i で暗号化したもの

M_S：送信メッセージ（送信者が生成）

M_R：返信メッセージ（受信者が生成）

3.3 スライスオニオン方式

3.3.1 経路探索フェーズ

step1：送信者はテンポラリの公開鍵のペア P_t，S_t を生成し，受信ノード N_R の探索要求データ $Query_R = ID_R \parallel P_t$ をブロードキャストする（このとき，送信アドレスのフィールドはブランクにすることによって匿名性を保つ）．

step2：各ノードはデータの中の ID を参照し，自己の ID と異なる場合，あらかじめ定められた制限ホップ数以内であれば再びブロードキャストをし，制限回数に達した場合はその場で破棄をする（ネットワークのオーバフローの防止のため）．

step3：N_R は，ID_R との照合で一致していることを確かめたら，自分が受け取ったすべての経路に対して応答データ

$$Answer_R = \mathbf{P}_t(P_R, ID_R, Null)$$

を送信（返信）する．

step4：N_R から $Answer_R$ を受け取った N_i は $Answer_R$ を

$$Answer_R = \mathbf{P}_t(P_i, ID_i, Answer_R)$$

のように更新し，$Query_R$ を受け取ったすべてのノードに $Answer_R$ を送信する．

step5：同様にして，すべての中継ノードでは，あらかじめ定められた制限ホップ数まで応答データ $Answer_R$ の更新と手前の中継ノードへの送信を繰り返し，制限ホップ数で送信者まで到達しない応答データについては破棄する．

step6：送信者 N_S は順次テンポラリの秘密鍵 S_t を用いて復号することにより N_R までの経路および各中継ノードの公開鍵と ID の組を得る（ここで，ネットワークの帯域に余裕がある場合には，N_R から始まる応答データについても，ホップ数制限付きのブロードキャストを用いることによって送信者がさらにより多くの経路を得られるる可能性がある）．

3.3.2 スライスオニオンヘッダ生成およびメッセージ連結フェーズ（送信者）

step1：まず送信者は経路探索フェーズによって入手した中継ノードのネットワークトポロジをもとに送信者から受信者までの各中継ノード N_i のヘッダ情報を以下のように構成する．

$$H_i = H_{f_i} \parallel H_{b_i}$$

☆ 匿名通信ノードを経由したホップ数のことである．

☆☆ ここでは，送信者から受信者までにホップする回数が一定になるように構成したネットワークのことを指すことにする（図 5 参照）．

ここで，H_{f_i} とは送信者から受信者に向けた通信における N_i の次のホップ先の ID 情報の一覧であり，また H_{b_i} とは，受信者から送信者に向けた通信（返信フェーズ）における N_i の次のホップ先の ID 情報の一覧である．

step2：各ヘッダ H_i を各公開鍵 P_i で暗号化したうえで各々連結させたものをスライスオニオンヘッダ H_{SO} とする．つまり，経路探索フェーズで入手した送信者までのすべての中継ノードの集合を $\mathbf{N}$ とおくと，

for all $i \in \mathbf{N}$ do[

$$H_{SO} = H_{SO} \parallel \mathbf{P}_i(H_i)$$

]

そして受信者ノード N_R のヘッダ情報を連結してヘッダ部分が完成する．

$$H_{SO} = H_{SO} \parallel \mathbf{P}_R(H_R, K_R)$$

ここで，K_R は，データ部分を復号するための共通鍵である．

step3：送信者は受信者へのメッセージ M_S を共通鍵 K_R で暗号化し，これをメッセージ部分としてstep2 で生成したヘッダ部分 H_{SO} と連結させたものをスライスオニオンデータ $Data_{SO}$ とする．

$$Data_{SO} = H_{SO} \parallel \mathbf{K}_R(M_S)$$

3.3.3 送信フェーズ

step1：送信者は $Data_{SO}$ を送信者用のホップ先の ID 情報 H_S の中の送信可能なノードに送信する．

step2：$Data_{SO}$ を受け取った中継ノード N_i（$N_i \in \mathbf{N}$）は各公開鍵 P_i に対する秘密鍵 S_i を用いて受け取ったデータ $Data_{SO}$ の中の，各自に割り当てられたヘッダ H_i の送信用ホップ先一覧 H_{f_i} を参照する．

step3-1：もし H_{f_i} に送信可能であるノードが存在していなければ，1 つ手前のノードに戻し，タグ **non-flip** を一時的に **flip** に変えて，迂回であることを知らせる．手前のノードはタグを **non-flip** に戻し，Step2 に戻り別の送信可能なノードへの送信を試みる．

step3-2：もし H_{f_i} に別の送信可能であるノードが存在するならばそのノード送信し，送信したノードが受信者ノードでない場合には Step2 に戻る．

step4：受信者は受け取った $Data_{SO}$ の中のヘッダ部分から得た共通鍵 K_R を用いてメッセージ部分 $\mathbf{K}_R(M_S)$ を復号し，メッセージ M_S を取り出す．

3.3.4 返信フェーズ

返信フェーズも送信フェーズとほぼ逆の操作で進める（以下返信フェーズでは，送信フェーズにおける受信者を返信者と呼び，送信者を返信先と呼ぶことにする）．

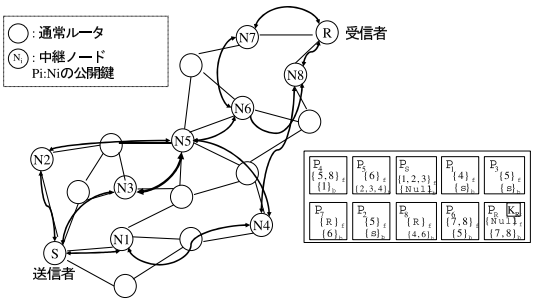

図 4 つねに同じヘッダ H_{SO} のまま送受信が可能なスライスオニオンルーティング
Fig. 4 Sliced Onion Routing scheme.

step0：返信者は返信メッセージ M_R を生成し，共通鍵 K_R を用いて暗号化し，$\mathbf{K}_R(M_R)$ をメッセージ部分としてスライスオニオンのヘッダ部分 H_{SO} と連結させる．つまり，

$$Data_{SO} = H_{SO} \parallel \mathbf{K}_R(M_R)$$

step1：返信者は $Data_{SO}$ をホップ先一覧 H_R 中の送信可能なノードに送信する．

step2：$Data_{SO}$ を受け取った中継ノード N_i は S_i を用いて受け取ったデータ $Data_{SO}$ の中の，各自に割り当てられたヘッダ H_i の返信用ホップ先一覧 H_{b_i} を参照する．

step3-1：もし H_{b_i} に送信可能であるノードが存在していなければ，1 つ手前のノードに戻し，**non-flip** タグを一時的に **flip** に変えて，迂回であることを知らせる．手前のノードはタグを **non-flip** に戻し，Step2 に戻り別の送信可能なノードへの送信を試みる．

step3-2：もし H_{b_i} に送信可能であるノードが存在するならば送信し，送信したノードが返信先ノードでなければ Step2 に戻る．

step4：返信先（送信者）は共通鍵 K_R を用いて，受け取った $Data_{SO}$ のメッセージ部分 $\mathbf{K}_R(M_R)$ を復号し，メッセージ M_R を取り出す．

ここで，**図 4** にスライスオニオンの送信/返信フェーズの様子を図で示す．オニオンとは異なり，スライスオニオンのヘッダの形 H_{SO} はつねに変わらない．

3.4 暗号化スライスオニオン方式

オニオンルーティング方式において返信の可用性を高くするためには，送信者があらかじめ返信用のルートを経路上のノードに順次知らせるための多重暗号文を複数個作成しメッセージの部分に加える方法があるが，スライスオニオン方式ではこの方法に比べ格段に

小さなヘッダサイズで同等の可用性を実現させている．しかしながら，スライスオニオンヘッダの形はつねに変わらないため，攻撃者同士が結託することによって（あるいは複数のノードを支配する強力な攻撃者によって）リンクがとられやすく，送受信者の匿名性がオニオンルーティング方式よりもきわめて低くなってしまう（詳しくは次章の比較検討参照）．そこで我々は，スライスオニオンヘッダの中を，ホップグループごとに共有させた共通鍵で多重に暗号化させることにより，可用性を落とさずに匿名性を高める暗号化スライスオニオン方式を以下で提案する．各フェーズはスライスオニオン方式と重複もあるため，各フェーズの差異の分かりにくい部分は言葉で付け加えることにする．

3.4.1 経路探索フェーズ

スライスオニオン方式における経路探索フェーズと同様（3.3.1 項参照）．

3.4.2 暗号化スライスオニオンヘッダ生成およびメッセージ部分連結フェーズ（送信者）

暗号化スライスオニオン方式では，送信者は経路探索フェーズによって入手した中継ノードのネットワークトポロジをもとに，送信者から受信者までのホップ数（1 ホップ〜h ホップ）ごとのホップグループに分けられるように中継ノードを選択する．

step0：送信者は各ホップグループごとで共有する送信用･受信用の共通鍵 K_{f_i} および K_{b_i}（$1 \leq i \leq h$）を生成する（計 $2h$ 個）．

step1：各経路上のノード N_{ij} 用のヘッダ H_{ij} を次のように作成する．

$$H_{ij} = H_{f_{ij}} \parallel H_{b_{ij}} \parallel K_{f_i} \parallel K_{b_i}$$
$$= ID_{i+1,1}, \cdots, ID_{i+1,|H_{f_{ij}}|}$$
$$\parallel ID_{i-1,1}, \cdots, ID_{i-1,|H_{b_{ij}}|} \parallel K_{f_i} \parallel K_{b_i}$$

step2：各ヘッダ H_{ij} を各ノード N_{ij} の公開鍵 P_{ij} で暗号化する．さらに受信者のヘッダ $\mathbf{P}_R(H_R, K_R)$ を共通鍵 K_{f_h} で暗号化したものを C_h と置き，以下の処理を繰り返す．

for $i = h$ to 1
do[
for $j = 1$ to $\mid H_{f_{ij}} \mid$
do[
$C_i = C_i \parallel \mathbf{P}_{ij}(H_{ij})$
$j = j + 1$;]
$C_i = \mathbf{K}_{f_{i-1}}(C_i)$
$i = i - 1$;]

こうして出力された C_1 に $\mathbf{P}_{1,j}(H_{1,j}, K_{f_1})$ を連結させたものを暗号化スライスオニオンヘッダ H_{SOE} とする．すなわち，

$$H_{SOE} = P_{1,j}(H_{1,j}, K_1) \parallel C_i$$

step3：送信者は受信者へのメッセージ M_S を共通鍵 K_R で暗号化する．さらに共通鍵 K_{f_h} から K_{f_1} を用い多重暗号化し，step2 で生成した暗号化スライスオニオンヘッダと連結させる．つまり，送信データ $Data_{SOE}$ はヘッダ部分 H_{SOE} およびメッセージ部分 M_{SOE} からなる．

$$Data_{SOE} = H_{SOE} \parallel M_{SOE}$$
$$M_{SOE} = \mathbf{K}_{f_1}(\mathbf{K}_{f_2}(\cdots \mathbf{K}_{f_h}(\mathbf{K}_R(M_S))))$$

3.4.3 送信フェーズ

step1：送信者は $Data_{SOE}$ を 1 ホップ目の中のノード $N_{1,j}$ に送信する．

step2：（$1 \leq i \leq h-1$ の間）$N_{i,j}$ は公開鍵 $P_{i,j}$ に対する秘密鍵 $S_{i,j}$ を用いて受け取ったデータ $Data_{SOE}$ の中の，自分に割り当てられたヘッダ H_{ij} を参照する．

step3-1：もしも $H_{f_{ij}}$ に送信可能であるノードが存在していなければ，1 つ手前のノードに戻し，タグ **non-flip** を一時的に **flip** に変えて，迂回であることを知らせる．手前のノードはタグを **non-flip** に戻し，Step2 に戻り別の送信可能なノードへの送信を試みる．

step3-2：もしも $H_{f_{ij}}$ に送信可能であるノードが存在し，ここで，ヘッダ部分の手前までのすべてのホップグループのヘッダ情報を K_{b_i} で暗号化し，以降のヘッダ部分およびメッセージ部分を K_{f_i} で復号したうえで $Data_{SOE}$ を送信する．$i = i + 1$ として Step2 に戻る．

step4：$N_{h,j}$ は受け取った $Data_{SOE}$ のヘッダ部分の手前までのすべてのホップグループのヘッダ情報を K_{b_h} で暗号化し，メッセージ部分を K_{f_h} で復号し，受信者に送信する．

step5：受信者は受け取った $Data_{SOE}$ のメッセージ部分 $\mathbf{K}_R(M_S)$ を共通鍵 K_R を用いて復号し，メッセージ M_S を取り出す．

3.4.4 返信フェーズ

暗号化スライスオニオン方式における返信フェーズもやはり送信フェーズとほぼ逆の操作で進める．ここで，$Data_{SOE}$ は今度は第 1 ホップグループの各々の公開鍵暗号化されたヘッダ H_{1j}（$1 \leq j \leq k_i$）が一番奥に格納されている形で K_i で順番に多重暗号化されている点に注意されたい．

step0：返信者は返信メッセージ M_S を生成し，共通鍵 K_R を用いて暗号化し，$\mathbf{K}_R(M_R)$ をメッセージ部分としてスライスオニオンヘッダと連結させる．

$$Data_{SOE} = H_{SOE} \parallel \mathbf{K}_R(M_R)$$

step1：返信者は $Data_{SOE}$ を H_R 中のノード $N_{h,j}$ に送信する．

step2：（$2 \leq i \leq h$ の間）$N_{i,j}$ は $S_{i,j}$ を用いて受け取ったデータ $Data_{SOE}$ の中の，自分に割り当てられたヘッダ $H_{b_{1,j}}$ を参照する．

step3-1：もし H_{b_i} に送信可能であるノードが存在していなければ，1 つ手前のノードに戻し，**non-flip** タグを一時的に **flip** に変えて，迂回であることを知らせる．手前のノードはタグを **non-flip** に戻し，Step2 に戻り別の送信可能なノードへの送信を試みる．

step3-2：もし H_{b_i} に送信可能であるノードが存在するならば，ヘッダ部分の手前までホップグループのすべてのヘッダ情報を K_{f_i} で暗号化し，以降のヘッダ情報は K_{b_i} で復号したうえで $Data_{SOE}$ を送信する．$i = i - 1$ として Step2 に戻る．

step4：$N_{1,j}$ は受け取った $Data_{SOE}$ のヘッダ部分の手前までのすべてのホップグループのヘッダ情報を K_{b_1} で暗号化し，メッセージ部分を K_{f_1} で復号し，返信先に送信する．

step5：返信先は受け取った $Data_{SOE}$ のメッセージ部分 $\mathbf{K}_R(M_R)$ を共通鍵 K_R を用いて復号し，メッセージ M_R を取り出す．

4. 比較検討

ここでは，我々の提案手法であるスライスオニオン方式と暗号化スライスオニオン方式，そして実時間双方向の匿名通信方式として代表的なオニオンルーティング方式の 3 方式を用いて，送信フェーズと返信フェーズとで種々の組合せにより，匿名性，可用性（つまり，ネットワークの動的変化に対する柔軟性），およびコストの観点から比較検討を行った．

4.1 比較環境

比較検討を進めていくにあたり，結果が見えにくくなるのを防ぐため，比較するネットワークなどの環境を以下のように設定した．

(1) 送受信者間のホップ数および，ホップグループのノード数を固定し，$h+1$ ホップ，1 ホップグループあたり k ノードとする（**図 5** 参照）．つまり，中継ノードの個数は hk 個．

(2) 動的ネットワークとして，送信時と返信時において各ノードのオンライン/オフラインの状態が変化すると仮定する．また各々のノードがオンライン状態で，可用である確率を a（$0 \leq a \leq 1$）とし，送信時と返信時でトポロジは完全独立事象として各ノードが変化する．

(3) ネットワーク上のすべてのノード数を n 個とし，その中で最も支配ノードが多い攻撃者の支配ノードの数は m 個であるとする．

(4) 送信者は hk 個の中継ノードを選ぶ際，n 個の中から無作為に hk 個のノードを選択する．どこに攻撃者から支配されたノードが存在したとしてもこのノード選択には影響しない．

(5) 匿名性は，最も支配ノード数が多い攻撃者が送信者と受信者を特定できてしまう確率と定義する．

(6) トレードオフの比較をよりクリアにするため，すべての方式で使用できるヘッダのサイズを制限し，統一する．本比較では，すべての中継ノードを使用する際のサイズに合わせ，$hk^2\,|k|$ とした．

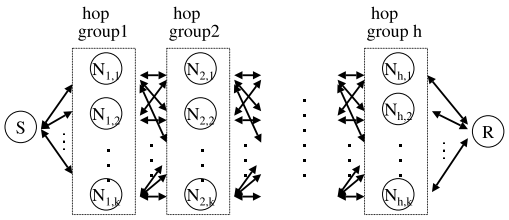

図 5 ホップ数を固定した比較環境
Fig. 5 Comparative environment with fixed hops.

表 1 種々の送信/返信方式の組合せによる比較結果
Table 1 Comparative results of several combinations.

送信方式	返信方式	匿名性	返信時の可用性	返信時のステップ数
オニオン	オニオン	○ $\{1-(m/n)^h\}^k$	× $1-(1-a^h)^k$	× $h+1+h^2/(1-a)$
スライス	スライス	× $1-[1-\{(n-m)/n\}^k]^3$	○ $\{1-(1-a)^k\}^h$	○ $h+1+2h/(1-a)$
暗号化スライス	暗号化スライス	△ $1-[1-\{(n-m)/n\}^k]^{h/2}$	○ $\{1-(1-a)^k\}^h$	○ $h+1+2h/(1-a)$
オニオン	スライス	× $1-[1-\{(n-m)/n\}^k]^3$	○ $\{1-(1-a)^k\}^h$	○ $h+1+2h/(1-a)$
オニオン	暗号化スライス	○ $1-[1-\{(n-m)/n\}^k]^h$	○ $\{1-(1-a)^k\}^h$	○ $h+1+2h/(1-a)$

4.2 比較結果および考察

以上の設定環境で，匿名性，可用性および返信までに要する時間コストの目安の値として比較した結果を**表 1** にまとめた．

ここでオニオン，スライス，暗号化スライスとはそれぞれ，オニオンルーティング方式，スライスオニオン方式，暗号化スライス方式のことである．まず，送信/受信ともに各々の方式のみで行った 3 つと，より

経路探索フェーズとも状態の差異の少ない送信時に，匿名性の高いオニオンルーティング方式を用いて，返信用としても可用性が落ちないスライスオニオン方式および暗号化スライス方式とを組み合わせた，計 5 種類の中で比較をした．

まずすべての方式でヘッダサイズをそろえた場合，オニオン方式の返信用のヘッダは，返信先まで h 個の匿名ノードを経由したものを k 経路（つまり k 種類の h 重暗号ヘッダ）作成することができる．匿名性については，送信者により任意に選択された中継ノードの中の悪意のあるノードがすべて結託したとしても送受信者のつながりを特定できない確率を求めた．たとえばオニオン方式では，k 経路のうち 1 つでも経路上の中継ノードすべてが結託していない限り特定ができないので $[1-(m/n)^h]^k$ となり，またスライスオニオン方式では第 1 ホップグループおよび第 h ホップグループの中にそれぞれ少なくとも 1 つ悪意のあるノードが含まれているときに特定できるので $1-[1-[(n-m)/n]^k]^2$ となり，暗号化スライスオニオン方式では，各ホップグループすべてに少なくとも 1 つは悪意のあるノードが含まれているときに特定できるので $1-[1-[(n-m)/n]^k]^h$ となる．

また返信時の可用性については，オニオン方式では各ルート（h 個）の匿名ノードのうち 1 個でも消滅していると，そのヘッダで返信先まで届けることはできないので，1 ルートあたりの可用性は a^h であり，k ルートのうち 1 ルートでも可用であればよいのでこの方式の返信可用性は $1-(1-a^h)^k$ となる（つまり k ルートとも可用でない確率を 1 から引く）．他の方式（スライス方式や暗号スライス方式）では，どれかのホップグループのノードがすべて消滅していない限りは可用であるので可用性は $[1-(1-a)^k]^h$ となる．

また返信時のステップ数については，どのやり方でも基本的にかかる $h+1$ ホップに加え，オニオン方式では消滅しているノードにあたるたびに返信者まで戻って新しい多重暗号化オニオンヘッダを参照しにいくことになるので $h+1+(h^2)/(1-a)$．他の方式（スライス方式や暗号スライス方式）では，消滅しているノードにあたってもその 1 つ手前のノードから送信可能ノードをあたりつづけることができるので $h+1+2h/(1-a)$ となる．

ここで各中継ノードの可用性 a を変化させた際の各返信方式の可用性の変化や，全体の中継ノードに対する悪意のあるノードの比率（m/n）を変化させた際の各方式の匿名性の変化について図示した（**図 6**，**図 7**）．

通常各匿名ノード間にはルータなど様々なノードが経由されており，その間に運用ポリシの異なるネットワークをまたぐこともあるが，本シミュレーションにおいてはそうしたいかなる要因についても加味された可用率を a としている．つまり匿名ノード N_i が可用であるということは，N_i 自身が消滅しているかどうかだけではなく，その前の匿名ノードから N_i に至るまでの経路が可用であるかどうかも含まれている．よって経路が複雑になるにつれて a の値は低くなることが考えられる．各返信方式の可用性については，$a=0.8$ あたりで提案方式とオニオンルーティング方式との間で一番大きな開きが生じていることなどから（図 6 参照），ある程度までの経路の複雑さによる各ノードの可用性の低下に対しても提案方式は高い可用性を保っていることを示している．

結果として，送信としてオニオンルーティング方式，返信として暗号化スライス方式の組合せ（以下，オニ

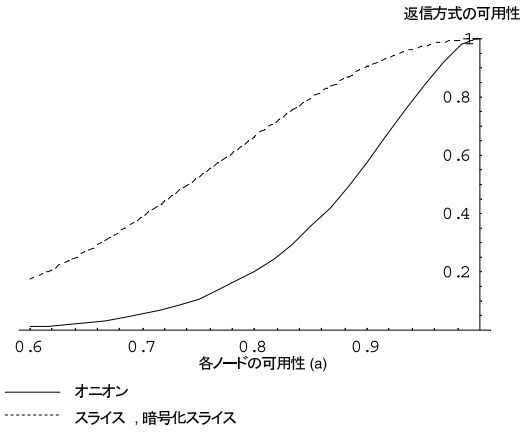

図 6 各中継ノードの可用性に対する各返信方式の可用性の変化
Fig. 6 Changes of availability.

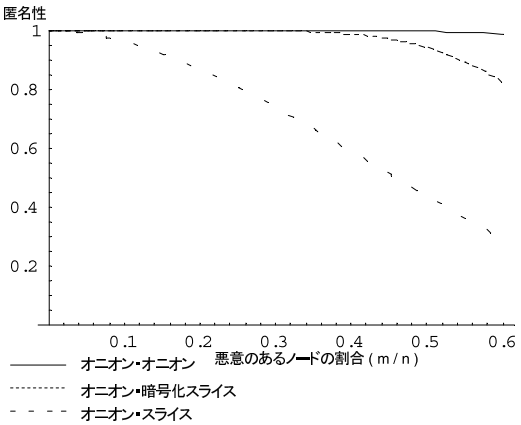

図 7 悪意のあるノードの割合に対する各方式の匿名性の変化
Fig. 7 Changes of anonymity.

オン/暗号化スライス方式と呼ぶ）が最もバランスが良く，送信/返信ともにオニオンルーティング方式を用いるのに対して匿名性の点のみで劣っている．ここで匿名性の差は，悪意のあるノードの全体との比（つまり m/n）によっても左右され，割合が高ければ高いほど差が大きくなり，低いほど差が小さくなる．しかしながら，近年の菊池らの研究によると，インターネット上での不正ホスト（故意であるか否かにかかわらず，不正なパケットの発信源になっているホストのこと）の数は現在使用されているグローバル IP アドレスの 0.0064%（約 15,000 台に 1 台）にすぎないという結果も報告されている[7]．これがそのまま攻撃者の匿名通信ノードの支配率にあてはめられるかどうかはまだまだ検討の余地はあるものの，m/n がきわめて低いことを想定するならば，オニオン/暗号化スライス方式の組合せはオニオンルーティング方式に比べ，唯一劣っている匿名性についてもほとんど遜色はないと我々は考える（図 7 参照）．

5. ま と め

本論文では，動的ネットワーク上の双方向性匿名通信におけるデータの可用性の問題について扱い，特に既存方式では解決できていなかった返信フェーズにおける経路確保の難しさを解決する方式を提案した．また，送受信者の匿名性，データの可用性，処理コストの点から種々の組合せで比較を行った結果，オニオンルーティング方式の送信フェーズと組み合わせたオニオン/暗号化スライス方式が，従来のオニオンルーティング方式のみを用いるよりもおおかた性能が高い方式となっていることを示すことができた．

参 考 文 献

1) Bleumer, G.: DC network (2004). available at http://www.francotyp.com/research/bleumer/EncInfSec/GBl.DCNetwork.pdf
2) Berthold, O., Federrath, H. and Kopsell, S.: Web MIXes: A system for anonymous and unobservable internet access, *Anonymity 2000*, Lecture Notes in Computer Science, Vol.2009, pp.115–129, Springer-Verlag (2000).
3) Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms, *Comm. ACM*, Vol.24, No.2, pp.84–88 (1981).
4) Chaum, D.: The dining cryptographers problem, *Journal of Cryptology*, Vol.1, No.1, pp.65–75 (1988).
5) Goldschlag, D., Reed, M. and Syverson, P.: Onion routing for anonymous and private internet connections, *Comm. ACM*, Vol.42, No.2, pp.39–41 (1999).
6) Gomulkiewiez, M., Klonowski, M. and Kutylowski, M.: Onion Based on Universal Re-Encryption-Anonymous Communication Immune Against Repetitive Attack, *WISA2004*, pp.400–410 (2004).
7) 菊池浩明，田中貴之，福野直弥，杉山太一，菊池大輔，寺田真敏，土居範久：ネットには何台の不正ホストがいるのか?，*Computer Security Symposium 2005*（*CSS2005*），愛媛県松山市 (2005).
8) Pfitzmann, A. and Waidner, M.: Networks without user observability, *Computer Security 2.6*, pp.158–166 (1987).
9) Reed, M.G., Syverson, P.F. and Goldschlag, D.M.: Anonymous connections and Onion routing, *IEEE Journal on Specific Areas in Communications*, Vol.16, No.4, pp.482–494 (1998).
10) Reiter, M.K. and Rubin, A.D.: Crowds: Anonymity for web transactions, *ACM Trans. Information and System Seurity*, pp.66–92 (1998).
11) Reiter, M.K. and Rubin, A.D.: Anonymous Web Transactions with Crowds, *Comm. ACM*, Vol.42, No.2, pp.32–38 (1999).

(平成 18 年 5 月 19 日受付)
(平成 18 年 11 月 2 日採録)

田村 仁

平成 13 年早稲田大学大学院理工学研究科博士前期課程修了．同年東京大学大学院情報理工学系研究科電子情報学専攻博士課程入学．平成 17 年東京大学・科学技術振興特任研究員．平成 18 年（独）産業技術総合研究所情報セキュリティ研究センター研究員．研究分野は PKI 構築，トラストメトリックス（信頼の定量化手法），およびプライベートインフォメーションリトリーバル，匿名通信路等．

古原　和邦

平成 7 年山口大学大学院博士前期課程修了．同年東京大学生産技術研究所入所．平成 12 年東京大学生産技術研究所助手．平成 14 年博士号（工学）取得．平成 18 年（独）産業技術総合研究所情報セキュリティ研究センター研究チーム長．同年同所主幹研究員．平成 8 年 SCIS 論文賞，平成 13 年 WISA 論文賞，平成 14 年 ISITA 論文賞，平成 15 年 SCIS20 周年記念賞，電子情報通信学会論文賞および猪瀬賞受賞．著書に『電子透かし技術—ディジタルコンテンツのセキュリティー』（画像電子学会編，共著，東京電機大学出版局），『情報セキュリティハンドブック』（電子情報通信学会編，共著，電子情報通信学会），『Mobile Communications Security』（Imai, H. (Ed.)，共著，Artech House Publishers），『ユビキタス時代の著作権管理技術—DRM とコンテンツ流通』（今井秀樹編著，共著，東京電機大学出版局）．

今井　秀樹（正会員）

昭和 41 年東京大学工学部卒業．昭和 46 年東京大学大学院博士後期課程修了（工学博士）．昭和 47 年横浜国立大学助教授．昭和 59 年同教授．平成 4 年東京大学教授（生産技術研究所）．平成 18 年より中央大学教授．東大名誉教授．平成 17 年産業技術総合研究所情報セキュリティ研究センター長兼務．現在に至る．この間，符号理論，情報セキュリティ，通信方式等の研究に従事．電子情報通信学会著述賞，論文賞，米澤メダル，猪瀬賞，業績賞，功績賞，IEEE シャノン記念論文賞，総務大臣表彰，経済産業大臣表彰，エリクソン・テレコミュニケーション・アワード等受賞．著書『情報理論』『符号理論』『暗号のおはなし』等．信学会理事，監事，IEEE 情報理論ソサイエティ会長，国際暗号研究学会理事，情報理論とその応用学会会長，CRYPTREC 委員長等を歴任．日本学術会議会員．IEEE，信学会フェロー．名誉博士（韓国，仏国）．