平成22年度　　修士論文

可用性と拡張性を有した
匿名通信方式の提案と実装

指導教官
齋藤 彰一 准教授

名古屋工業大学　情報工学専攻
平成20年度入学　20417541番

近藤 正基

目 次

第1章

はじめに

近年インターネットの普及により，高度な機密性が求められる医療や行政と個人との通信にもインターネットが使われるようになっている．このような分野では，通信の秘密をより厳重に守らなければならない場合がある．しかし，通信の内容は暗号化によって守ることができるが,「だれ」が「どこ」と通信を行ったといった通信そのものを機密にすることはできない．つまり，適切な場面では強固な通信の匿名性が必要である．本論文ではオーバーレイネットワークを用いて匿名通信を行う新しい方式の実装と評価について述べる．通信を行う際は送信者, 受信者が共に匿名のシステム内にあり, その中でユーザの匿名性を確保しつつ, サービスを提供できることが重要といえる．

インターネットにおける通信 (TCP/IP) では，送受信者の IP アドレスは外部から観測可能である．つまり，ネットワーク管理者や通信記録システムは通信している事実や通信量，頻度などを知ることができる．このため，医療情報検索や内部告発などの慎重に取り扱うべき通信も，ネットワーク管理者や通信記録システムによって観測されている．最悪の場合，管理者の不注意やマルウェア感染などで，通信の事実が明るみに出る可能性がある．これらから自身を守るために，送受信者が外部から特定困難な通信方式が必要である．

送受信者が外部から分からない通信の成立要件として，通信中の各メッセージについて以下の 3 種が挙げられる [1]．以下，これら 3 種を匿名のための性質 (**匿名性**) と言い，これら匿名性を備えた通信を**匿名通信**，匿名通信が使用する通信路を**匿名通信**

路と言う.

- 送信者が特定できないこと (送信者匿名性)
- 受信者が特定できないこと (受信者匿名性)
- 送受信者間を追跡できないこと (追跡不可能性)

この 3 種の匿名性の中で，送信者匿名性が最も重要である．匿名通信の利用者は，まず第一に自らを他者から保護したいと考えるからである．受信者匿名性は，Web などのアプリケーションによっては，送信者匿名性が十分であれば不要な場合もある．しかし, 非公開のサーバーとの通信では，受信者の匿名性も同時に実現する必要がある．また，追跡不可能性は，送信者匿名性か受信者匿名性の一方が成立することで成り立つ性質である．これらの匿名性を実現するには，メッセージに当該送受信者の情報 (IP アドレス，経路情報) が含まれないこと，もしくは特定できないことが必要である．

このような匿名性を実現した通信方法として，これまで様々な匿名通信手法が開発されている．代表的な方式として Mix-net[2]，Onion Routing[3][4]，Tor[5]，Crowds[6]，北澤らの方式 [7]，Aerie[8] がある.. これらの方式の共通点は，多数のノードによるオーバレイネットワークを用いてメッセージを多段中継するによって，送信者の IP アドレスを隠蔽している点である．一方，共通する問題点としてメッセージを中継しているノード (以下，中継ノードと言う) の故障やネットワークトラブルなどによる突然の離脱に対して，十分な対応ができない点である．また，既存手法は単一のサーバがネットワークの構成や，通信の制御などを把握するものが多く大規模で動的なネットワークに対応することが難しい.

そんな中，ノードの突然な離脱に対応した匿名通信路として Cashmere[9] が提案されている．この手法はシステム内のノードをグループに振り分け，多段中継することで匿名性を得る．そして，ノードが突然離脱した際には同グループ内のノードが中継を代行することで匿名性と耐離脱性を両立している．また各グループごとの中継は分散ハッシュテーブルによって管理され，ネットワークの動的な変化に対応できると考えられる．しかし，この手法は中継を代行するために経路の情報を同グループ内のノードに知らせる必要があり，匿名性を下げてしまう問題点がある.

本論文では，動的で大規模なネットワークに対応可能で，ノードの突然の離脱にも対応するためにノード管理と匿名通進路の管理を分離する二層構造をもつ匿名通信路を提案する．特にノードの管理には分散ハッシュテーブルの一種である Chord[13] と受信エリアというノード群を用いる．これらによって，強固な匿名性と可用性，スケーラビリティを両立する新しい匿名通信方式”Bifrost”を実現する．

本論文では，2 章で関連研究について述べる．3 章では Bifrost の詳細について述べ，4 章でその匿名性と耐離脱性，既存手法との比較を行う．また 5 章で Bifrost の実装と性能評価を述べ，最後に 6 章で本論文をまとめる．

第2章

関連研究

本章では，まず Bifrost が用いる分散ハッシュテーブルの一種である Chord の概要について述べる．次に，既存手法である Mix-net，Tor，Crowds，Cashmere，環状方式について述べる．

2.1 Chord

Chord は，数ある分散ハッシュテーブル [13][14][15] [16] の中の一種の手法である．分散ハッシュテーブルとは，中央にサーバが存在しない Pure-P2P のようなネットワークで効率よく通信相手を見つけることが出来る手法であり，このネットワーク構造は，拡張性と可用性に優れる．

また，Chord は，分散ハッシュテーブルの中でも，環状の ID 空間を用いることから，ID の始点と終点がないという特徴がある．環状空間を用いた匿名通進路の手法として Aerie や環状方式があり，匿名通進路において通信の始点と終点を秘匿するためには Chord は適している．

Chord は，ID 空間 (0〜$2^{160}-1$) の始点と終点を連結させることで環状の ID 空間と環状なノード群を形成する．各参加ノードは，ID 空間の部分空間を担当し，当該部分空間に割り当てられたコンテンツに対する検索と保持に責任を持つ．各参加ノードは，Fingertable と呼ばれる経路表を管理する．Fingertable には，自ノードの ID から $2^n(n = 0...159)$ だけ離れた ID を担当するノードとの接続情報 (IP アドレスなど) を保持しており，コンテンツ検索に用いられる．また，Successor(自ノードの ID よりも大

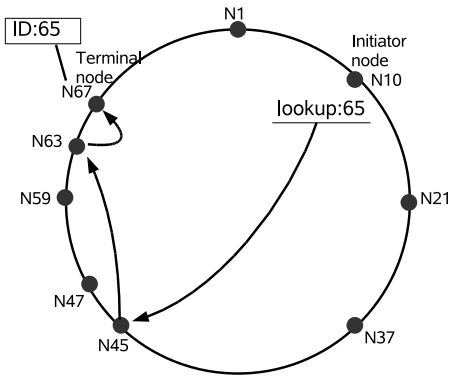


図 2.1: Chord よる探索

きな ID を担当するノードの中で最小の ID を担当するノード) と，Predecessor(自ノードの ID よりも小さな ID を担当するノードの中で最大の ID を担当するノード) という両隣接ノードとは，常に接続を維持している．

ノード離脱などに備えて，Successor List と呼ばれるノード情報を保持している．Successor List は，Successor の Successor や Successor の Successor の Successor という様に数ノード先までの接続関係のリストである．ノード離脱が発生した場合には，Successor List によって接続を修復する．また，Fingertable についても定期的な接続の確認と必要な場合は修復を行う．

コンテンツ検索時には，Fingertable に基づいて検索を行う．目的のコンテンツをハッシュ関数にかけて求められた Key を元に，Key の値以下で最大の ID を担当するノードを Fingertable から求める．さらに，求められたノードに対して同様に，Key の値以下で最大の ID を担当するノードを問い合わせる．これを繰り返すことで，最終的に担当ノードに接続する．図 2.1 に Chord の検索例を示す．なお，本論文では，最初に要求を出すノードを Chord 開始ノード (Initiator Node)，他ノードからの要求をさらに他ノードに転送するノードを Chord 中間ノード (Intermediate Node)，検索結果ノードを Chord 終了ノード (Terminal Node) と言う．図 2.1 は，Chord 開始ノード (N10) がコンテンツ (Key:67) を探索する場合を示している．まず，自ノードが保持している Fingertable の中から Key:67 以下で一番近いノード (N45) に検索要求を送る．N45 が Key:67 を担当していない場合，N45 は Chord 中間ノードとして自ノードが保持してい

表 2.1: 既存手法の特徴

手法	特徴
Tor	・多重暗号通信を行うことで高い匿名性を保有 ・ノードの離脱耐性がない ・スケーラビリティが低い
Crowds	・確率を用いた経路構築 ・暗号化を用いないため応答が早い ・受信者の匿名性は守られない ・スケーラビリティが低い
環状方式	・少数ノードで構成された環状ネットワークに沿ってメッセージを送受信 ・暗号処理は一度のみのため比較的応答が早い ・スケーラビリティが低い
Cashmere	・DHT と多重暗号通信の連携 ・高いスケーラビリティと可用性をもつ ・受信者匿名性が低い

る Fingertable の中から Key:67 以下で一番近いノード (N63) に対して検索要求を送る．このように検索を繰り返し，最終的にコンテンツを管理する Chord 終了ノードに検索要求を送る．

2.2 既存研究

既存の匿名通信方式より，代表的な方式である Tor と Crowds，環状経路を用いる方式 (環状方式)，提案手法と同様に DHT を用いた方式 Cashmere の概要を述べる．先にそれぞれの特徴を表 2.1 に示す．以下，各手法を詳細に説明する．

2.2.1 Tor

Tor は, Onion Routing の次世代版として開発され，一般公開 [5] されている匿名通信手法である．Tor は，ネットワーク上に複数配置された中継ノードで構成される．送信者は Directory Server から全ノードの情報を入手し，それらから選択した中継ノードと共有鍵を用いてメッセージを多重暗号化する．各中継ノードは受信した暗号メッ

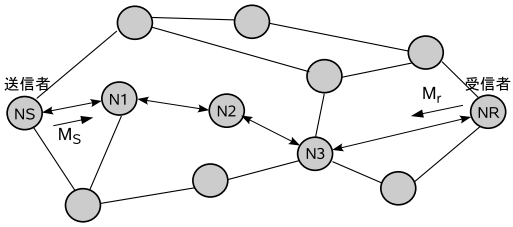


図 2.2: Tor の概要

セージを復号し，次ノードの情報を得る．これを受信者までの全ノードで繰り返し行う．この方式では，各中継ノードは，直接通信する前後の 2 ノードのみしか知ることができない．この多重暗号化によって，Tor は匿名性を実現している．

通信方式は，まず送信者が受信者に至る各中継ノードと共有する鍵を送信と返信用に 1 組ずつ生成し, 各中継ノードと共有する．これによって送受信の経路を確立する．次に，これら共有鍵を用いてメッセージを多重暗号化する．図 2.2 では, 送信者 NS から受信者 NR までの間に中継ノード N_i(i=1, 2, 3) がある．それぞれとの共有鍵を K_i(i=1, 2, 3), 通信内容を V すると，多重暗号化メッセージ Ms は以下に示す再帰的計算で得られる．なお，A‖B は A と B を結合したものを表し，IP_n はノード n の IP アドレスを示す.

$$Ms = IP_1||K_1(IP_2||K_2(IP_3||K_3(IP_{NR}||K_{NR}(V))))$$

上記の式により得られたメッセージを経路上の各ノードが共有鍵を用いて復号しながら受信者まで送る．受信者の NR は，メッセージを共有鍵を用いて復号することでデータ V を得る．返信時も同様に多重暗号化したメッセージを復号しながら送信者 NS まで送る.

2.2.2 Crowds

Crowds はいくつかの協調しあうノードのグループで構成される分散システムである．ユーザが匿名性を確保するためには Crowds のグループに加入しメンバリストを受け取る．それぞれのノードは中継ノードとしての役割も果たす．何かしらのデータ

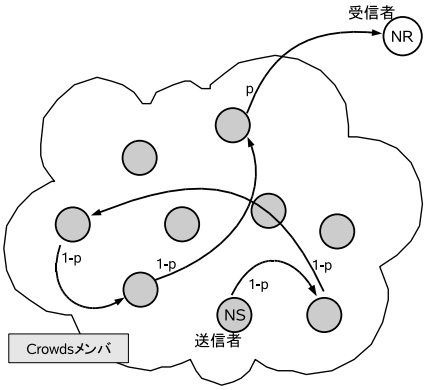


図 2.3: Crowds の概要

を匿名性を確保して送る場合, 受信者を記述したデータを宛先として Crowds のメンバに送る．動作の概要を図 2.3 に示す．送信者 NS はメッセージを p の確率で受信者 NR へ, (1 − p) の確率で他の Crows のメンバに送る．リクエストを受けとったメンバは, NS 同様に確率 p で受信者へ, 確率 (1 − p) で他のメンバにメッセージを送信する．この処理を繰り返すことにより, 最初にリクエストを発信したノードを容易に特定できなくする．また，返信の際には直前のノードに対して各ノードが返信メッセージを返すことで通信が可能となる．

2.2.3 初等的な環状経路と用いた匿名通信方式

文献 [7] で提案された通信方式について述べる．この手法は，参加ノード群から複数のルーティングエージェント (以下，RA と言う) を選択し，RA を環状に接続した通信経路による匿名通信を行う．本方式の特徴として，閉じた環状経路を利用しての往路 (送信者から受信者) と復路 (受信者から送信者) が異なる点がある．環状経路は，送信者から 0 個以上の RA を経由し受信者に至る．さらに，受信者から 0 個以上の RA を経由して送信者に至る．送信者と受信者を担当する RA が，環状経路のどこかに配置されていれば良い (図 2.4 参照)．このため，往路と復路の送受信処理に違いがなく，匿名性に対する通信状況の解析攻撃への耐性が高い．

さらに，送信者と受信者は必ず RA を利用する．一方，RA は複数のノードの送信者

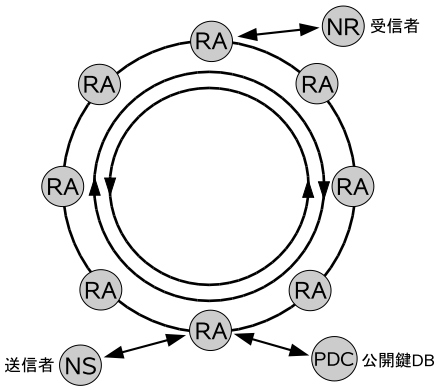


図 2.4: 環状経路方式の概要

もしくは受信者として担当する．そのため，送信者と受信者が共に環状経路に表れず，かつ，RA が判明しても送信者と受信者の匿名性を保護することができる．このため，環状経路内では，多重暗号などの経路を保護する機構が不要である．受信者を担当する RA の指定には，当該 RA のみが復号できる公開鍵暗号を用いる．公開鍵の管理は，公開鍵データベースセンタを用いる．

ノードの離脱時の対応は，異なる環状経路によるメッセージのバックアップ通信で行う．このため，通信遅延は少なくなる可能性があるが，メッセージの重複検知などの機構が必要である．

2.2.4 Cashmere

文献 [9] で提案されたノード離脱耐性のある匿名通信手法である．図 2.5 に示すように，Cashmere はいくつかのグループを経由する多重暗号通信を用いる．また，各グループごとに復号する鍵を共有している．そしてそのグループのどこかに受信者を配置することで，匿名性を得る手法である．またノードが離脱した際は，同グループ内のノードが復号を代行する．そのため，本来復号するノードが離脱したとしても途切れることなく通信を行うことができる．Cashmere は分散ハッシュテーブルの一つである Pastryq[14] と Tor と同様な多重暗号通信を組み合わせることでこれを実現している．

通信方式を詳細に述べる．まず Cashmere は分散ハッシュテーブルの一種である Pastry

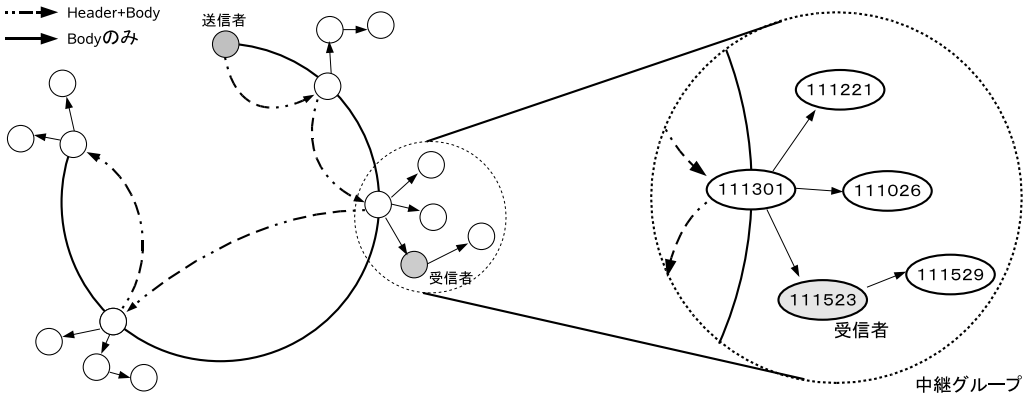


図 2.5: Cashmere の概要

を構成する．Pastry は木構造で ID 空間を管理する DHT であるため，ある ID の範囲を指定するとその範囲の ID 空間にも親となるノードと子となるノードが存在する．この範囲をグループとして，いくつかのグループ同士で多重暗号通信をおこなう．指定したグループの根となるノード (以下ルートノード) は，メッセージを自身の鍵で復号し，復号内容と共有鍵をグループ全体にブロードキャストする．もし，ルートノードが故障などで突然離脱してしまった場合，グループ内のノードからルートノードが選出され，このノードが復号を代行する．

また Cashmere はメッセージをヘッダ部とボディ部に分離し，それぞれを別に暗号化している．このとき暗号化にはヘッダ部はルートノードの鍵，ボディ部は受信者の鍵を用いて暗号化される．またメッセージを中継する際，ルートノードはヘッダ部の復号を行い，グループ内の全ノードはボディ部の復号を試みる．これにより，経路を制御するノードと受信者を分離することができ，経路の途中に受信者を配置することが出来るようになる．また，ルートノードは共有鍵を同グループ内のノードにブロードキャストするが，この共有鍵はヘッダを復号する鍵であり，通信内容を見られてしまうことを防ぐことができる．

Cashmere はこれらの手法によりノードが突然離脱した際も途切れることなく，匿名に通信を行うことができる．しかし，グループ内の全ノードにヘッダ用の共有鍵を配布してしまうことが前提であり，これは匿名性を下げてしまう一因となる．

第3章

提案手法

本章では，提案する本方式について述べる．まず，本方式の全体構成を示し，続いて通信手順について述べる．

3.1 全体構成

本方式の構成は，オーバーレイネットワークを構成するノード群と，各ノードの公開鍵を管理する**公開鍵サーバ**(PKS) で構成される．全体像を図 3.1 に示す．各参加ノードは，Chord の参加手続きに基づいてオーバーレイネットワークに接続し，公開鍵サーバに自身の公開鍵を登録する．公開鍵サーバは，参加ノードの ID と公開鍵を関連付けて管理し，要求に応じて参加ノードに提供する．

本方式は，**ノード管理層**と**匿名通信路層**の二層構造で構成される．ノード管理層は，参加ノードの ID 管理を行う層で Chord による分散ハッシュテーブル機能を有する．この層が，ノードの参加と離脱，経路情報に関する保守を行う．また Chord の検索には再帰検索と反復検索の二種がある [17] が，匿名性の観点から再帰検索を行う．匿名通信路層は，ノード管理層の上位に位置し，匿名通信路の決定，構築，暗号化など，実際の通信を行う層である．

3.2 ノード管理層

ノード管理層は，参加ノードの ID 管理と参加離脱管理と経路情報管理を行う．具体的には，分散ハッシュテーブル Chord を利用し，ノードの参加や離脱は Chord のアルゴリズムに従って処理する．また，匿名通信路層からのノード検索要求に対して，Chord の経路制御機構 (FingerTable) に従って経路制御を行う．

ノード管理層を独立させる利点は，ノード状態管理と匿名通信路の維持管理の依存関係をなくすことができる点である．オニオンルーティングなどに見られる多重暗号化を用いた匿名通信路は，匿名としての性質を保つため，必要最小限 (前後) のノードとしか接続を維持しない．これは，前後以外のノードが分かることで，経路全体の情報を得ることになり匿名性が低下するためである．そこで，本方式ではノード管理を匿名通信路の機能から分離する．これによって，匿名通信路の構成を考慮することなくノード管理が可能となる．ここで本方式では，分散でのノード管理と経路制御が可能な分散ハッシュテーブルに着目し，その中でも環状空間を構成する Chord をノード管理層の基盤とする．

3.3 匿名通信路層

匿名通信路層は，匿名通信路の経路決定と構築を行い，必要な暗号処理を実施して実際に通信を行う層である．匿名通信路層は，ノード ID の割り当て，稼働状況確認を行う必要がない．一方，匿名通信に関するすべての処理を行う．本節では匿名通信路層が行う各処理の詳細について述べる．

3.3.1 通信方式の概要

本方式の通信は，オニオンルーティングと同様の多重暗号を用いる．オニオンルーティングと異なる点は，受信者を匿名通信路の途中に配置することで匿名性を向上させる点と，送信と返信を別経路で構築する点の 2 点が存在する．前者の利点として受信者以降の通信がダミー通信となるため，通信を分析して匿名性を低下させる解析攻

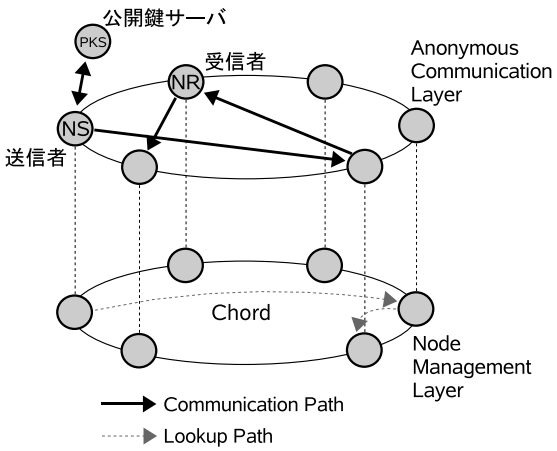


図 3.1: 本方式の全体構成

撃にも有効な点がある．また後者の利点は，中継ノードを含め観測ノードが送信，返信メッセージを区別することができなくなるため，送信者と受信者のつながりを特定しにくくなるという点がある．

匿名通信路の構築には，3 種のメッセージを用いる．1 種類目は，匿名通信の開始時に 1 度だけ送信される**構築メッセージ**，2 種類目は通信データを運ぶ**データメッセージ**，3 種類目は**制御メッセージ**である．構築メッセージには，各中継ノードとの共有鍵と受信者との共有鍵，返信用の構築メッセージを含む (共有鍵は，匿名通信路毎，中継ノード毎に異なり，返信用を含めてすべて送信者が作成する)．構築メッセージは，公開鍵暗号を用いた多重暗号化を行い，送信者が指定した経路 (中継ノード群) を辿る．構築メッセージを受信した各中継ノードは，自身の秘密鍵を用いて構築メッセージから送信者との共有鍵を取り出し，匿名通信路毎の**コネクション情報**を作成する．コネクション情報は，経路の識別 ID と，直後のノードの IP アドレスと Chord 空間における ID，当該構築メッセージに含まれていた共有鍵で構成する．コネクション情報は以降のデータメッセージの送受信の際に，使用するべき共有鍵を判断や次に送るノードを判断などに使用する．なお，共有鍵は定期的に更新し，漏洩に備える．

データメッセージは，一般の通信データを送信者と中継ノードとの共有鍵で多重暗号化したものである．各中継ノードは，コネクション情報に基づいてデータメッセー

ジの受信と復号を行い，次ノードに中継する．制御メッセージは，メッセージ内に制御命令を保持し，当該匿名通信路に対する各種制御に用いる．例えば，経路破棄命令や，鍵更新命令がある．制御命令は，共有鍵で暗号化されており，無関係な匿名通信路を制御することはできない．

匿名通信路が経由する中継ノード群と受信者，返信の経路の決定は，匿名通信路毎に送信者が行う．送信者は通信内容の性格や許容しうる通信遅延時間などを検討して，中継ノード数を決定する．その後，中継ノード数分の中継ノード ID を決定する．この中継ノード ID はノード管理層が提供する ID の範囲であれば任意でよい．送信者は，中継ノード ID を担当する中継ノードと受信者の公開鍵を公開鍵サーバから取得し，多重暗号による構築メッセージを作成する．次に，最初の中継ノードを次ノードとするコネクション情報を作成し，指定された IP アドレスに構築メッセージを送信する．構築メッセージを受けとった各中継ノードは，最後の中継ノードに至るまで中継を繰り返す．

3.3.2 受信エリア

本方式は多段中継による通信を行うため，次の問題が考えられる．問題 1) 中継ノード間の経路制御をノード管理層の Chord に依存するため構築メッセージの通信時間が長い．問題 2) 多重暗号のための暗号化と復号処理にかかる時間が長い．そこで，本方式では，これらの問題に対して**受信エリア**を導入し，コスト軽減を図る．

受信エリアとは，Chord の ID の連続した部分空間であり，その ID 部分空間を担当するノード群のことである．受信エリアは以下の性質を有する．

- 受信エリアに属するノードは，すべて当該匿名通信路の中継ノードである
- 受信エリア内では，ノード検索による経路制御を行わずに，隣接する Successor にメッセージを中継する
- 受信エリアの終点は，構築メッセージのヘッダを復号できたノードである
- 受信エリアは 1 つの匿名通信路に複数設定することができる

- 受信エリアはノード 1 つ以上で構成する

つまり受信エリアは，構築メッセージのヘッダに記述された ID から始まり，当該構築メッセージを復号できるノードまで続く．その間の経路制御は，静的に Successor に送るだけの固定経路となる．構築メッセージヘッダを復号できた受信エリアの終点ノードは，復号で得られたヘッダに記載された次の受信エリアの始点ノードに向けてメッセージを中継する．この際の経路制御は，ノード管理部の Chord によるノード検索である．この受信エリアは次の利点がある．1) 受信エリア内の中継ノード間のノード検索が不要となる．2) 多重暗号化の対象が受信エリア終点ノードに限定できるため，暗号化コストが軽減できる．1) と 2) 共に中継ノードの数を減らすことなく可能である．これは，先の問題 1) と 2) に対する解決策となる．一方，中継ノードが一つでも判明すると当該中継ノードの周りが受信エリアと推測できる．受信エリアがない場合は，1 つの中継ノードが判明しても他の中継ノードには影響がない．受信者は中継ノードの一つであることを考慮すると，匿名性を低下させている．以上より，送信者は次の利用方針を考慮して，受信エリアの利用を決定する．1) 高い匿名性を要する場合は，受信エリア内ノード数を 1 として受信エリアを増やすことで匿名性を確保する．2) 高速な通信が必要な場合には，受信エリア内ノード数を大きくして受信エリアを少なくし，暗号化とノード検索コストを抑える．

3.4 通信手順

本方式は，メッセージ生成フェーズ，送信フェーズ，返信フェーズからなる．本節では動作の詳細を説明する．表 3.1 に本方式の構成要素を示す．

3.4.1 メッセージ生成フェーズ

メッセージ生成フェーズの疑似コードを図 3.2 に示す．また，図 3.2 中に使われる MakeHeader() と MakeBody() の疑似コードをそれぞれ図 3.3 と図 3.4 に示す．送信者は，図 3.2 のアルゴリズムで構築メッセージを作成する．まず，指定した受信エリアの範囲と配布する共有鍵を受信エリア

表 3.1: 構成要素

送信者	NS	匿名通信の始点ノード
受信者	NR	匿名通信の終点ノード
Chord 中間ノード	$R_{i,j}$ $(j = 1, 2, ...)$	受信エリア A_i へ Chord に従い中継するノード
受信エリア数	n	受信エリアの数
受信エリア	A_i $(i = 1, 2, ...)$	NS が指定した ID 空間. 1,2... の順で送信される
受信エリアの最小 ID	$ID_{min}(A_i)$	受信エリア A_i の始点 ID
受信エリアの最大 ID	$ID_{max}(A_i)$	受信エリア A_i の終点 ID
受信エリアの始点ノード	As_i	$ID_{min}(A_i)$ を管理するノード
受信エリアの終点ノード	At_i	$ID_{max}(A_i)$ を管理するノード
公開鍵サーバ	PKS	参加ノードの公開鍵を 管理するサーバ
公開鍵	P_{node}	ノード $node$ の公開鍵
公開鍵暗号化データ	$P_{node}(Y)$	データ Y を P_{node} で 暗号化したデータ
中継ノード共有鍵 (送信者用)	Cs_{At_i}	NS と中継ノードとの共有鍵
中継ノード共有鍵 (受信者用)	Cr_{At_i}	NR と中継ノードとの共有鍵
受信者との共有鍵	C_r	NS 受信者との共有鍵
初期構築メッセージヘッダ	HS	通信開始前の 構築メッセージのヘッダ部
通信途中の構築メッセージヘッダ	HS_i	通信途中の 構築メッセージのヘッダ部 i は通信中の受信エリア番号
初期構築メッセージボディ	BS	通信開始前の 構築メッセージボディ部
初期返信構築メッセージヘッダ	HR	返信用構築メッセージヘッダ部

終点のノードの公開鍵を用いて多重暗号化する (図 3.3 参照)．多重暗号化した送信先がヘッダ部であり，この構成を図 3.5 に示す．また，この時作成した共有鍵は保管し，GetKey($nodename$) により $nodename$ の共有鍵を取得することができるとする．次に，受信者が送信者に返信する際のヘッダ部を送信用と同様の方式で生成する．これは送信者が返信の際の経路を指定するためである．

```
GenerateConnectMessage(){
    HS = MakeHeader(AS_IDmin[1〜n] , AS_IDmax[1〜n]);//送信用ヘッダ生成
    HR = MakeHeader(AR_IDmin[1〜n] , AR_IDmax[1〜n]);//返信用ヘッダ生成
    BR = MakeBody(null , Psender , mr, AR_IDmax[1〜n]);//返信用ボディ生成
    受信者との共有鍵Crを生成;
    CAR = null;
    for(i = 1; i <= n; i++)
        CAR = CAR || GetKey(AR_IDmax[i]);//返信用各エリア終端ノードの共有鍵取得
    BS = MakeBody(HR||BR||CAR||Cr, Preceiver, ms,
                                        AS_IDmax[1〜n]);//送信用ボディ生成
    return HS||BS;
}
```

図 3.2: メッセージ生成の疑似コード

```
MakeHeader (IDmin[1〜n] , IDmax[1〜n]){
    HS = null;
    for(i = n; i >= 1; i–){
        P = PKSよりIDmax[i]の公開鍵を取得;
        C = 共通鍵生成;
        共通鍵(C)を保存;
        HS=P( IDmin[i-1] || C || HS );
    }

    return  HS ;
}
```

図 3.3: MakeHeader の疑似コード

```
MakeBody (Data, Pnode, m /*受信者のエリアの番号*/,
                    IDmax[1〜n] ){
    BS = Pnode(Data);
    for(i = m-1; i >= 1; i--){
        C = GetKey(IDmax[i]);
        BS = C(BS);
    }
    return  BS;
}
```

図 3.4: MakeBody の疑似コード

次に受信者が送信者に返信する際のボディ部を生成する．これは null のデータを自身 (送信者) の公開鍵で暗号化した後，返信用に「送信者にメッセージが届く以前に現れる受信エリア終点ノード」に配布する共有鍵で多重暗号化したものである (図 3.4 参照)．例えば，受信エリア数が 5 として，受信者 (送信者に返信する場合は送信者) が 3 番目のエリアに位置したとすると，2 番目と 1 番目の受信エリア終点ノードで多重暗号化する．

次に，送信用のボディ部を生成する．まず，受信者との共有鍵と返信用の受信エリア終点ノードの共有鍵を取得する．その後，返信用のヘッダ (HR) とボディ (BR)，受信者との共有鍵 (Cr)，受信エリア終点ノードに配布する共有鍵 (CAR) を受信者の公開鍵で暗号化した後，受信者にメッセージが届く以前の受信エリア終点ノードに配布する共有鍵で多重暗号化する．以上により作成したヘッダ部とボディ部を結合することでメッセージを作成する．

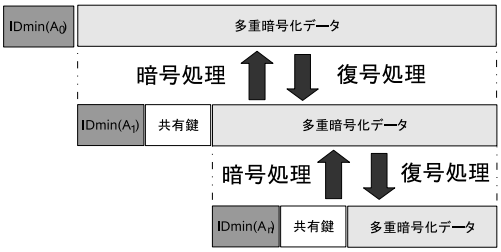


図 3.5: ヘッダ部の構成

3.4.2 送信フェーズ

```
Relay(){
    msg = 受信メッセージ;

    if(受信エリア外){
        ヘッダより次ノードID(nextID)を取得;
        nextAddress = ChordSearch(nextID);
        SendMessage(nextAddress, msg);
    }
    else{ // 受信エリア内
        If (nextID == 自ノードID)
            msg中のHSからnextIDを削除;

        if(decrypt(HS) == false){ //HSの復号が失敗
            nextAddress = ChordSuccessor();
            SendMessage(nextAddress, msg);
        }
        else{  //HSの復号が成功(受信エリア終端ノード)
            復号したヘッダより次ノードID(nextID)と
                送信者との共有鍵CとBSを取得, 保管;
            if(nextID == null)
                return; // 通信路終端の場合は処理終了
            nextMsg = Cを用いて復号(BS);
            nextAddress = ChordSearch(nextID);
            SendMessage(nextAddress, nextMsg);
        }

        if(decrypt(BS) == true){ // 受信者確認の復号
            HR,BR,共有鍵を入手, 保管;
        }
        セッション情報を登録;
    }
}
```

図 3.6: 送信フェーズの疑似コード

送信フェーズの疑似コードを図3.6に示す．この中のChordSearch($nextID$)はChordの検索手法を行って$nextID$を管理するノードのIDアドレスを返す関数である．SendMe--ssage($address$,msg)は指定した$address$にmsgを送る関数である．また，decrypt($data$)は$data$を自身が持つ秘密鍵を用いて復号を試み，復号が成功した場合にtrueを返す関数である．

まず，自身が受信エリア内か外かを判断する．受信エリア外の場合，Chordの検索アルゴリズムにより得られたアドレスにメッセージを送信して，処理を終える．受信エリア内の場合，まず送信ヘッダにあるIDを削除する．自身の秘密鍵を用いてヘッダの復号を試みる．復号できなかった場合，Chord空間での次ノード(=Successor)にメッセージを送る．復号できた場合は，ノードは受信エリア終点ノードである．ヘッダから，送信者との共有鍵と次のID(=nextID)を得る．この時nextIDがnullだった場合，メッセージの送信を終了する．また受信エリア終点ノードは，得られた共有鍵を用いてボディ部(BS)を復号し，nextIDを管理するノードをChordの検索処理を用いて探索してメッセージを送る．

最後に，受信エリア内のすべてのノードはボディ部の復号を試みる．ボディ部を復号することができた時，当該ノードが受信者である．受信者は，返信の為のメッセージと，送信者との共有鍵，返信の際の受信エリア終点との共有鍵を得る．

3.4.3 返信フェーズ

経路構築の際，受信者が受け取るメッセージには返信用のメッセージと，返信時のAt_iとの共有鍵，送信者との共有鍵が含まれている．これらは，あらかじめ送信者が作成したものである．受信者はこの時受け取るを返信用メッセージを用いて経路の構築を行う．この方法により，送信者が返信経路も指定する．この多重暗号化は送信メッセージと同様に受信エリアの指定やメッセージを受信するノード(返信時は送信者を指す)の位置の設定などが行われている．そのため，受信者は返信用メッセージのヘッダ部に従い図3.6が示す送信フェーズと同様にメッセージを送信することのみで，受信者から送信者への経路構築が行われる．また，返信メッセージは送信メッセージと同様の形式をとるため，中継ノードも図3.6の送信フェーズと同様の動作を行う．送

信フェーズと返信フェーズの異なる点は，経路を構築するためのメッセージを作成するか否かにある．返信時は，前述の通り経路構築メッセージを作成しない．

この手法の利点は，送信と返信が別経路をとること，返信の経路構築が全て送信者によって行われることの 2 点がある．送信と返信が別経路をとることは，送受信者のつながりが推定しにくくなる．Tor や Crowds の様に同様の経路をとる場合，中継しているノードは送信，返信のどちらのメッセージも中継することになり，この場合送信者と受信者のつながりを推定しやすくなると考えられる．Bifrost では，まったく違う経路をとり返信時も送信時と同様のメッセージを使用するため，送受信者のつながりは推定しにくい．

また返信の経路構築が全て送信者によって行われることで，受信者に対し送信者の情報を一切あたえることなく通信を行うことができるようになる．受信者は返信時，どのノードが中継しているのかということすら知ることができない．そのため，送信者は受信者に対しても匿名性を守ることができる．

また経路構築後は，一度目の通信で得た共有鍵を用いて返信する内容を多重暗号化し，メッセージを送信する．受信者は返信すべき内容を取得した共有鍵で単純に暗号化し，構築した経路に対して送信するのみであるため，送信者の情報を得ることはないと考えられる．

3.4.4 動作概要

図 3.7 に受信エリア数 2 の場合の動作を示す．なお図中には中継ノードと Chord 中間ノードのみを記した．まず，送信者 NS は最初の受信エリアの始点ノード $ID(A_1s)$ にメッセージ Ms を Chord のアルゴリズムに基づいて送信する (図 3.7:Lookup Path)．A_1s からは Successor にメッセージを中継する (図 3.7:Communication Path)．受信エリア内のノードは，自身の通信用秘密鍵を用いてヘッダ部の復号を試みる．復号できなければ Successor に送る．もし復号できた場合は, 次のエリアの始点ノードの $ID_{min}(A_2)$ を知ることができるので, initiator となって Chord のアルゴリズムに従いメッセージを送る．このときメッセージの対応関係を知られないために，ボディ部を得られた共有鍵を用いて復号する．これを繰り返し行うことで，ヘッダ内に受信エリアがなくな

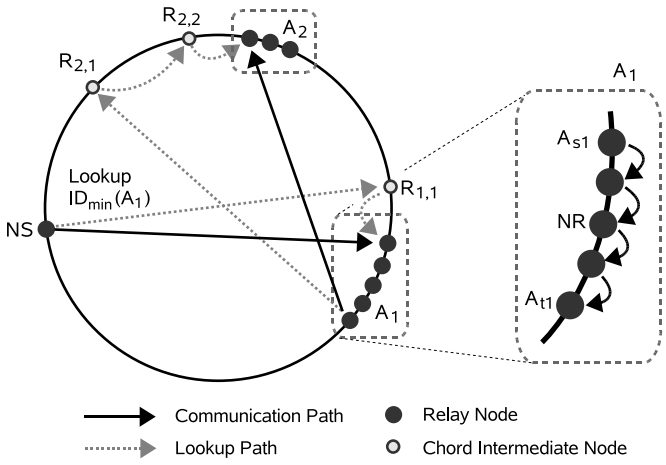


図 3.7: 本方式の経路制御

るまでメッセージを送る．

また, 受信エリア内のノード (A_iN_k) はメッセージ中継後に, データ部をデータ用秘密鍵を用いて復号を試みる．復号できた場合, 当該ノードが受信ノード NR であることが判明する．返信は, 送信メッセージのデータ部に含まれる返信用ヘッダを用いて同様に送る．

データ部とヘッダ部を独立に暗号化することと, 返信時のヘッダを送信者が用意すること, さらに受信後もすべての受信エリアについて送信が続くことで, 送受信者の匿名性と送受信者のつながりの匿名性を確保する．

3.5 バックアップノード

Bifrost は中継するノードを Chord の Lookup を用いて検索する．また，受信エリア内では単純に Successor に対してメッセージの中継を行う．そのため，ほとんどの中継ノードが突然に離脱した際にも，Chord のネットワークが破綻しない限り通信を続けることが出来る．しかし，これまでの提案では受信エリアの終端ノード At_i が故障した場合のみ，メッセージの中継が不可となってしまう．この単一の故障点を防ぐために，Bifrost は At_i のバックアップをとる．

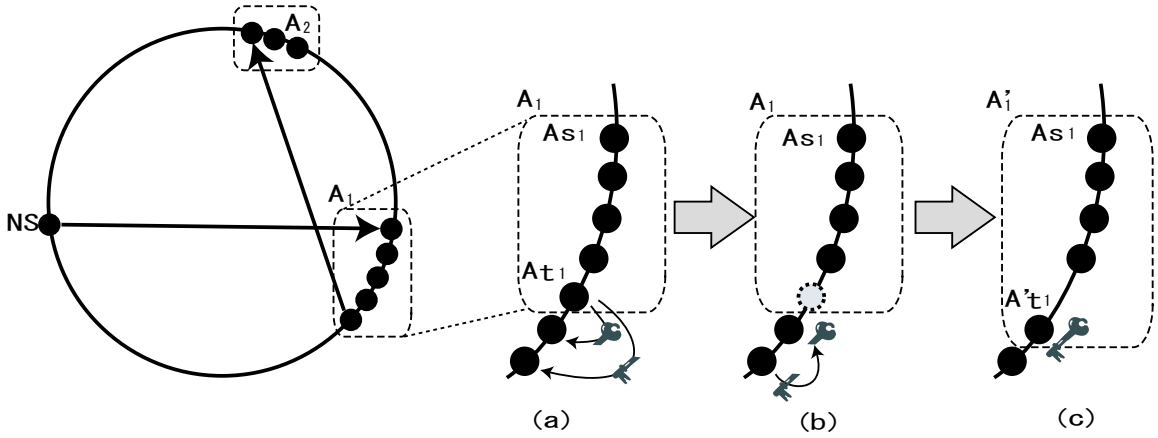


図 3.8: バックアップの概要

図 3.8 にバックアップの概要を示す．図 3.8 の例では，At_1 のノードが離脱してしまった場合，メッセージを復号することができず，経路が途切れてしまう．そこで，At_1 は自身の鍵を分割し Successor と Successor の Successor(以下 Next Successor) に対して配布する (図 3.8(a))．Chord はそのネットワークの構造を崩さないために，一定時間ごとに Successor と通信を行う．そのため，もし At_1 が離脱した場合 At_1 の Successor は At_1 の離脱を感知する．感知後，Successor は Next Successor に対し分割された At_1 の鍵を要求する．Next Successor から分割された鍵を受けとると，自身が保持している At_1 の鍵と組み合わせることで，メッセージ復号のための鍵を得る (図 3.8(b))．その後，Successor は At_1 の代わり ($A't_1$) となり，メッセージを復号と中継を代行する (図 3.8(c))．

受信エリア内のノードは単純に Successor に対してメッセージを送る処理を行うため，At_1 の Successor は特別な処理なくメッセージを受け取ることができる．また，復号用の鍵がバックアップノードの結託などにより漏洩してしまった場合も，ヘッダ部とボディ部を分離して暗号化しているため，受信者の情報の漏洩には直結しない．しかし，少なからず匿名性を低下させてしまうことは自明なので，一定時間ごとに鍵の再設定を行うことが望ましい．

第4章

検証

本章では提案手法の理論的な検証を行う．まず匿名性の観点から提案手法を検証し，次に耐離脱性を評価する．そして最後に既存手法との比較を行う．

4.1 システム全体の匿名性検証

4.1.1 匿名性評価方法

現在，匿名性の一意的な評価として，いくつかの手法 [18][19][20] が提案されている．我々は，Claudia らが提唱する評価基準 [18][19] を用いて提案手法を評価した．評価のために，Bifrost のパラメータを以下の様に設定する．

- N:ネットワーク内の総ノード数
- f:ネットワーク内の攻撃者の割合
- g:受信エリア内のノード数
- L:経路 (往路もしくは復路) の復号回数 (=受信エリア数)

また全てのノードは攻撃者になりうる状況を想定する．攻撃者となるノードの割合が増えることは，(1) 経路が特定されやすくなること，(2) 中継者がもつ共有鍵が漏洩しやすくなること，の 2 つに直結する．また検証のために，受信エリア内のノード数は g で統一している．

またここで記述する匿名性は Pfitzmann らが提唱したもの [21] に準ずる．これは，ある匿名システム内に $\Omega(=N)$ ノードが含まれているとすると，送信者 (もしくは受信者) の匿名性がもっとも高い場合は，全てのノードが送信者 (受信者) と他のノードと区別がつかない状況を指す．文献 [21] では Ω をその状況下での匿名性を評価するパラメータとし，同状況下での送信者 (受信者) が判別される確率は $P(\Omega)^2$ で表される．また実際にはシステムから情報が漏洩する．ある状況下で情報が漏洩し，攻撃者が送信者 (受信者) を $\Omega_i(\cup\Omega)$ 内にいると特定できるとき，特定される確率はは $P(\Omega_i)^2$ となる．つまり最悪の場合は，送信者 (受信者) が単一のノードに特定された場合であり，その時の特定される確率は 1 となる．

我々はこの考えを発展させた Claudia らが提唱する評価基準 [18][19] を用いて提案手法を評価した．これは，システム全体のエントロピーを用いて匿名性を一意に評価する手法である．以下に，システムのエントロピーの計算方法と匿名性の計算方法を簡潔に記述する．

定義 1：エントロピー計算方法

ネットワーク内のノードが Ω であるとする．このとき送受信者間の通信でいくつかの情報が漏洩し，それぞれのノード u が p_u の確率で送信者 (もしくは受信者) だと特定できるとする．このときシステムのエントロピーは以下の式で表される．

$$H(\Omega) = -\sum_{u\in\Omega} p_u \log_2(p_u)$$

あるシステムにおいて情報が攻撃者に全く漏洩していない場合，システム内全てのノードが送受信者，中継者などの判別がつかない．この場合:$\forall u \in \Omega,\ p_u = \frac{1}{\Omega}$ となる．そのため，この状況下のエントロピー H_m は $H_m = \log_2(|\Omega|)$ で計算される．この値は，情報が攻撃者に全く漏洩していないシステムのエントロピーであり，そのシステムのエントロピーの最大値である．

定義 2：匿名性計算手法

定義 1 を用いてシステム全体の匿名性 D を以下の計算式で表される．

$$D = \frac{H(\Omega)}{H_m(\Omega)} = \frac{-\sum_{u \in \Omega} p_u \log_2(p_u)}{\log_2(|\Omega|)}$$

上記の式で表される匿名性は，評価する状況下でのシステムのエントロピーをシステムのエントロピーの最大値で割ったものである．そのため D は，$0 \leq D \leq 1$ をとる．$D = 1$ の場合，攻撃者はシステム内全てのノードにおいて送受信者，中継者などの判別がつかない状況を指し，$D = 0$ の場合，送受信者を単一ノードまで特定できることを指す．

この方法で評価される匿名性はシステムのエントロピーに起因するため，局所的に匿名性を評価するより，様々なシステムを公正に評価することができると考えられる．

この評価方法を用いて提案手法を評価する．また評価に用いるシステムは $N = 2000$ とする．また受信エリア内のノード数と受信エリア数は文献 [9] のリレーグループ数に習い $L = 5$，$g = 4$ とした．

4.1.2 送信者匿名性

攻撃者がいない場合，送信者は受信エリア終点ノードと区別がつかない．システム内の攻撃者の割合が増えるにつれ，送信者は特定されていく．また受信エリア始点のノード (AS) は，前受信エリアを知ることができるため，このノードが攻撃者である場合，他のノードが攻撃者である場合に比べ，情報の漏洩は大きい．また，送信者より前にメッセージを中継するノードがいないことは明らかなので，いくつかの AS が攻撃者であった場合，最初にメッセージを受信した AS の一つ前のノードは，他のノードに比べ送信者である確率が高いと言える．評価のために受信エリア数が L，AS の結託によって受信エリアが n 特定されたと仮定すると，最初にメッセージを受信した攻撃者である AS の一つ前のノードが送信者である確率は，$\frac{1}{L-n+1}$ で表すことができる．またその他の攻撃者でないノードが送信者である確率は，$1 - \frac{1}{L-n+1}$ である．つまり，

攻撃者でないノード u が攻撃者である確率 p_u は以下の式で評価できる．

$$p_u = \begin{cases} \frac{1}{L-n+1}： \text{”攻撃者である最初の } AS\text{”の 1 つ前のノード} \\ (1-\frac{1}{L-n+1}) \times \frac{1}{(1-f)N-1}： \text{他ノード} \end{cases}$$

攻撃者に AS が含まれない場合，攻撃者を除いた他のノードは全て送信者，それ以外のノードと区別がつかない．そのため，攻撃者でないノードが送信者である確率は $\frac{1}{(1-f)N-1}$ となる．

この式を用いて Bifrost，Cashmere，Tor，Mix-net の送信者匿名性を評価した．評価方法は $n(=0,1,2,...,L)$ 以下の様に場合分けする．その後，エントロピーを計算し，n の発生確率を G としてそれぞれを乗算することで評価した．以下に Bifost の場合分け $(L=5)$ を例示する．

$n=0$ の場合 : $G=(1-f)^5$, $p_u = \frac{1}{N(1-f)}$ ： N ノード

$n=1$ の場合 : $G =_5 C_1 f(1-f)^4$, $p_u = \begin{cases} \frac{1}{5} & ： 1 \text{ ノード} \\ (1-\frac{1}{5}) \times \frac{1}{(1-f)N-1} & ： N-1 \text{ ノード} \end{cases}$

$n=2$ の場合 : $G =_5 C_2 f^2(1-f)^3$, $p_u = \begin{cases} \frac{1}{4} & ： 1 \text{ ノード} \\ (1-\frac{1}{4}) \times \frac{1}{(1-f)N-1} & ： N-2 \text{ ノード} \end{cases}$

$n=3$ の場合 : $G =_5 C_3 f^3(1-f)^2$, $p_u = \begin{cases} \frac{1}{3} & ： 1 \text{ ノード} \\ (1-\frac{1}{3}) \times \frac{1}{(1-f)N-1} & ： N-3 \text{ ノード} \end{cases}$

$n=4$ の場合 : $G =_5 C_4 f^4(1-f)$, $p_u = \begin{cases} \frac{1}{2} & ： 1 \text{ ノード} \\ (1-\frac{1}{2}) \times \frac{1}{(1-f)N-1} & ： N-4 \text{ ノード} \end{cases}$

$n=5$ の場合 : $G=f^5$, $p_u = 1$ ： 1 ノード

上記の計算と同様にして Bifrost，Cashmere-max，Cashmere-min，Tor，Mix-net を評価した．Cashmere-max は文献 [9] に示されている値である．文献 [9] では受信するグループ内のノードを「復号を行うノード」と「それ以外のノード」に分けて評価され，「それ以外のノード」が攻撃者である場合の情報の漏洩は低く見積もられている．しかし，Cashmere のアルゴリズムはグループ内全てのノードに鍵を配布するため，我々はグループ内のノード全てを「復号を行うノード」と同等の情報量を保有していると仮定して評価した．その値を Cashmere-min とした．評価結果を図 4.1 に示す．

Mix-net は，常にシステム内の全てのノード対してブロードキャストを行うことで送信者，受信者を秘匿する方法である．そのため RTT は非常に低速であり，本論の背景

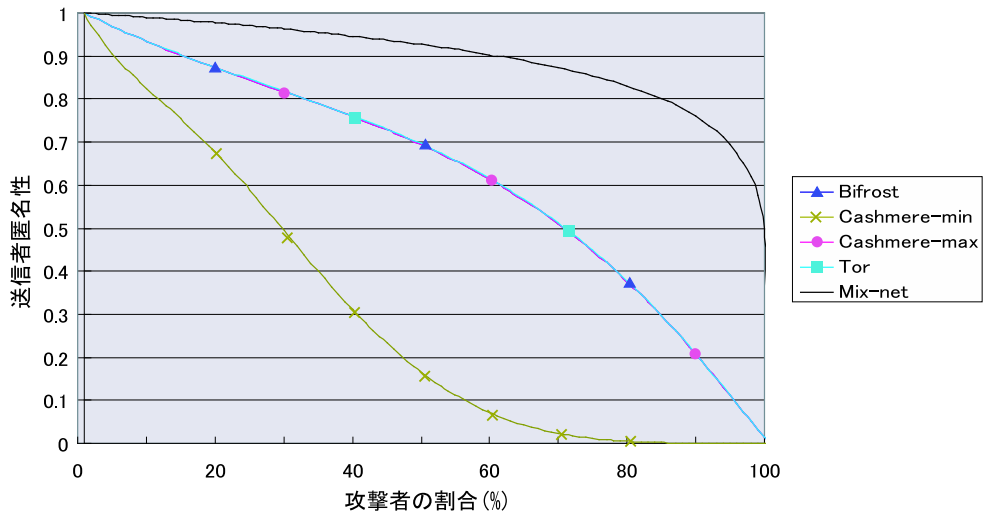


図 4.1: 送信者匿名性

にある大規模な匿名通信システムには適応することは現実的でない．しかし，Mix-net 内のノードの送信者である確率は常に $\forall u \in \Omega$，$p_u = \frac{1}{N}$ をとる．これは，Mix-net が匿名システムにおいて常に最大の匿名性をとることを表し，Mix-net の曲線がある攻撃者の割合での理想の値を示している．

ほか，Bifrost，Tor，Cashmere-max は同等の値をとり中央の値で重なっている．これらの手法はすべて多重暗号通信を用いているので，あるノードが送信者である確率がメッセージを受け取った最初の攻撃者の前のノードで高く，他のノードは同等であるという条件が同様の条件となったためである．一方 Cashmere-min は，他の手法と比べ大きく低い値をとった．これは，Cashmere がノードの離脱のために共有鍵をグループ内全てのノードに対して配布することにより，1 ノードでもグループ内に攻撃者がいる場合，当該グループの前グループが特定されてしまうといった，最悪の状況を想定した場合を示している．よって，Cashmere の送信者匿名性は Cashmere-max から Cashmere-min の間をとると考えられる．そのため Bifrost は少なくとも Cashmere 以上の送信者匿名性を保有すると考えられる．また Tor は Bifrost と同等の送信者匿名性を保有している．しかし 2.2.1 に示した通り大規模なネットワークに対応することは難しい．

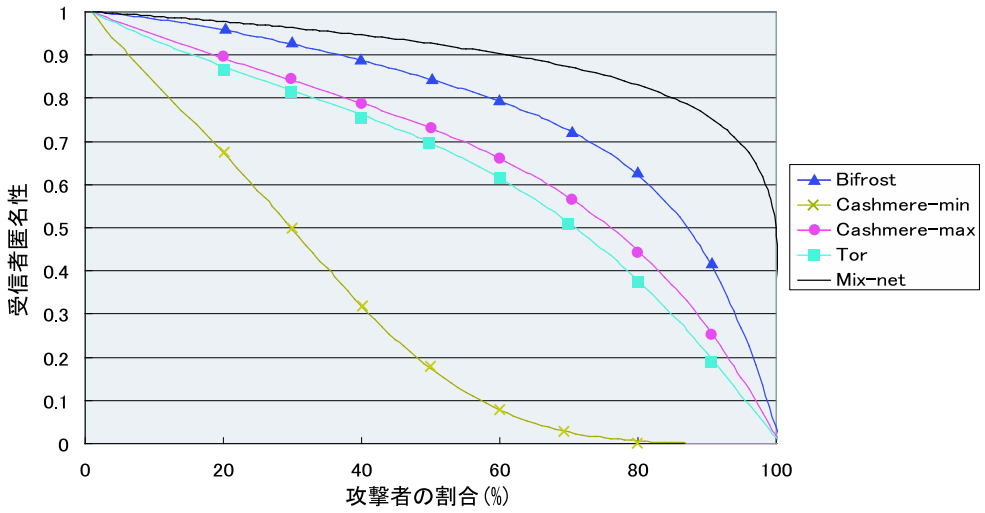


図 4.2: 受信者匿名性

4.1.3 受信者匿名性

提案手法は受信エリア内のどこかに受信者を設定する．そのため，攻撃者によって受信エリアの範囲を知られてしまうことは，受信者である確率の高いノードを知られてしまうことと同義である．ここで，攻撃者によって知られてしまった受信エリア内のノード群を S とする．ノード群 S 内には攻撃者であるノード群 S_1 と攻撃者でないノード群 S_2 が存在する．つまり，$S = S_1 \cup S_2$，$|S_1| = f|S|$，$|S_2| = (1-f)|S|$ である．また，受信エリア内の全ノード数は Lg である．S_2 内のノードは S 外のノードに比べ受信者である可能性が高い．評価のため，L と g が特定されたとすると S_2 内のノードの受信者である可能性は，$\frac{1}{Lg-f|S|}$ となる．また S 外のノード群が受信者である確率は $1 - \frac{(1-f)|S|)}{Lg-f|S|}$ である．よって，あるノード u が受信者である確率 p_u は以下の式で評価できる．

$$p_u = \begin{cases} \frac{1}{Lg-f|S|} : & u \in S_2 \\ (1 - \frac{(1-f)|S|}{Lg-f|S|}) \times \frac{1}{N-|S|} : & u \notin S \end{cases}$$

この式を用いて Bifrost，Cashmere-max,Cashmere-min，Tor，Mix-net の受信者匿名性を評価した．送信者匿名性と同様に $n(= 1, 2, ..., L)$ で場合分けし，それぞれの発生確率 G を乗算することで計算した．評価結果を図 4.2 に示す．

Mix-net は，前項で説明した通り常にシステム内の全てのノード対してブロードキャストを行うことで送信者，受信者を秘匿する方法である．そのため RTT は非常に低速であり，本論の背景にある大規模な匿名通信システムには適応することは現実的でない．しかし，Mix-net 内のノードの受信者である確率は常に $\forall u \in \Omega$，$p_u = \frac{1}{N-|S|}$ をとる．これは，Mix-net が匿名システムにおいて常に最大の匿名性をとることを表し，Mix-net の曲線がある攻撃者の割合での理想の値を示している．

また Bifrost は Tor，Cashmere に比べ高い受信者匿名性を保有していることが確認できる．Tor と Bifrost の差は受信者の設定方法の違いにより生じたと考えられる．Tor における受信者は常に経路の終点であるが，Bifrost においては必ずしも受信者は経路の終点ではない．受信エリアの導入により，受信者を経路の途中に配置できることで受信者の情報が漏洩しにくくなったと考えられる．

Cashmere との違いは受信者を配置することのできる受信エリアの作成方法の違いにより生じたと考えられる．Cashmere は Bifrost と同様に経路の途中に受信者を配置できる．しかし，Cshmere はルートノードと呼ばれるメッセージの復号，グループ内ノードへメッセージのマルチキャストを行うノードが存在する．そのためルートノードが攻撃者であると，その時点で受信エリアが特定されてしまう．一方 Bifrost は，受信エリアの始点 (As) が前受信エリアからメッセージを受けとり，受信エリアの終点 (At) がメッセージの復号を行う．そのため As と At 単体が攻撃者であっても受信エリアを特定することができない．As と At 両方が攻撃者であって初めて受信エリアを特定することができる．この責任の分散により，Bifrost の受信者匿名性は Cashmere を上回ったと考えられる．

4.2 局所的から見た匿名性の検証

4.2.1 単体での通信解析攻撃

本手法における攻撃者の位置を文献 [7] を参考に以下の様に (図 4.3 参照) 分類した．

(1) Chord のルーティングを行う際の中継ノード (受信エリア外)

(2) 受信エリア内 (A_i) かつ A_it, A_is でないノード

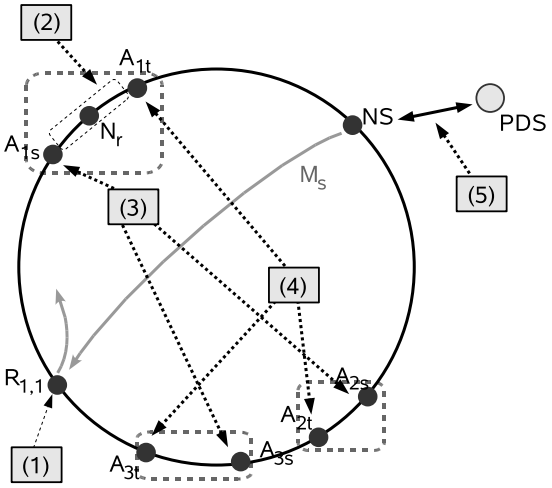


図 4.3: 攻撃者の位置

(3) 受信エリアの始点ノード (A_is)

(4) 受信エリアの終点ノード (A_it)

(5) 送信者とヘッダ部とデータ部公開鍵サーバとの通信

それぞれのノードを局所的に観測された場合の匿名性の考察を行う．また，(5) に関する通信は暗号化されていること, さらに PDS は十分に信頼できると仮定する．よって, ここでは対象外とする．

(1) 中継ノード $R_{i,j}$ の監視

Chord 中間ノードは，ノード管理層におけるノード検索時に利用されるノードである．このノードが知ることのできる情報は，ノード検索を要求したノードと，検索対象の ID である．提案手法の場合，検索を要求行うノードは受信エリア終端ノードであり，検索対象の ID が次受信エリアの始点ノードである．よって，このノードを監視される，もしくは攻撃者である場合，当該ノードの前後のエリアの一端を知ることができる．しかし，複数ある受信エリアの一端のみの情報であるため，送受信者を知ることには直結しない．また，より強固な匿名通信を望む場合，ノード検索を SSL など

暗号化することで，保護することができる．

(2) 受信エリア内かつ A_it，A_is でないノードの監視

ここに分布するノードは少なくとも前後のノードが受信エリア内であることを知ることができる．受信者は受信エリアのどこかに配置されるため，そのためこの前後のノードは無作為に選んだノードより受信者である可能性が比較的高い．また送信者も，受信エリア内のノードに紛れて通信するため，そのためこの前後のノードは無作為に選んだノードより送信者である可能性が比較的高い．しかし，ノード単体では受信エリア数自体を知ることができず，送信者，受信者を特定されることに直結することはない．

(3) 受信エリアの始点ノード (A_is) の監視

ここに分布するノードは前受信エリアからメッセージを受けとる．そのため前受信エリアを知ることができ，メッセージを中継してきたノードは受信エリア内にいることが分かる．また，当該ノードの Successor は受信エリア内であることも知ることができる．4.2.1 で記述した通り，受信エリア内にいると知ったノードは無作為に選んだノードより送信者，受信者である可能性が比較的高い．しかし，ノード単体では受信エリア数自体を知ることができず，送信者，受信者を特定されることに直結することはない．

(4) 受信エリアの終点ノード (A_it) の監視

ここに分布するノードはメッセージを復号する．そのため次の受信エリアと知ることができる．よって，少なくとも次受信エリア内のノードを 1 つは知ることができる．また，当該ノードの Predecessor は受信エリア内であることも知ることができる．4.2.1 で記述した通り，受信エリア内にいると知ったノードは無作為に選んだノードより送信者，受信者である可能性が比較的高い．しかし，ノード単体では受信エリア数自体を知ることができず，送信者，受信者を特定されることに直結することはない．

(5) 公開鍵サーバとの通信の監視

公開鍵サーバには，全参加ノードが中継ノードや受信者の公開鍵を要求する．したがって，公開鍵サーバの通信を監視することで，全参加ノードの通信路の様子を知ることができる．これを防ぐには，1) 問い合わせを SSL などで保護すること，2) ダミーとして不必要なノードの鍵を取得する，もしくは別のタイミングで以前取得した公開鍵を使用する．この 2 つの方法で保護することができる．

4.2.2 結託攻撃

上記複数のノードが中継攻撃を共有した場合を述べる．本章では，共有する情報は結託した中継ノードの役割 (復号などの処理) とパケットの内容を共有できるという仮定のもと考察する．

まず匿名通信路を特定するには，メッセージの同一性を確認する必要がある．ここで，受信エリアの終点ノードが攻撃者ではない状況を想定する．多重暗号においてパケットは，復号されてから次に復号されるまで同じであり，一度復号されると，パケットの内容は変わってしまう．そのため，提案手法では受信エリア終点から次の受信エリアの終点までが同一パケットである．つまり受信エリアの終点ノードが攻撃者でない場合，解析できるのはこの区間のみであり，通信路全体を解析することも，送信者と受信者に関する情報を得ることもできない．

次に，受信エリア終点ノードが結託している場合を想定する．受信エリア終点ノードはメッセージを復号して中継するため，次の受信エリアを通るパケットの内容を知ることができる．つまり，上記で示した解析できる区間を連結することができる．もし全受信エリアの終点ノードが結託をした場合，通信路全体を知ることができる．しかし，匿名通信路の起点である送信者は攻撃者になりえないことから，受信エリアの数がいくつあるのかということに対して確証を得られない．つまり，結託によって知りえた区間の起点が送信者なのか，それより以前に受信エリアが存在したのかということに対して確証をえられない．そのため，この攻撃で確実に送信者もしくは受信者を特定することはできない．

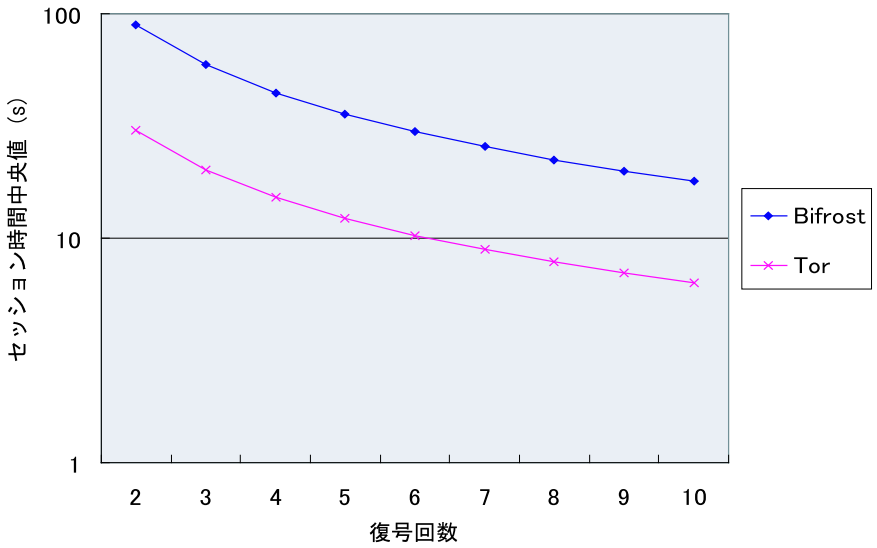


図 4.4: 予測される経路持続時間

4.3 耐離脱性検証

既存研究 [22][23][24] ではノードの激しい離脱と参加 (churn) において，セッション時間を評価の指標に用いられている．本章では，これらと同様にノードのセッション時間を用いて，ノードの耐離脱性を検証する．評価するネットワークはランダムに出入りのあるものを用いる．そのため，1 ノードのセッション時間は，パラメータ μ の指数分布に従う．

今回評価するネットワークのモデルは P2P ネットワークの耐離脱性検証の既存研究 [22][25] を参考に設定した．評価するネットワークモデルを以下に示す．

- ネットワークの大きさを $\Omega(= N)$ とする．また，ノードの出入りは同等であり，ネットワークの大きさは常に一定である．
- セッション時間はパラメータ μ の指数分布に従う．よって，ノードの離脱は $N\mu$ のポアソン過程に従う．また，以上より 1 ノードの平均セッション時間は $\frac{1}{\mu}$.

また上記の条件よりノードが時間 τ に離脱する確率 p は，$p = 1 - e^{-\mu\tau}$ である．これを τ について解くと，$\tau = (1/\mu)\ln 2$．$p = 1/2$(ある時間 τ の間にシステム内のノー

表 4.1: 既存手法との比較

手法	Crowds	Tor	Cashmere	Bifrost
送信者匿名性	○	○	△	○
受信者匿名性	×	○	○	◎
つながりの匿名性	△	○	○	◎
可用性	△	×	◎	○
スケーラビリティ	×	△	○	○
応答時間	◎	○	○	△
鍵管理コスト	○	×	-	△

ドの半分が離脱したことを示す) と仮定すると，システムの中央値は $\ln 2/\mu$ となる．

既存研究である P2P ネットワークを用いたファイルシステム値 [24] では，ノードのセッション時間の中央値 ($= 60min$) という実測値を用いて評価されている．受信エリア終点とバックアップノードが共にいなくなる時，経路が切断されたとして，予測される経路の持続時間を評価した．評価には μ を用いて，受信エリアとバックアップノードがいなくなる確率を近似して計算し，また上記の実測値 (中央値 $= 60min$) を用いた．

計算結果を図??に示す．結果より，Tor に比べ経路の持続時間は長くなると予想される．これにはバックアップノードの配置により，例え復号するノードが離脱しても経路の切断に直結しないためだと考えられる．また，上記の評価はバックアップノードが受信エリア終点になった場合，バックアップノードを再配置することを考えていない．そのため，バックアップノードの再配置を行えば更に飛躍的に経路の持続時間は増加すると考えられる．

4.4 既存手法との比較

Tor，Crowds，Cashmere と提案手法である Bifrost との，送信者の匿名性, 受信者の匿名性, 送受信者のつながりの匿名性, 可用性とスケーラビリティを比較する．それぞれの比較をまとめたものを表 4.1 に示す．

4.4.1 Crowds

Crowds は 2.2.2 に示した通り, ある確率で受信者か中継ノードかを選んでメッセージを送信することで, 送信者の匿名性を確保する手法である．確率によって次のノードを決めるため, 一度も他のノードを経由せず, 直接受信者と通信を行った場合であっても, 送信者の匿名性は確保される．しかし，受信者に到達することを保証するために, 受信者の情報は秘匿されない．つまり, 受信者の匿名性の確保は考えられていないため, 受信者は中継したすべてのノードに知られる．

次に可用性について述べる．Crowds は確率によって次のノードを決めるため, 動的に受信者までの経路を確保することができることから, 高い可用性があると言える．しかし, 返信は, 送信時に中継したノードが送られてきたノードに返すことで行う．つまり, 中継ノードのどれか 1 つでも離脱した場合, 通信が途絶える．この場合, 送信者がタイムアウト処理等を行い, 再送信を行わなければならない．以上のことから, 送信時においては高い可用性があるが, 双方向通信に着目した場合, 可用性が高い通信を行えるとは言えない．

スケーラビリティについて述べる．Crowds はグループのメンバシップを専用のサーバによって集中的に管理する．サーバによって集中管理する方式は, ノード数の増加に対してスケーラブルに動作することができない．このためスケーラビリティは低いと言える．

4.4.2 Tor

Tor は 2.2.1 に示した通り, メッセージを多重暗号化することで送信者と受信者を共に秘匿させる手法である．これにより, 1 つでも信頼できる中継ノードがある場合, 送信者と受信者を共に知ることができない．そのため, 送信者と受信者のつながりの匿名性も確保される．また, 経路生成時にそれぞれの中継ノードとの共通鍵を生成するため, 本手法に比べ高速であると考えられる．

しかし, 経路生成時に固定的に経路を決めるため, 可用性が低い．送信時と返信時ともに中継ノードが一つでも離脱した場合, メッセージを復号することができず, 再度経

路生成からやりなおす必要がある．また, 経路生成のためにディレクトリサーバと呼ばれるサーバが，ネットワークの構成各ノードの IP アドレスや，現在故障などしていないかなどの状況の把握，各ノードの公開鍵などを把握しておく必要がある．このコストは非常大きく，スケーラビリティは低いと考えられる．

4.4.3 Cashmere

Cashmere は提案手法と同様に DHT を用い，復号するノードのバックアップを用意することからスケーラビリティは高いと考えられる．また，2.2.4 で記述した通り，受信するグループ全てに対して復号する鍵を配布するため，グループ内のノード全てが離脱してしまわない限り経路が切断されることはない．そのため非常に高い可用性を保有していると考えられる．しかし，グループ内全ノードに復号の鍵を配布することは，それだけ多くの情報を漏洩することと同義であり，4.1 で示した通り匿名性は低くなる．

また鍵の管理コストについては，文献 [9] に記述がないため比較していない．しかし，Bifrost と同様に DHT のアルゴリズムにネットワーク構成の把握などを任せていることから，同様のコストではないかと考えられる．

第5章

評価

5.1 実装

実装には，オーバーレイ開発環境である OverlayWeaver[26][27] を利用した．OverlayWeaver には Chord のオーバーレイネットワーク構築機能，ルーティング機能，メッセージ送受信機能などが提供されている．このメッセージ送受信機能に 3.4 で示した多重暗号化処理，復号処理，経路制御処理を追加実装する形でシステムを実現した．主な構成要素を図 5.1 に示す．ここで実装した構成要素は「通信制御」と「暗号化処理，復号処理」の部分である．本方式では 1 度目の通信の際に，受信エリア終端のノードと受信者に共有鍵を配布し，中継したすべてのノードは自身のコネクション情報を登録する．以降のはコネクション情報をもとに匿名通信処理を行う．図 5.1 内の通信制御では, 送信者から指定されたアドレスを元に多重暗号化を用いてメッセージを生成する．また，メッセージを受信後，1 度目の通信であれば復号を試みることで自身が受信エリア内であるかどうかを適宜判断し，それに応じた経路制御を行う．このときの動作の流れを図 5.2 に示す．さらに，2 度目以降の通信はコネクション情報に従い復号や中継処理を行う．また，実装言語は Java を用いた．

5.1.1 性能評価

本方式における遅延時間と通信速度を調べる実験を行った．実験では，100Mbps のイーサネットにネットワークスイッチを介して接続した 32 台の計算機を用いて，環状

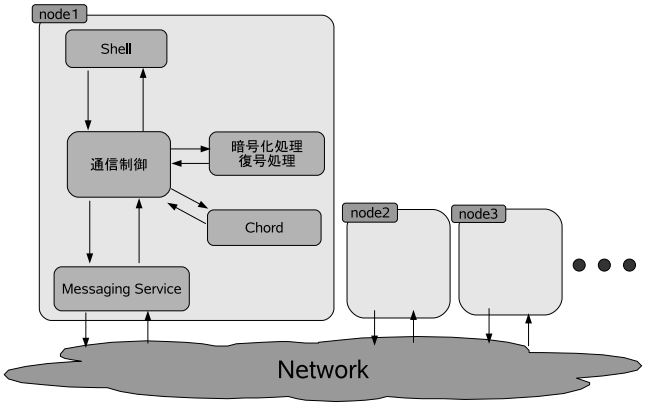


図 5.1: 本方式のコンポーネント

なオーバーレイネットワークを構成した．実験した計算機は Sempron/ 2800+，メモリ 1GB，OS は Linux である．32 台の計算機のうち，どの計算機も送信者と受信者になることができる．

今回行った実験の評価パラメータはメッセージサイズ (実験 1)，受信エリア数 (実験 2)，中継ノード数 (実験 3) であり，それぞれを変化させ評価を行った．また全てのノードを同一受信エリア内にある場合と異なる受信エリア内にある場合を比較した．なお 3.3.1 で示した制御メッセージは実装していない．以下に各実験の概要を示す．

実験 1 往路復路ともに受信エリア数を 2(往復で 4)，16 ホップの経路 (往路で 32 ホップ) を生成し，送信データサイズを $L_M = 1, 64, 128, 256, 512, 1024, 2048, 4096[KB]$ と変化させた場合の RTT と，暗号化処理と復号処理にかかる時間を計測する．

実験 2 経路を 16 ホップ (往復で 32 ホップ) と固定し，受信エリア数を m=1,2,3,4,5 と変化させた場合の経路生成時間と送信データ (100KB) に対する処理時間を計測する．

実験 3 受信エリア数を 5 と固定し，往復で経由するノード数 n=5,10,15,20,25,30 と変化させた場合の経路生成時間と送信データ (100KB) に対する処理時間を計測する．

実験 1 の結果を図 5.3 に示す．実験 2 と実験 3 の経路生成時間をそれぞれ図 5.4 の

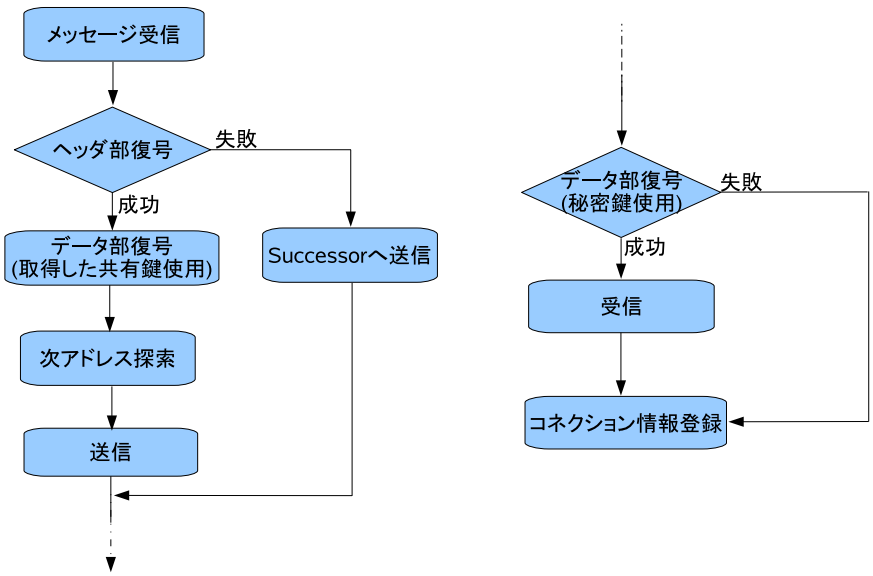


図 5.2: 動作の流れ

(a)，(b) に，共有鍵を用いた 100KB の匿名通信処理時間をそれぞれ図 5.5 の (a)，(b) に示す．

5.1.2 考察

図 5.3 から RTT はメッセージサイズに比例していることが分かる．しかし，送信データサイズが 1MB であっても 2 エリア (往復 4 エリア)，16 ホップ (往復 32 ホップ) の場合に 3.4 秒である．現在運用されている匿名通信である Tor[5] では，4 ホップで RTT が約 2 秒であり [28]，千田氏が実装した手法 [29] では 2 ホップで 1MB の通信で約 2 秒である．今回の実験では LAN 接続で中継数が 16 ホップであることを考慮に入れると，これらと大差はない．また，Web サービスとして見た場合，十分実用的な性能であると考えられる．

次に図 5.3 より約半分が暗号化処理時間と復号処理時間であることが分かる．そこで実験 2 と実験 3 により詳細な分析を行う．図 5.4 と図 5.5 を比較すると，全体的な処

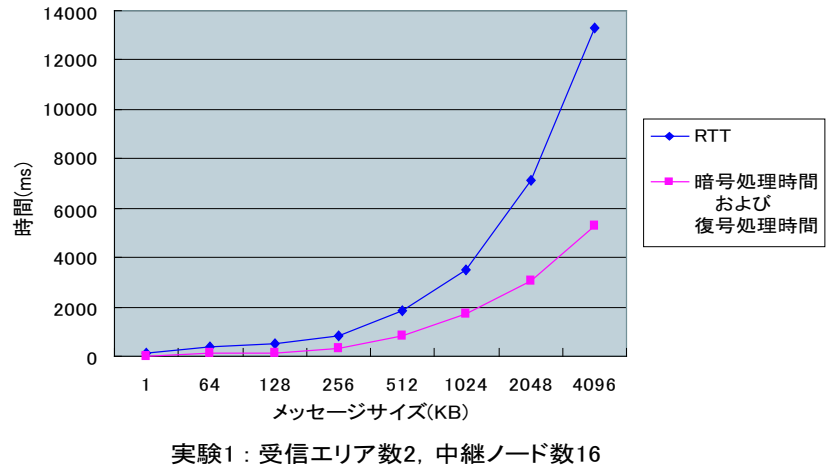
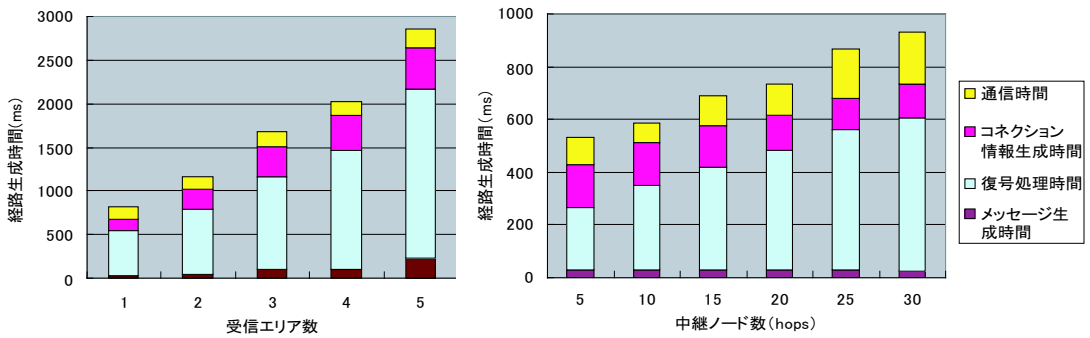


図 5.3: 通信データサイズによる RTT の変化と暗号処理と復号処理による処理時間の変化

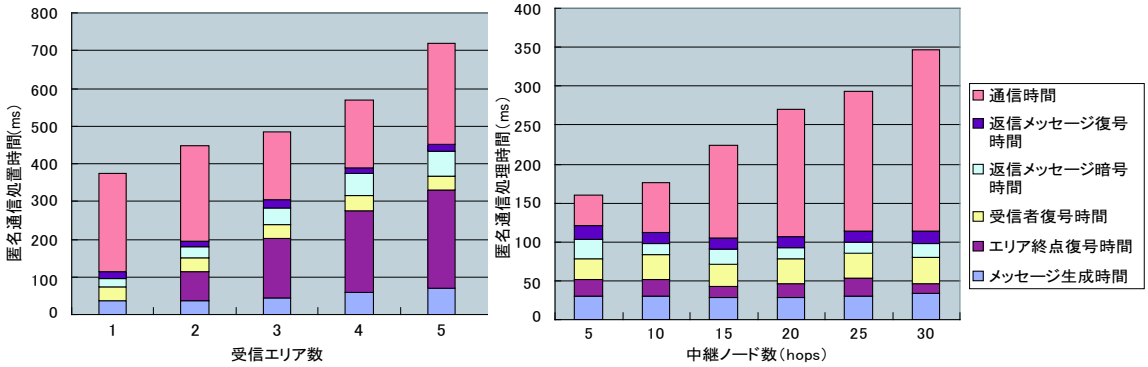
理時間は匿名通信処理時間の方が経路生成時間と比べ高速であることが分かる．これは，経路生成時は公開鍵を用いて暗号化処理と復号処理が必要だが，データ通信では共有鍵を用いるため高速な通信を行うことができるためである．図 5.4 と図 5.5 においてはそれぞれの (a) と (b) を比較すると，受信エリア数を変化させる (a) の場合はメッセージ生成時間に増加が見られたが，中継ノード数を変化させる (b) の場合は増加が見られなかった．これは，受信エリア数が多重暗号化の処理回数となるため，受信エリアの増加に伴い送信メッセージの暗号化処理時間が増加するためである．復号処理時間は図 5.4 の (a) の場合，(b) と比べて大幅に増加していることが分かる．受信エリアが増加することは復号することが可能なノード (= エリア終点ノード) が増えるため，そこでの復号処理コストが増加することを意味する．図 5.4 の (a) は (b) と比べて復号処理時間の増加量が多いことから，エリア終点ノードの処理コストは一般中継ノードのコストより大きいと考えられる．

ここで，受信エリア数と中継ノード数を決定するための指針について検討する．攻撃者が本方式の受信者を推定するには，まず受信者がいるエリアの推定を行い，その後にエリア内の受信者を推定するという手順が必要である．3.3.2 でも述べたように，



(a) 実験2:中継ノード数16　　(b) 実験3:受信エリア数 5

図 5.4: 経路生成時間



(a) 実験2:中継ノード数 16　　(b) 実験3:受信エリア数 5

図 5.5: 匿名通信処理時間

受信エリア数が増えると多重暗号の回数が増えるため，受信者がいるエリアの推定が難しくなる．逆に，受信エリア数が少ない場合は，受信者のいる位置を推定される可能性が高くなる．一方，中継ノード数が増えると受信者である可能性のノードが増えるため受信者の推定が難しくなる．以上から，秘匿性の高い通信を行う場合には，受信エリア数を増やし受信者エリアの推定自体を困難にする．また，秘匿性が比較的低く高速性が重視されるような通信では受信エリア数を減らして，中継ノード数を増や

すことで，一定の匿名性を確保しつつ高速な通信を行う．

なお，実験 2 と実験 3 からも暗号化処理時間と復号処理時間が全処理時間の大部分を占めていることが分かる．これは，本実装が Java を用いたことによると考えられる．C 言語を用いて，暗号化と復号処理を高速化することで，さらに高速な通信が期待できる．

5.2 Tor との比較

次に Bifrost の実用性を評価するために，現在実用されている Tor との比較実験を行った．比較対象の Tor は The Free Haven Project[10] よりバージョン 0.2.1.19 を取得し比較した．

実験では，100Mbps のイーサネットにネットワークスイッチを介して接続した 16 台の計算機を用いて，環状なオーバーレイネットワークを構成した．実験した計算機は Atom Z530，メモリ 1GB，OS は Ubuntu である．

以下に各実験の概要を示す．

実験 1 Tor，Bifrost ともにホップ数と復号回数をを m=1,2,3,4,5,6,7,8 と変化させた場合の経路生成時間と送信データ (100KB) に対する処理時間を計測する．暗号化処理と復号処理にかかる時間を計測する．

実験 2 Bifrost の経路を 8(往復 16) ホップと固定し，受信エリア数を m=1,2,3,4,5,6,7 と変化させた場合の経路生成時間と送信データ (100KB) に対する処理時間を計測する．

実験 1 の経路生成時間を図 5.6 に，共有鍵を用いた 100KB の匿名通信処理時間を図 5.7 に示す．続いて実験 2 の経路生成時間を図 5.8 に共有鍵を用いた 100KB の匿名通信処理時間を図 5.9 に示す．

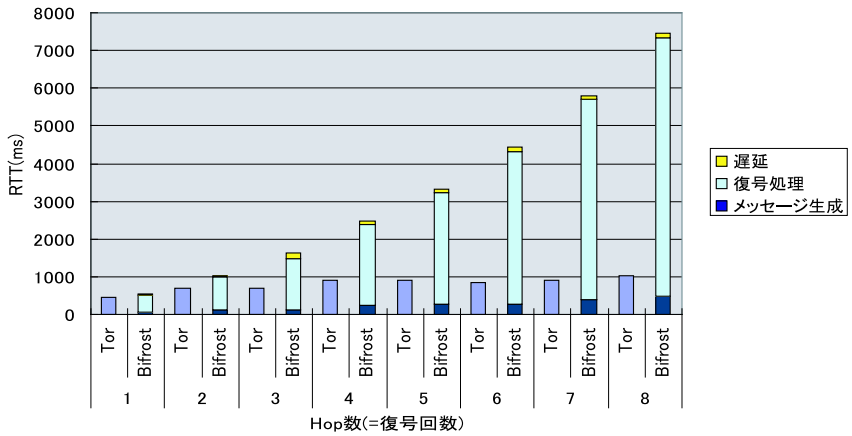


図 5.6: Tor との比較 (経路構築時間)

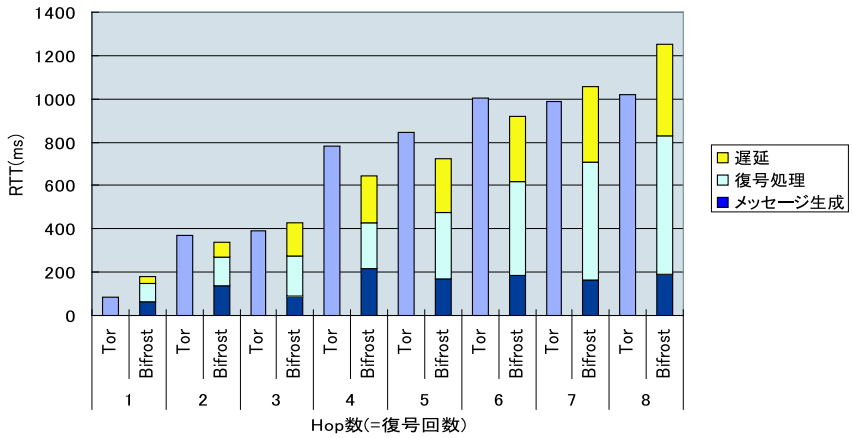


図 5.7: Tor との比較 (匿名通信処理時間)

5.2.1 考察

図 5.6 より，Bifrost は Tor に比べ経路を構築するために多くの時間を要していることが分かる．この原因の一つに往復経路の違いが挙げられる．Tor は往路，復路ともに同じノードを経由するが Bifrost は全くの別経路をとる．そのため，Tor は往路の際に一度経路を構築してしまえば，復路の際経路を構築する必要がない．一方，Bifrost は復路の際も往路と同様に経路の構築を行う必要がある．そのため，Bifrost の経路構築時間は Tor に比べ，理論上二倍の経路構築時間が必要である．これが，両者の経路構築時間の差の原因の一つである．

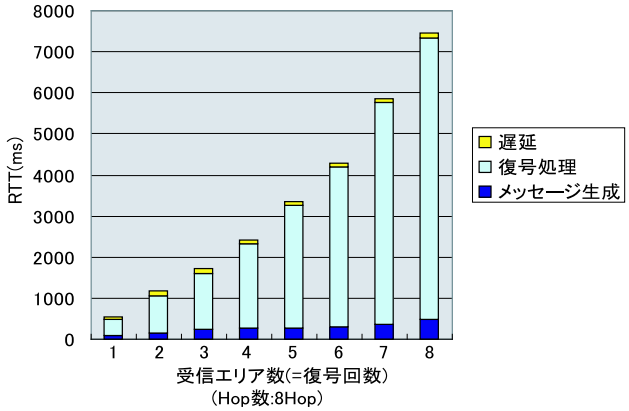


図 5.8: 受信エリアを変化させた際の経路構築時間 (8 ホップ)

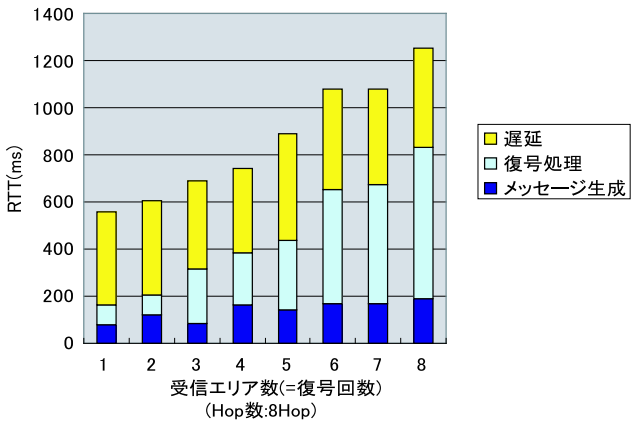


図 5.9: 受信エリアを変化させた際の匿名通信処理時間 (8 ホップ)

また，Bifrost の復号コストの高さも一因である．ホップ数 1 の場合の経路構築時間を比較すると Bifrost，Tor の間にはあまり差がない．しかし，ホップ数と復号回数が増えるにつれ，Biforst の経路構築時間は大幅に増加し，両者の経路構築時間は五倍以上の差が生じている．この大幅な増加は，復号処理時間によるものであることが分かった．一方，Tor は C 言語を用いた実装であるため比較的復号処理が高速であること，さらに中継するメッセージを 512B ずつ分割し，並列に中継処理を行っているため増加率が低いと考えられる．そのため今後 Tor と同様に，Bifrost の暗号化と復号処理を高速化すること，中継メッセージの Cell 化などを行うことで，経路構築時間を二倍まで近づけることが可能であると考えられる．

次に図 5.7 より，経路構築時間では両者に五倍以上の差があったが，匿名通信処理時間を比較すると大差ないことが分かる．これは Bifrost が経路を構築した際にコネクション情報を保存したこと，共有鍵を配布したことにより，両者の復号コストの差が小さくなったためだと考えられる．これにより，Bifrost の実用性は十分だと考えられる．

最後に Biforst は Tor とは違い，ホップ数と復号回数を独立に制御できる．つまり Bifrost は受信エリアの導入することで，ホップ数以下の復号回数を設定することができる．前述した通り，図 5.8，5.9 は，ホップ数を 8 で固定し，それ以下に受信エリア (=復号回数) を減らしていった場合の RTT である．受信エリアを減らした場合，復号回数が減ることにより復号のコストが削減されるため，Tor と同ホップ数でも更に高速に通信できていることが分かる．これにより，ユーザが自由に通信経路を設定することができ，各々の目的に適した柔軟性のある匿名通信を構築することが期待できる．

第6章

まとめと今後の課題

本論文では，多重暗号化を用いた通信と Chord を用いたノードの管理二層による匿名通信方式 Bifrost を提案，実装，評価を行った．Bifrost は受信エリアと呼ばれるノード群を構成し，多重暗号通信を行う．そして，DHT やメッセージを復号するノードのバックアップの導入により，高い匿名性と可用性，拡張性の両立を実現した．

また Bifrost の匿名性を示す指標としてネットワークを定義し，送信者匿名性と受信者匿名性の解析を行った．Claudia らが提唱する評価法に準じた評価では，送信者匿名性と受信者匿名性共に既存手法と同等以上であることを確認した．また，可用性を評価することを目的として，一定の確率でノード離脱するネットワークにおいての経路持続時間を評価した．結果より既存手法に比べ，高いノード離脱耐性を持つことを確認し，可用性に優れていることを確認した．さらに，既存手法との比較を考察し Bifrost の優位点を明らかにした．次に Bifrost を実装し，通信におけるオーバーヘッドの評価を行った．評価結果より，Bifrost は多くのノードを経由するため暗号化と復号のコストが大きいことを確認した．そして Bifrost を用いる際の受信エリアと中継ノード数を決定する指針を提案した．また，現在実用されている Tor と通信速度を比較し，実用性を評価した．評価結果より，Bifrost は十分な実用性を秘めていることを証明した．

今後の課題としては，Bifrost の現在の実装は暗号化，復号のコストにまだまだ改善の余地があるため，暗号化と復号の高速化を行い実用性を更に検証していく予定である．パケットの Cell 化や鍵管理サーバの実装などの実装を行い，より一般的なモデルを提案し，より大規模で一般的な環境を用いた性能評価実験を行いたい．

謝辞

本研究のために多大な御尽力を頂き、日頃から熱心な御指導を賜った名古屋工業大学の齋藤彰一准教授に深く感謝致します。

また本研究に対し熱心に御検討、御協力を頂きました、松尾啓志教授に深く感謝を致します。

加えて、本研究の際に多くの助言、協力を頂いた斎藤研究室ならびに松尾・津邑研究室の皆様に深く感謝致します。

参考文献

[1] Pfitzmann, A. and Waidner, M.: Networks without user obserbility, Eurocrypt'85, LNCS 219, pp.245-253(1986).

[2] Chaum, D.L: Untraceable electronic mail, return address, and digital pseudonyms, Comm.ACM, Vol.24, No.2, pp.84-88, ACM Press(1981).

[3] Goldschang, D., Reed, M. and Syverson, P.: Onion routing for anonymous and private internet connections, Comm.ACM, Vol.42, No.2, pp.39-41(1999).

[4] 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).

[5] Dingledine, R. and Mathewson, N.: Tor : The Second-Generation Onion Router, Proceedings of 13th USENIX Security Symposium, pp. 303-320 (2004).

[6] Reiter, M.K. and Rubin, A.D.: Crowds:Anonymity for web transactions, ACM Trans.Information and System Security, pp.66-92(1998).

[7] 北澤 繁樹, 長野 悟, 双紙 正和, 宮地 充子: 初等的な環状経路を用いた匿名通信方式, 情報処理学会論文誌, Vol. 41 No.8, pp.2148-2161 (2000).

[8] 阿部 洋丈, 加藤和彦: Aerie:www のための完全分散匿名プロキシ, 情報処理学会論文誌, Vol. 46 No. SIG3, pp. 51-61 (2005).

[9] Zhuang, L. Zhou, F. Zhao, B. and Rowstron: Cashmere: Resilient Anonymous Routing. In Proc. of Nerworked System Design and Implementation(2005).

[10] The Free Haven Project: Tor: anonymity online, http://www.torproject.org/.

[11] 井上 大介，松本 勉: マルチキャストを用いた匿名通信方式, 電子情報通信学会技術報告，ISEC-99-29 (1999).

[12] Kikuchi, H.: Sender and Recipient Anonymous Communication without Public Key Cryprography, 情報処理学会研究報告書，98-CSEC-1-8 (1998).

[13] I.Stoica,R.Morris,D.Karger,M.Kaashoek, and H.Balakrishnan: Chord:A Scalable Peer-toPeer Lookup Service for Internet Applications, Proc.ACM SIGCOMM,pp.149-160. (2001).

[14] Rowstron, A and Druschel, P.: Pastry:Scalable.decentralized object location and routing for large-scale peer-to-peer systems, Proc.Middleware (2001).

[15] Zhao, B. Y. et al.: Tapestry: A Resilient Global-Scale Overlay for Service Deployment, Journal on selected area in communications, Vol 22, No.1, pp.41-53(2004).

[16] S.Ratnasamy, P.Francis,M.Handley,R.Karp, and S.Shenker: A scalable content addressable network, In Proc. ACM SIGCOMM'01, San Diego, CA, Aug.(2001).

[17] 首藤一幸, 加藤大志, 門林雄基, 土井裕介: 構造化オーバーレイにおける反復探索と再帰探索の比較. 情報処理学会研究報告, 2006-OS-103(2006).

[18] Dosz, C., Seys, S., Claessens, J , and Preneel, B.: Towards measuring anonymity. In Proc. of PET (2002).

[19] Serjantov, A., and Danezis.G : Towards an information theoretic metric for anonymity. In Proc of PET(2002).

[20] S.Rhea, D.Geels, T.Roscoe, and J.Kubiatowicz.: Handling churn in a DHT.IN Proc. USENIX'04, (2004).

[21] Pfitzmann, A., and Kohntopp, M.: Anonymity, unobservability nad pseudonymity. In Proc. of the Intl. Workshop on the Design Issues in Anonymity and Observavility(2000).

[22] Rhea, S., Geels, D., Roscoe, T., and Kubiatowicz, J.: Handling churn in a DHT. In Proc of USENIX(2004).

[23] Gummadi, K. P., Dunn, R. J., Satoiu, S., Gribble, S. D., Levy, H. M., and Zahorjan, J.: Measurement, modeling and analysis of a peer-to-peer file-sharing workload. In Proc. of SOSP(2003).

[24] Satoiu, S. Gummadi, K. P., Dunn, R. J., and Gribble, S. D.: A measurement, study of peer-to-peer file sharing workload. In Proc. of MMCN(2002).

[25] D.Liben-Nowell, H.Balakrishnen and D.Karger: Analysis of the evolution of peer-to-peer system.(2002).

[26] OverlayWeaver: An overlay construction toolkit,http://overlayweaver.sourceforge.net/.

[27] 首藤一幸, 田中良夫, 関口智嗣.: オーバーレイ構築ツールキット Overlay Weaver. 情報処理学会論文誌:コンピューティングシステム, Vol.47, No.ACS15, (2006).

[28] Andriy Panchenko, Lexi Pimenidis, and Johannes Renner.: Performance Analysis of Anonymous Communication Channels Provided by Tor, The Third International Conference on Availability, Reliability and Security, pp. 221-228 (2008).

[29] 千田 浩司, 小宮 輝之, 塩野入 理, 金井 敦: 不正者追跡可能な匿名通信方式の実装と評価, 情報処理学会研究報告書, 2005-CSEC-29 , pp. 25-30 (2005).