2009－MBL－48 (4)
2009／1／29

アドホックネットワークにおける共通鍵共有方式の提案と実装・評価

森 郁海† 高橋 修‡

† 公立はこだて未来大学大学院 システム情報科学研究科

‡ 公立はこだて未来大学

概要 Mobile Ad-hoc NETwork(MANET)は，無線技術を用いたマルチホップ通信により固定インフラが存在しない状況でもネットワークを構築することが可能である．一方で，情報セキュリティとネットワークセキュリティ双方の確保も重要な課題である．これら二つのセキュリティにおける重要な技術として，非対称鍵(公開鍵)暗号と対称鍵(共有鍵)暗号がある．特に，前者はディジタル署名などに利用され，強固なセキュリティを提供している．しかしながら，このような非対称鍵暗号は暗号化/復号化に際して多くの計算資源を必要とし，主にモバイルノードで構成される MANET には不向きである．本稿では，非対称鍵暗号が利用可能な状況において，計算資源を消費しにくい対称鍵暗号を MANET 上で利用する場合に必要な，秘密共有方式について議論する．具体的には，ノードディスジョイントな経路 n 本に共通鍵を公開鍵で暗号化したデータを送信する方式(共通鍵暗号化)，共通鍵を(k,n)秘密分散法を用いて n 本の経路へ送信する方式((k,n))，共通鍵を宛先の公開鍵で暗号化し更に(k,n)秘密分散法を適用し n 本の経路へ送信する方式(共通鍵暗号化+(k,n))を定義し，シミュレータを用いて評価する．また，攻撃耐性を定性的に評価し，これらの方式の安全性を議論する．

The Study of Common-key Secret Sharing Method in Ad-hoc Networks

Ikumi MORI† Osamu TAKAHASHI‡

† Graduate School of Future University–Hakodate, Systems Information Science

‡ Future University–Hakodate

Abstract Mobile Ad-hoc NETwork(MANET) that is multi-hop communication using radio link can build a network in the situation that there is not fixed infrastructure. However, both Information security and Network security in MANET are important problems. As important technology in these two securities, there are an asymmetric (Public-key) cryptography and a symmetric cryptography. For example, the former is used for Digital Signature and provides for strong security. However, such an asymmetric cryptography needs a lot of calculation resources on the occasion of coding / decoding. Thus, an asymmetric cryptography is unsuitable for MANET consisting of a mobile node. In this paper, we argue about Common-key Secret Sharing (CSS) Method that is necessary when using a symmetric cryptography that needs low calculation resources in MANET in the situation that asymmetric cryptography can use. We defined Three methods of CSS; (1) Encrypted Common-key Method, (2) (k,n) Method, (3) (k,n) + Encrypted Common-key Method. First of all, all methods create *n* Node-disjoint paths. Next, (1) and (2) encrypts a common-key using a public-key of the destination node. Then, (2) and (3) applies (k,n) Secret Sharing Method against a common-key or encrypted Common-key. Finally, all method sends each secret data using *n* Node-disjoint paths. These three methods are evaluated on the Network Simulator NS2. In addition, we evaluate attack tolerance qualitatively and argue about the safety of these methods.

1. はじめに

現在の有線ネットワーク，あるいは無線ネットワークにおいて，情報セキュリティとネットワークセキュリティ双方の確保は重要である[1]．これら二つのセキュリティにおける重要な技術として，非対称鍵(公開鍵)暗号や対称鍵(共有鍵)暗号がある．特に，前者はディジタル署名などに利用され，強固なセキュリティを提供している．しかしながら，このような非対称鍵暗号は，暗号化/復号化に際して多くの計算資源を必要とする[2]．アドホックネットワークを構成するノード(モバイルノード)は，一般的に電源として小型のバッテリーを用いている[3]．アドホックネットワークにおいて，小さな電源しか持たないモバイルノードが，多くの電力を消費する非対称鍵暗号を頻繁に利用することは好まれない．また，文献[1]からわかるように，セキュアルーティングプロトコルの多くが対称鍵暗号の利用を要求条件としている．対称鍵暗号は非対称鍵暗号に比べ計算資源が圧倒的に少なく，さらに処理が高速でありエンド間で非対称鍵暗号とほぼ同等なセキュリティを提供することができる(RSA1024[bit]鍵長は，共通鍵暗号における80[bit]鍵長と等価の安全性のレベル)[4]．しかしながら，対称鍵暗号の利用には事前にエンド間で共通の秘密を共有する必要がある．

本稿では，非対称鍵暗号の利用を軽減するために，対称鍵暗号をアドホックネットワークに適用する場合に必要な共通鍵(共有すべき秘密)の共有方法について議論する．具体的には，ノードディスジョイントな経路を n 本作成し，(k,n)秘密分散法[5]などを用いて共通鍵をセキュアに共有する 3 方式を提案し，シミュレータで評価する．3 方式とは，共通鍵暗号化方式，(k,n)方式，共通鍵暗号化+(k,n)方式である．

次章以降の構成は次のようになる．2 章で関連研究を述べ，3 章で提案方式のアルゴリズムを説明する．そして，4 章でシミュレーション評価と考察を行い，5 章でまとめを述べる．

2. 関連研究

この章では，既存の共通鍵の配送方法であるハイブリッド暗号方式と(k,n)秘密分散法，アドホックネットワーク上でのノードディスジョイントな経路作成方法に関する研究を説明する．

2.1. ハイブリッド暗号方式

ハイブリッド暗号方式[6]は，暗号/復号化処理が高速である共通鍵暗号方式と，鍵の配送(配布)が容易な公開鍵暗号方式の双方の利点を利用したものである．ハイブリッド暗号方式では，まず，送信者は共通鍵で平文を暗号し，暗号文を作成する(図 1)．さらに，平文の暗号化に用いた共通鍵を宛先の公開鍵で暗号化する．送信元は，暗号文とこの暗号化した共通鍵を宛先へ配送する．宛先は，まず最初に受信した暗号化された共通鍵を自身の秘密鍵で復号化し，共通鍵を得る．そして，その共通鍵を用いて暗号文を復号化して，平文を得る．このように，ハイブリッド暗号化方式では，共通鍵暗号方式と公開鍵暗号方式の双方の利点を利用しており，非常に効率的である．

本稿では，この方式の共通鍵の配送部分だけを利用する．攻撃が行われた場合でも到達率が高く，さらにセキュアである必要がある．

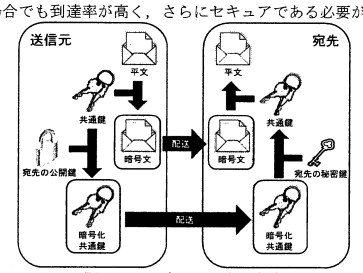

図 1 ハイブリッド暗号方式

2.2. (k,n)秘密分散法((k,n)閾値法)

(k,n)秘密分散法[5]は閾値秘密分散法の一種で，ある情報を n 個の分割し，そのうち k 個の分割情報があれば秘密情報を復元できる手法である(図 2)．n,k は，n,k∈ℕでかつ n≧k を満たす．

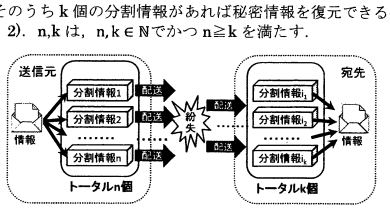

図 2 (k,n)秘密分散法

分割情報は秘密情報とは相関性がないため，分割情報を k 個集めない限りは秘密情報を復元できないという情報量的安全性を有する．ア

ドホックネットワークでは，盗聴等が困難な場合は中継ノードとして攻撃者が k 本の経路に同時に侵入しない限り復元されない．分割には，(k-1)次元多項式を利用する．今，k=2 の場合を考えると，1 次元多項式となる．即ち，$f(x) = ax + b$(但し，$a \neq 0$)を用いる(図 3)．1 次元多項式は，任意の 2 点の座標があればパラメータa, bを復元できる．秘密情報をbとすると，分割情報は n 個の座標となる． n 個の座標を$(x_1,\ f(x_1))$, $(x_2,\ f(x_2))$, …$(x_n,\ f(x_n))$とすると，その中の 2 点$(x_c,\ f(x_c))$, $(x_d,\ f(x_d))$ $(n \geq c, d \geq 1, c \neq d)$を得ることができれば，$a = (f(x_c) - f(x_d))/(x_c - x_d)$, $b = f(x_c) - ax_c$とa, bを計算できる．同様に，k>2 の場合も多項式の切片を分割したい情報とすれば良い．

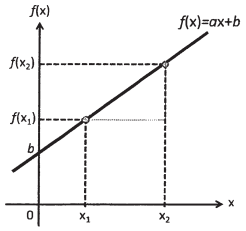

図 3 k=2 の場合の多項式：$f(x) = ax + b$

本稿では，ネットワークトポロジを 1km² に 100 ノードと想定し，ノードディスジョイントな経路の構築具合から k=2,n=4 とする(図 4(公開鍵長 2048[bit]固定時))．グリッド，ランダムは，図 15 に従う．$2 \leqq n \leqq 3$では，高エラーレートな環境の場合，分割情報の損失が約 33%以内でなければならず到達率が低下すると考えられる．従って，損失が 50%まで耐えられるように n=4 とした．

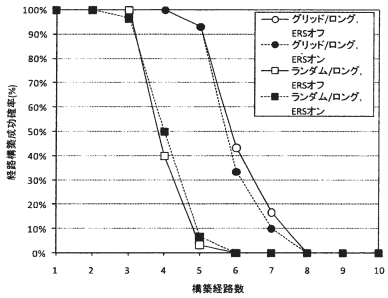

図 4 本稿で使用する環境における経路構築確率

2.3. ノードディスジョイントな経路作成方法

アドホックネットワークの通信では，送信元，宛先以外にデータ中継に第三者ノードを介する．シングルパスルーティングの場合，ループ回避がなされているので中継ノードとなる第三者ノードが重複することがない．一方，マルチパスルーティングの場合，ある経路とその経路と異なる経路の中継ノードが共有している場合がある． ノードディスジョイントな経路とは，マルチパスルーティング時において，各経路間で中継ノードが重複しない状態を指す(図 5)．

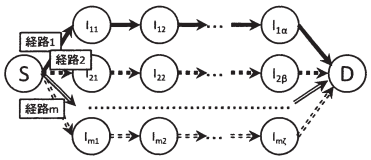

図 5 ノードディスジョイントな経路例

ノードディスジョイントな経路の作成法として NAF (Node Association Factor)値[7]を利用する方法がある．図 5 のような経路の場合，経路 1 から見た経路 2 との共有している中継ノードはない($I_{1a} \neq I_{2b}, 1 \leq a \leq \alpha,\ 1 \leq b \leq \beta$). さらに，経路 1 から見た経路 3,経路 4,…, 経路 m にも共有している中継ノードはない($I_{1a} \neq I_{3c}$, $I_{1a} \neq I_{4d}, \ldots, I_{1a} \neq I_{mz},\ 1 \leq c \leq \gamma,\ 1 \leq d \leq \delta,\ \ldots,\ 1 \leq z \leq \zeta$). 同様に経路 i から見たときの経路 i+1,経路 i+2,…,経路 m $(2 \leq i \leq m-1)$の共有中継ノード数を調査していく．NAF 値はこれらの共有中継ノード数の合計であるので， NAF 値が 0 であれば送信元から宛先までの経路がノードディスジョイントである．

3. 提案手法

本稿におけるアドホックネットワークでの秘密共有方式は，以下の 2 フェーズからなる．

(1) 経路構築フェーズ　このフェーズでは，NAF 値を利用してノードディスジョイントな経路を 4 本以上作成する．

(2) 秘密共有フェーズ　このフェーズでは，まず(1)で作成した 4 本の経路それぞれに分割情報，あるいは暗号化した共通鍵を送信する．そして，宛先が各秘密共有方式(共通鍵暗号化, (2,4), 共通鍵暗号化+(2,4))で共通鍵の復号化に成功した場合，送信元へ向かい共通鍵の復号化の成功を示す ACK を送信する．また，送信元は指定時間内に ACK を受信できなかった場合，分割情報あるいは暗号化された共通鍵の再送(リトライ処理)を行う．

以降でこれら 2 フェーズの詳細を述べる．

3.1. 経路構築フェーズ

本稿では，ネットワークに流れるデータに対して，パケットの長さに応じたエラーが発生するような環境を想定する．従って，既存のルーティングプロトコルでは適切な経路構築が難しい．本稿で提案する SAODV-NDMR(Secure AODV - Node Disjoint Multipath Routing)は，このような環境においてもノードディスジョイントな経路を複数本作成できるように，SAODV[8]の基本機能に加え AODV[9]の Expanding Ring Search (ERS)をマルチパス拡張した機能を備える．複数回の Route Discovery によって，エラーによる RREQ(Route Request)パケットのドロップに対するロバスト性を強化している．また，ERS を行わない場合でも，秘密共有時にはバックオフ値による Route Request 送信間隔の増加を無視することで同様の働きをする．経路形成は，以下に示す 6 ステップからなる．

(1) 送信元による RREQ 送信

SAODV-NDMR は，SAODV を継承しているので，RREQ に送信元による署名が施されており，さらにハッシュチェインの利用によって変更可能領域が保護されている．従って，中継ノードによる RREQ 情報の偽造は即座に検出される．しかしながら，RSA による署名を行う場合，公開鍵長に比例して署名長も増加するのでエラーの影響を受けやすい．送信元は，RREQ を図 6 に従い複数回送信する．従来の SAODV と異なる点は，図 6 の灰色部分である．マルチパス拡張を行うため，収集経路数のチェックと RREQ の送信回数(reqs)及び RREQ_ID の加算タイミングを変更している．そして，NET_TRAVERSAL_TIME は本来 reqs が加算されるごとに輻輳回避のためのバックオフ値が乗算されるが，SAODV-NDMR では経路構築を優先させるためバックオフ値を乗算しない(2 × NODE_TRAVERSAL_TIME × NET_DIAMETER とする)．RING_TRAVERSAL_TIME　(2 × NODE_TRAVERSAL_TIME × TTL)は，従来通りである．

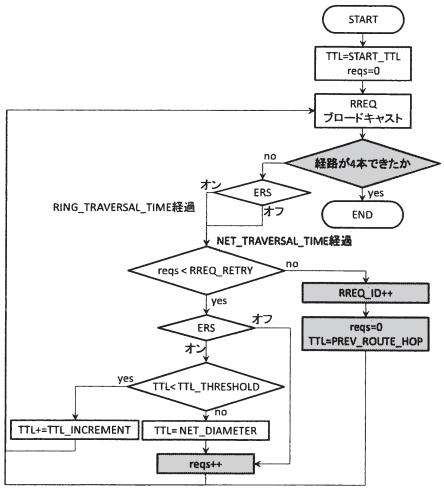

図 6 RREQ 送信処理

(2) 中継ノードによる RREQ 受信と処理

● RREQ の受信

中継ノードはまず始めに，送信元の公開鍵を使用して RREQ の署名を検証し，署名が有効であることを確認する．署名が有効な場合は

RREQ を受信し，有効でない場合は破棄する．

● **NAF 値の計算，逆経路の作成**

RREQ には，送信元を含む中継したノードを記録したソースリストが記載されている．RREQ を受信した中継ノードは，このソースリストを使用して自身から送信元までの経路の NAF 値を計算する．NAF 値が 0 ならば，その RREQ を許可し，逆経路(送信元への経路)を作成する(図 7)．また，複数回 Route Discovery が行われるので，過去に許可した経路と同じ経路を通って来た RREQ は無条件に許可する．

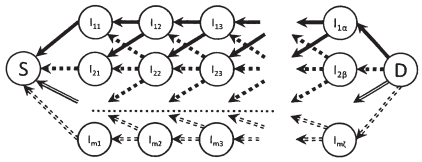

図 7 RREQ による逆経路の作成

● **NAF 値の再計算，逆経路の更新**

通常 NAF 値の計算は，受信した RREQ と過去に許可した経路に対して行われる．しかしながら，図 8 のように 1 度目の検索で，冗長な経路を発見してしまった場合，本来ノードディスジョイントな経路が 2 本できなければならないが，1 本しか作成できない．つまり，最適な経路形成が行われていれば検索できるはずの経路が許可されず，ノードディスジョイントな経路の発見数が減少してしまう．

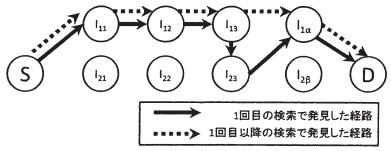

図 8 NAF 値の再計算が発生する例

この問題を回避するために，NAF 値の再計算(更新)を行う．今，図 8 のようなトポロジを考えると，中継ノード$I_{1\alpha}$で NAF 値の再計算が発生する．1 度目の RREQ で発見された経路の中継ノード集合を$\{R_1^1|I_{11},I_{12},I_{13},I_{23}\}$，1 度目以降の RREQ で発見された経路の中継ノード集合を$\{R_2^1|I_{11},I_{12},I_{13}\}$とすると，$R_2^1 \subset R_1^1$であることがわかる．このような時，部分集合である$R_2^1$は，$R_1^1$より冗長でない経路である．$I_{1\alpha}$は，現在保持している$R_1^1$を$R_2^1$に置き換え，逆経路を更新する．さらに，$I_{1\alpha}$は$R_2^1$とそれ以外の全ての保持経路に対して，NAF 値が 0 になるように経路を削除する．この NAF 値の再計算は，D でも同様に発生する．

● **RREQ フォワード処理**

中継ノードは，RREQ を許可し逆経路を作成または更新すると，自身のアドレスを RREQ にソースリストに追加し，再ブロードキャストする．

(3) 宛先による RREQ 受信と処理

● **RREQ の受信**

宛先が RREQ を受信すると，中継ノードと同様に送信元による署名を検証し，受信処理を行う．

● **NAF 値の計算，逆経路の作成**

中継ノードと同様に， NAF 値が 0 ならばその経路に対し逆経路を作成する．

● **NAF 値の再計算，逆経路の更新**

中継ノードと同様に，NAF 値の再計算が発生すると逆経路を更新する．宛先はさらに，最短経路を優先的に使用するように，送信元への経路のネクストホップノードをホップ数の短い順にソートする．

(4) 宛先による RREP 送信と処理

● **RREP の送信**

宛先は RREQ を許可すると，送信元への逆経路を作成し RREP (Route Reply)を送信元へ送信する．RREQ と同様に，RREP は宛先によって署名される．RREP には，RREQ の宛先と宛先の隣接を除くソースルートがコピーされ，その後 RREP を RREQ が通ってきた逆経路に対し送信する．

● **RREP の定期送信**

過去に送信された RREP がエラーによりドロップしている可能性があるので，直近の RREP 送信時刻から RREP_RESEND_TIME が経過するか，新しい有効な RREQ が到着する毎に，過去に送信した RREP を再送信する(図 9)．但し NAF 値の再計算が発生した際に，削除された経路に対しては RREP を送信しない．RREP の再送は，秘密分散共有のパケット(分割情報)あるいは，暗号化された共通鍵が宛先に到着した時点で停止する．SAODV-NDMR では，RREQ と同様に宛先による署名が施され，さらにハッシュチェインの利用によって変更可能領域が保護されている．

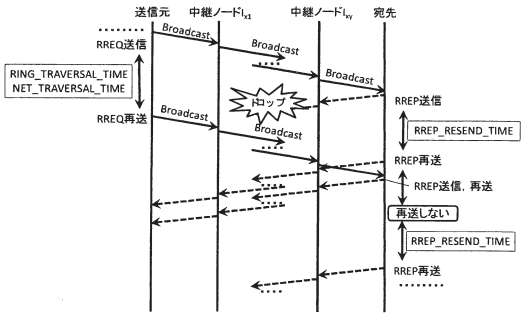

図 9 RREP の再送シーケンス

(5) 中継ノードによる RREP 受信と処理

● **RREP の受信**

中継ノードは，まず宛先の公開鍵を使用して RREP の署名を検証し，署名の有効性を確認する．署名が有効な場合は RREP 受信し，有効でない場合は破棄する．

● **逆経路の作成/更新**

RREP を受信すると，宛先への逆経路を作成し双方向経路とする(図 10)．既に経路が作成されている場合は，今の RREP の逆経路に更新する．そして，RREP のソースルートから次に送信すべきネクストホップを調べ，RREP をフォワードする．各中継ノードは送信元への経路を複数保持している可能性があるので，送信元への経路のネクストホップ(主経路)を RREP をフォワードしたノードに更新する．

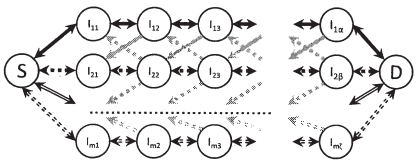

図 10 RREP による逆経路の作成

(6) 送信元による RREP 受信と処理

● **RREP の受信**

RREP が送信元へ到着すると，中継ノードと同様に署名を検証し署名の有効性を確認する．署名が有効な場合は RREP を受信し，有効でない場合は破棄する．

● **逆経路の作成/更新**

RREP を受信すると，送信元は宛先への逆経路を作成する．宛先と同様に，宛先への経路のネクストホップノードをホップ数の短い順にソートする．

● **Route Discovery の続行/停止**

送信元は，指定された経路数(本稿では n=4 本)が集まれば検索を終了し，集まっていなければ検索を続行する(ステップ(1)に戻る)．

3.2. 秘密共有フェーズ

ノードディスジョイントな経路が 4 本発見できた時点で，送信元は秘密共有フェーズへ移行する．このフェーズでは，秘密共有方式ごとの仕様に従って暗号化されている，あるいは暗号化されていない共通鍵を秘密情報として秘密分散法を適用し，経路構築フェーズで形成した経路 4 本を使用して秘密共有を行う．この節では，各方式の特徴を説明した後，秘密共有フェーズの詳細を述べる．

3.2.1. 秘密共有方式の特徴

本稿では，共通鍵暗号化方式，(2,4)方式，共通鍵暗号化+(2,4)方式について比較を行う(表 1)．秘密情報を宛先が復元するために必要な分割情報数は，共通鍵暗号化方式で 1 つ，その他の方式では 2 つである．共通鍵暗号化方式と共通鍵暗号化+(2,4)方式では，送信元は共通鍵を宛先の公開鍵で暗号化し秘密分散法を適用するため，攻撃者による盗聴が不可能である．分割情報 1 つあたりの大きさは，宛先の公開鍵長によって変化する(表 1，図 25)．

表 1 秘密共有方式の仕様一覧

	共通鍵暗号化方式	(2,4)方式	共通鍵暗号化+(2,4)方式
秘密分散法	(1,4)閾値法	(2,4)閾値法	(2,4)閾値法
使用する多項式	$f(x) = b$	$f(x) = ax + b$	$f(x) = ax + b$
秘密情報,b	宛先の公開鍵で暗号化した共通鍵	暗号化していない共通鍵	宛先の公開鍵で暗号化した共通鍵
必要経路数	4	4	4
復元に必要な分割情報数	1	2	2
各経路に流れる分割情報	経路1〜4: b (宛先の公開鍵で暗号化した共通鍵)	経路1: 座標$(x_1, f(x_1))$ 経路2: 座標$(x_2, f(x_2))$ 経路3: 座標$(x_3, f(x_3))$ 経路4: 座標$(x_4, f(x_4))$	経路1: 座標$(x_1, f(x_1))$ 経路2: 座標$(x_2, f(x_2))$ 経路3: 座標$(x_3, f(x_3))$ 経路4: 座標$(x_4, f(x_4))$
送信元による署名	あり	あり	あり
分割情報1つの大きさ [bit]	$C(Common_key)_{length}$	$x_{length}+Common_key_{length}+1$	$x_{length}+C(Common_key)_{length}+1$
盗聴	不可能	可能	不可能

表 1 での分割情報 1 つの大きさを表す $C(Common_key)_{length}$ は，暗号化後の共通鍵のサイズである．(2,4)閾値法を用いる方式では，座標$(x, f(x))$を表現するのに必要なビット数は以下のように計算している．

$$f(x) = ax + b,\ \begin{cases} a_{length} = 32[byte](256[bit],\ ランダム数) \\ b_{length} = 秘密情報 b のビット数[bit] \\ x_{length} = 32[byte](256[bit],\ ランダム数) \end{cases}$$

$$f(x)_{length}\ [bit] = \log_2(ax + b) = \log_2(2^{a_{length}+x_{length}} + 2^{b_{length}})$$

ここで，$b_{length} \geq a_{length} + x_{length}$ ならば，

$$2^{a_{length}+x_{length}} + 2^{b_{length}} < 2^{b_{length}+1}$$

従って，$f(x)_{length}\ [bit] < \log_2(2^{b_{length}+1}) = b_{length} + 1$

また，$b_{length} < a_{length} + x_{length}$ ならば，

$$2^{a_{length}+x_{length}} + 2^{b_{length}} < 2^{a_{length}+x_{length}+1}$$

従って，$f(x)_{length}\ [bit] < \log_2(2^{a_{length}+x_{length}+1}) = a_{length} + x_{length} + 1$

よって，

$$\begin{cases} f(x)_{length}\ [bit] = b_{length} + 1[bit], & b_{length} \geq a_{length} + x_{length} \\ f(x)_{length}\ [bit] = 513[bit], & b_{length} < a_{length} + x_{length} \end{cases}$$

(2,4)方式での分割情報は，公開鍵の長さに依存しないので，将来的に公開鍵長が増加したとしても情報の大きさが変化しない．(2,4)閾値法を用いる方式では，ネットワークエラーあるいはデータの破棄を目的とする単独の攻撃者によってパケットがロストしたとしても，合計(n-k)個のロストまで耐えることができる(但し k<n)．攻撃者が送信元近傍で盗聴を行うと 2 つ以上の分割情報を取得できる確率が高まるが，(2,4)方式を除く方式では共通鍵を復号化されることはない．盗聴に脆弱な(2,4)方式の秘密情報は，共通鍵そのものではなく，Diffie-Hellman 鍵交換法[10]での公開数などのような第三者に公開できる情報が望ましい．公開鍵などの第三者に公開できない秘密情報を利用する場合は，リンク間やエンド間で暗号化(図 11)を行うことで，盗聴を防止することができる．

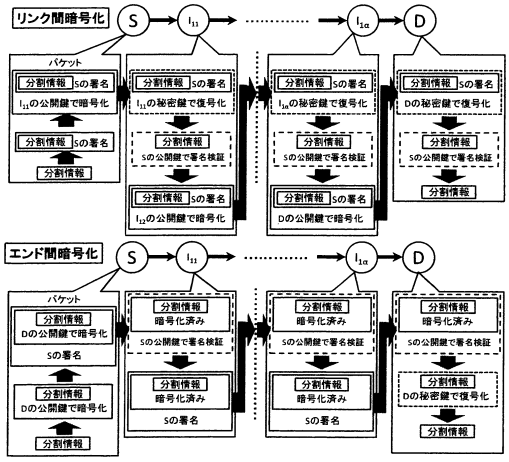

図 11 盗聴防止のための暗号化例

しかし，このような暗号化を用いる場合はパケットの大きさが公開鍵長に依存して増大する可能性がある．また，パケットの暗号化では基本的に公開鍵長を超える平文(電文)の暗号化となるため，平文を分割するか電文の暗号化に適した非対称暗号(楕円曲線暗号[11][12]など)を利用すべきである．共通鍵暗号化+(2,4)方式は，共通鍵暗号化方式と異なり情報を分散配送するので，暗号文(暗号化された共通鍵情報)そのものも攻撃者へ与えにくい．従って，暗号文単独攻撃[13]などが困難になる．

3.2.2. 秘密共有フェーズの詳細

秘密共有フェーズは，以下の 3 ステップからなる．

(1) 送信元による分割情報の計算と送信

送信元は，表 1 の仕様に従い秘密情報に対して秘密分散法を適用し，分割情報に相当する座標を計算する．その後，計算した分割情報を含むパケットが作成され，送信元によって署名される．これらの署名されたパケットを，ノードディスジョイントな 4 本の経路それぞれに送信する．中継ノードおよび宛先は，まずこれらのパケットの署名を検証し，中継ノードは有効な署名であった場合のみ宛先に向けフォワードする．

(2) 宛先による ACK の送信

宛先は，分割情報あるいは暗号化された共通鍵を必要数受信し，共通鍵の復元に成功すると図 12 のように隣接に向けて共通鍵が復元できたことを示す宛先によって署名された ACK をブロードキャストする．宛先からの ACK を受信した隣接(中継ノード)は，まず署名を検証し ACK を送信元へ向けユニキャストする．

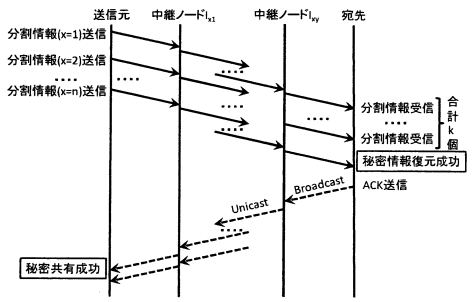

図 12 ACK 送信シーケンス

(3) 送信元によるリトライ処理

送信元は，SHARED_SECRET_RETRY_TIME 内に ACK を受信できない場合，宛先への分割情報が n-k 個以上ロストしたとみなし図 13 のように分割情報あるいは暗号化された共通鍵の再送を行う．宛先は，1 度目に i 個の分割情報($D_i, (1 \leq i < k)$)を受信したとすると，2 度目以降は残りの $j(j \geq k - i)$個(但し，$D_j \neq D_i$)の分割情報を受信できれば，秘密情報を復元することができる．復元に成功すると，図 12 に従い ACK を送信する．送信元は，ACK を受信することにより宛先と秘密共有が完了したとみなし，再送処理を停止する．

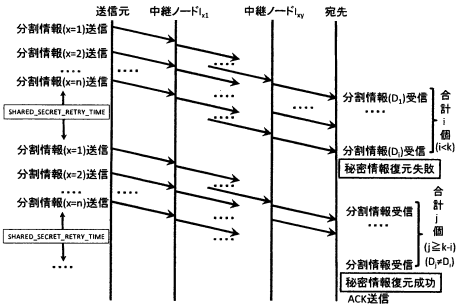

図 13 秘密分散共有再送シーケンス

4. シミュレーション

AODV-UU[14]を拡張した A-SAODV (Adaptive SAODV)[15]をベースとして SAODV-NDMR を実装し，ネットワークシミュレータ ns2[16]を用いて秘密共有をシミュレートする．暗号化，及び巨大整数ライブラリには，libgcrypt[17]を用いる．このシミュレーションにより，SAODV-NDMR を用いた秘密共有が可能かどうか(経路構築成功確率，

秘密共有成功確率)を調査し，可能な場合はその性能(経路構築処理時間，構築経路のホップ数，秘密共有完了時間)を評価する．シミュレーションパラメータを表 2 示す．

表 2 シミュレーションパラメータ

実験方式	共通鍵暗号化, (2,4), 共通鍵暗号化+(2,4)
非対称暗号方式	RSA
Public_key$_{length}$(公開鍵長)	256[bit], 512[bit], 1024[bit], 2048[bit]
Common_key$_{length}$(共通鍵長)	256[bit]
α_{length} ($f(x)=\alpha x+b$ のαのビット長)	256[bit]
x_{length} ($f(x)=\alpha x+b$ のxのビット長)	256[bit]
試行回数，シミュレーション時間	各30回，各30秒
最大バックアップ経路数	30(保持経路総数31)
Expanding Ring Search (ERS)	オン，オフ オン:START_TTL=3 オフ:START_TTL=NET_DIAMETER
RREQ_RETRY	2
TTL_INCREMENT	2
TTL_THRESHOLD	7
NET_DIAMETER	35
RREP_RESEND_TIME (RREP再送間隔)	500[ms]
SHARED_SECRET_RETRY_TIME (秘密共有リトライ間隔)	500[ms]
Byte Error Rate(BER)	0.00025
ノード密度	100 [ノード/1km^2]
モビリティ	グリッド(0[m/sec]), ランダム(20[m/sec], Random Waypoint)
トラフィックディスタンス	ロング，ショート

各ノードの In Queue と Out Queue には Byte Error Rate (BER) を設定する．本実験では，BER = 0.00025 とする．BER=0.00025 における Packet Error Rate (PER，パケットが破棄される確率)とパケットが 1 ノードを通過する確率(通過率)は，$PER = 1-(1-BER)^{Packet_size\,[Byte]}$，通過率 $=(1-PER)^2$ と計算される．本実験で用いる RREQ，RREP，RERR，分割情報の各パケットは署名されているため，公開鍵長により PER 及び通過率が変化する(図 14)．ノード密度は 100 ノード/1km^2，ノード配置はグリッド (モビリティなし)とランダム(Random Waypoint[18]，20[m/sec], Pause time 0[sec])である．トラフィックディスタンスは，ショートとロングとした(図 15)．

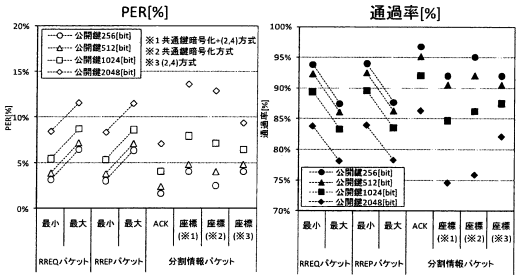

図 14 BER=0.00025 における PER と通過率

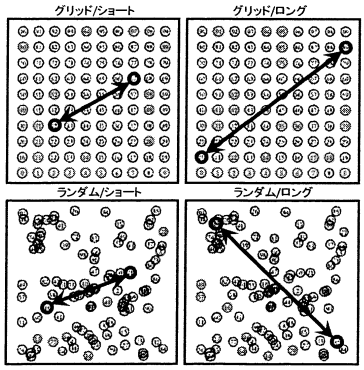

図 15 トラフィックディスタンス(グリッド,ランダム(初期配置))

4.1. 評価メトリック

SAODV-NDMR を以下の 5 つの基準によって性能評価する．

- **経路構築成功確率**　秘密共有に必要な経路が構築できた確率の平均値である．本稿では，必要な経路数 n は 4 本である．
- **秘密共有成功確率**　経路構築が成功した時の秘密共有が成功した確率(=秘密共有成功回数/経路構築成功回数)の平均値である．
- **経路構築処理時間**　3 方式に必要な経路(n 本)が構築されるまでに要した時間の平均値である．
- **ホップ数**　構築した n 本の各経路の平均ホップ数である．本稿ではホップ数を中継ノード数+1 と定義する．各方式で同じ経路構築方法を用いるので，全ての方式の平均値をプロットする．
- **秘密共有処理時間**　経路構築が完了した後，送信元が秘密共有処理を開始してから，宛先からの秘密共有の完了を示す ACK が送信元へ届くまでの平均時間である．

各評価メトリックは，図 16 のような位置づけとなる．

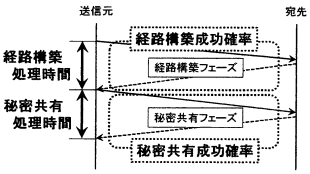

図 16 評価メトリックの位置づけ

4.2. 予備実験

秘密分散共有で特に重要な経路構築のパフォーマンスを相対的に評価するため，SAODV-NDMR の性能評価を行う前に経路の構築に関して他のマルチパスルーティングプロトコル(MRP)との比較実験を行う．比較する MRP は，AODV ベースの AOMDV[19]と DSR[20]ベースの DSR-NDMR[21]である．構築経路数は 4 本，トポロジはグリッド/ショートとし，RREQ のサイズ[Byte]によってどれだけ性能が変化するかを見るため，経路構築成功確率と経路構築処理時間を測定する．SAODV-NDMR と AOMDV，DSR-NDMR の比較結果を図 17 に示す．SAODV-NDMR は，本稿の実験に使用する RREQ のサイズの範囲のみプロットしている．AOMDV は，RREQ のサイズが増加するにつれ，経路構築成功確率が徐々に低下していくのがわかる．また，経路構築処理時間もそれに伴い増加していく．DSR-NDMR は，AOMDV に比べ RREQ のサイズ増加による経路構築成功確率の低下が抑えられている．実験で使用する範囲の RREQ サイズでは 100%である．経路構築処理時間も AOMDV より高速である．一方，SAODV-NDMR は経路構築確率 100%を維持し，経路構築処理時間も AOMDV，DSR-NDMR と比較して短い．時間の増加も緩やかである．

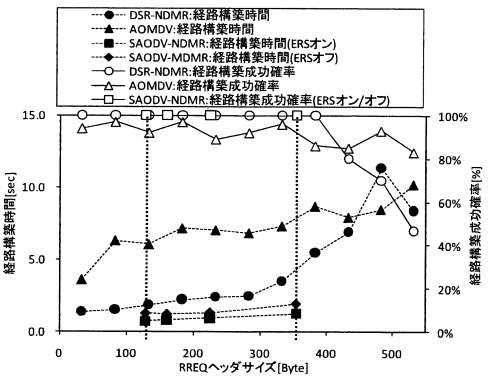

図 17 AOMDV,DSR-NDMR,SAODV-NDMR の予備実験結果

4.3. 結果と考察

4.1 での評価メトリックに基づいたシミュレーション結果とその考察を示す．

グリッド/ショート，ランダム/ショートにおける経路構築成功確率と秘密共有成功確率は，公開鍵長によらず 100%であった．ショートディスタンスにおける経路構築処理時間の結果を図 18 に示す．

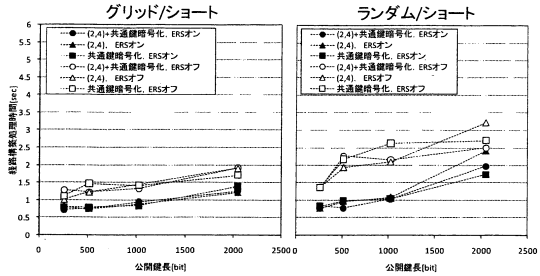

図 18 ショートディスタンスでの経路構築処理時間

どの方式でも経路構築プロセスは同じであるため，同じような傾向を示している．公開鍵長の増加によって，RREQ，RREP の署名領域が大きくなるので，緩やかに時間が増加する．グリッド/ショートでは 0.5〜2 秒の間で推移し，ERS を用いる方が 0.5 秒ほど高速である．ランダム/ショートもグリッド/ショートと同様に，公開鍵長の増加によって経路構築処理時間が増加する．ERS オン時で 0.5〜2.5 秒，ERS オフ時で 1.0〜3.5 の間で推移する．ERS を用いる方が 0.5〜1.0 秒ほど高速である．トラフィックディスタンスが短い場合は，ERS が有効であることが分かる．

ショートディスタンスにおけるホップ数の結果を図 19 に示す．

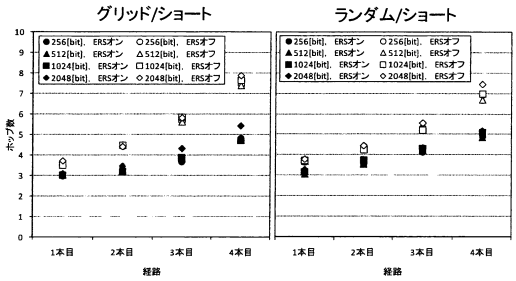

図 19 ショートディスタンスでのホップ数

公開鍵長によって，各経路のホップ数が大きく変化することはないが，公開鍵長が 2048[bit]まで増加するとホップが大きくなる傾向がある．グリッド/ショートを見ると，ERS が有効でない場合は 3〜8 ホップの経路を採用するのに対して，有効な場合は 3〜4 ホップと経路間のホップ数の差が少ない経路を発見している．ランダム/ショートでも，ERS が有効でない場合は 4〜7 ホップ程度するのに対して，有効な場合は 3〜5 ホップと短い経路を発見している．グリッド/ショートと同様に，3 本目以降の経路のホップ数の増加が抑えられている．PER はホップ毎に適用されるので，ホップ数の短い経路を発見する方が良い．

秘密分散共有処理時間の結果を図 20 に示す．グリッド/ショートでは公開鍵長の増加に対して，0.1〜0.4 秒の間で緩やかな増加傾向が見られる．これは，ERS の利用によってホップ数が低減されるが，それによる影響はほとんどないことを示している．ランダム/ショートでも公開鍵長の増加に対して，0.1〜0.6 秒の間で緩やかな増加傾向が見られる．移動性があるため，ホップ数の短い経路が切断されることがあり，各方式間で有意な差がなかった．

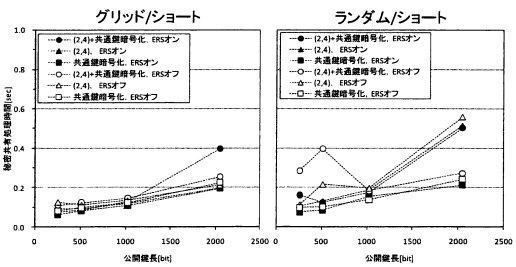

図 20 ショートディスタンスでの秘密分散共有処理時間

グリッド/ロングにおける経路構築成功確率と秘密共有成功確率は，公開鍵長によらずともにほぼ 100%であった．ランダム/ロングにおける経路構築成功確率と秘密共有成功確率を図 21 に示す．ロングディスタンスでは，経路構築に多くの時間を必要とする．そのため，シングルルートを用いる方式((1,1)+共通鍵暗号化方式)との比較を行う．図 21 から，移動性により経路構築がかなり困難になることが分かる．公開鍵長が 2048[bit]で 40〜50%まで低下する．一方，(1,1)+共通鍵暗号化方式は 100%の確率で経路構築が可能である．秘密共有成功確率は，経路構築が成功すれば 80%以上の確率で成功する．経路構築性能が秘密共有の成功に大きく影響することがわかる．

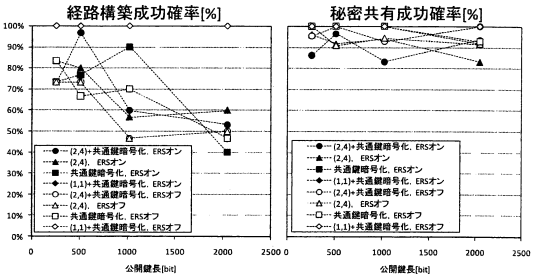

図 21 ランダム/ロングでの経路構築成功確率

経路構築処理時間の結果を図 22 に示す．公開鍵長の増加によって，各方式とも緩やかに時間が増加する．ロングディスタンスの場合，ERS を行っても TTL が小さいうちは経路が発見できないので，処理時間が増加する．グリッド/ロングでは，ERS を利用しない方が 2.5 秒ほどの高速である．しかし，構築経路数 n=1 の(1,1)+共通鍵暗号化方式を用いる方がさらに高速である．ランダム/ロングでも公開鍵長の増加によって，15〜20 秒の間で緩やかに時間が増加する．グリッド/ロングの場合と異なり，(1,1)以外の各方式間で差があまりない．これは，経路構築成功確率が低いためである．2000[bit]付近で時間が少し低下するが，これは経路構築失敗によって除かれた試行が多いためである．(1,1)+共通鍵暗号化方式は 2.5〜4.0 秒とかなり高速である．ロングディスタンスの場合，経路構築処理が深刻な問題となる．

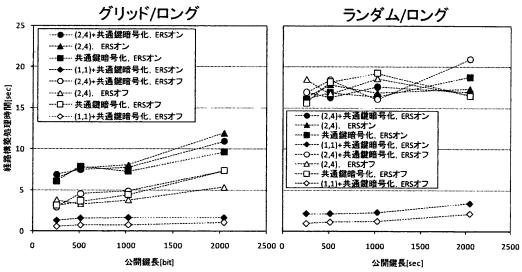

図 22 ロングディスタンスでの経路構築処理時間

ホップ数の結果を図 23 に示す．グリッド/ロングでは ERS が有効な場合，6〜9 ホップ程度，無効な場合は 7〜9 ホップ程度である．ERS を利用することによるホップ数の低下はほとんどない．

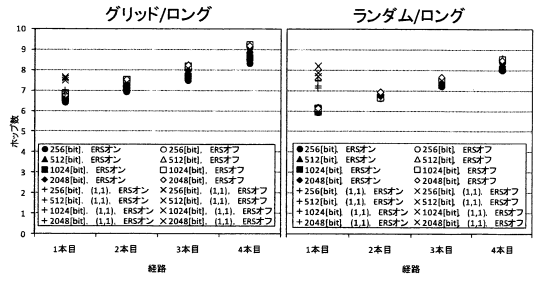

図 23 ロングディスタンスでのホップ数

しかし，(1,1)の場合はERSの利用で，1ホップほど短い経路を発見できる．(1,1)では，秘密共有で使用する経路が1本であり，その経路は最初に発見されたものであるため，よりホップ数の短い経路を最初に発見できるERSが，シングルパスでは有効に働くことを示している．ランダム/ロングの場合は，ERSのオン/オフに関わらず6～9ホップ程度である．(1,1)+共通鍵暗号化方式でERSを利用しない場合は，1ホップほど長い経路を用いている．

秘密分散共有処理時間の結果を図24に示す．グリッド/ロングでは公開鍵長の増加に対して，0.1～0.8秒の間で緩やかな増加傾向が見られるが，(1,1)+共通鍵暗号化方式ではエラーによるドロップ耐性がないため処理時間が安定していない．結果，(1,1)以外の方式と比較して処理時間が若干増加している．ランダム/ロングでは公開鍵長の増加に対して，多くの方式は緩やかな増加傾向にあるが，時間が減少しているものもある．これは，経路構築処理時間と同様に，対象となる試行が減少しているためである．また，(1,1)+共通鍵暗号化方式でERSがオフの場合は，エラーによるドロップにより処理時間が増加している．

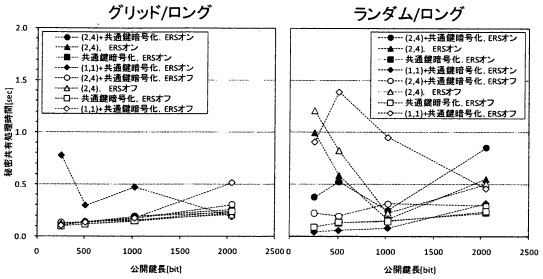

図 24 ロングディスタンスでの秘密分散共有処理時間

以上の結果から，SAODV-NDMRにおける秘密分散共有は，モビリティとトラフィックディスタンスにかなり影響されることがわかる．具体的には，トラフィックディスタンスが短い(3ホップ～6ホップ程度の経路を用いる)場合は，モビリティがあった場合も安定した動作するが，6ホップ以上の経路を用いるロングディスタンスでかつハイモビリティ(Random Waypoint, 20[m/sec], Pause Time 0[sec])環境下で，性能が著しく低下する．性能低下の主な原因は，経路構築の失敗であり，必要経路数nが大きいほど性能低下が顕著に現れる可能性がある．ERSの利用によるメリットは，トラフィックディスタンスやモビリティに依存するが，ホップ数の短縮である．デメリットは，ロングディスタンスの場合はホップ数の短縮効果があまりなく，経路構築処理時間が増大することである．秘密共有処理時間に関しては，(2,4)や共通鍵暗号化方式のほうが，(1,1)+共通鍵暗号化方式の様にシングルルートを用いる場合より若干短くなる．即ち，ネットワークエラーに対して耐性があると言える．

5. まとめ

本稿では，非対称鍵暗号の利用を軽減するために，対称鍵暗号をアドホックネットワークに適用する場合に必要な共通鍵(共有すべき秘密)の共有方法について議論した．具体的には，共通鍵暗号化方式，(2,4)方式，共通鍵暗号化+(2,4)方式を提案し，シミュレータを用いて性能評価した．その結果，トラフィックディスタンスが短ければ性能が安定的であるが，トラフィックディスタンスが長くモビリティが高い場合に性能低下すること定量的に示された．

今後は，ネットワークエラーによる影響を軽減するために，分割情報の縮小や攻撃耐性の定量評価が課題となる．そして，最適なk,nの算出アルゴリズムを求める問題も残されている．問題解決の指針として，分割情報の縮小手法と(k,n)閾値法の攻撃耐性について述べる．

5.1. 有限体上のランプ型秘密分散法を用いた分割情報の縮小

本稿での(k,n)秘密分散法では，秘密情報(共通鍵あるいは暗号化された共通鍵)より各々の分割情報のサイズが大きくなる($b_{\text{length}} < (x_{\text{length}} + f(x)_{\text{length}})$)．そのため，分割後の情報量は元の秘密情報量のn倍になり，かつ高エラー環境の場合，パケットドロップの可能性が高くなる．そこで，各分割情報$f(x)_{\text{length}}$が$f(x)_{\text{length}} = b_{\text{length}}/L$になる(k,L,n)ランプ型秘密分散法[22]を利用することを考える．(k,L,n)ランプ型秘密分散法は，(k,n)閾値法の拡張であり以下の条件を満たす($2 \le L \le k \le n$)．

- 任意のk個の分割情報から，秘密情報を復元できる
- 任意のk-t個($1 \le t \le L-1$)の分割情報から，秘密情報に関する情報が得られる
- k-L個以下の分割情報から，秘密情報に関する情報を得ることができない

分割には，(k,n)閾値法と同様に(k-1)次元多項式を利用する．秘密情報$\{\mathbb{S} | s_0, s_1, \ldots, s_{L-1}\}$とすると，$L, k, n \in F_q$[a]である．但し，$\max_{0 \le i \le L-1} s_i < q$となる素数であり，その有限体$F_q$上で，分割情報を生成する多項式を次のように定義する：$f(x) = (s_0 + s_1 x + s_2 x^2 + \cdots + s_{L-1} x^{L-1} + a_L x^L + \cdots + a_{k-1} x^{k-1}) \bmod q$ ，$(a_i \in F_q, a_i \neq 0, L \le i \le k-1, L < k)$．
今，L=k=2とすると，秘密情報$\{\mathbb{S} | s_0, s_1\}$となり

$f(x) = (s_1 x + s_0) \bmod q$ ，

$$\begin{cases} (s_i)_{\text{length}} = \mathbb{S}_{\text{length}}/2 \ ((s_0)_{\text{length}} = (s_1)_{\text{length}} \text{ のとき}) \\ q_{\text{length}} = 0 \sim q-1 \left(\max_{0 \le i \le 1} s_i < q, q: \text{素数}\right) \text{を表現するのに必要な領域} \\ x_{\text{length}} = \{x | \forall x \in F_q\}_{\text{length}} = q_{\text{length}} \end{cases}$$

$$f(x)_{\text{length}}\ [\text{bit}] = \log_2((s_1 x + s_0) \bmod q) = \log_2(2^{q_{\text{length}}})$$

従って，
$f(x)_{\text{length}}\ [\text{bit}] \le q_{\text{length}}$

$f(x)_{\text{length}}$に必要な領域は，q_{length} [bit]である．$(s_i)_{\text{length}}$が等しいとすると$(0 \le i \le L-1)$，$q_{\text{length}} = (s_i)_{\text{length}}$を満たせば，$f(x)_{\text{length}} = \mathbb{S}_{\text{length}}/L$となる．$\mathbb{S}$を共有すべき秘密を宛先の公開鍵で暗号化した情報とし，(k,n)閾値法および(k,L,n)ランプ型秘密分散法を適用しない場合と比較すると， 分割情報一つ当たりの情報の大きさは，共通鍵暗号化:ランプ型(k,L,n)秘密分散法=
$C(\text{Common_key})_{\text{length}} : (x_{\text{length}} + C(\text{Common_key})_{\text{length}}/L) = C(\text{Common_key})_{\text{length}} : (2 \times C(\text{Common_key})_{\text{length}}/L)$となる．即ち，$(2/L) < 1 (2<L)$ならば，ランプ型(k,L,n)秘密分散法を用いた方がパケットサイズを縮小できる(図 25)．

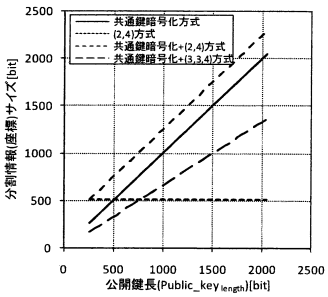

図 25 共通鍵暗号化+(3,3,4)方式での分割情報サイズの比較

5.2. (k,n)閾値法の攻撃耐性

アドホックネットワーク上で(k,n)閾値法を用いた場合の，攻撃ノードによってデータが復元される確率とブラックホール[23]ノードが存在した際の宛先のデータ復元率を定性的に評価する．

5.2.1. 攻撃者によるデータ復元確率

攻撃者が，送信元近傍での盗聴が困難である状況において，秘密分散共有過程で(k,n)閾値法によって分散される前の秘密情報をどれだけの確率で復元できるかを定性的に評価する．

まず，i本目の経路が安全である(中継ノードに攻撃ノードを一つも含まない)確率$P(R_i)$は，以下の式で計算可能である．ここで，N=総ノード数，A=攻撃ノード数，H_{ai}=i本目の経路のエンド間平均ホップ数(中継ノード数= H_{ai}-1)であり，Cは組み合わせ(Combination)である．

$$P(R_i) = \frac{(N-2-(i-1)(H_{ai}-1)-A)C_{(H_{ai}-1)}}{(N-2-(i-1)(H_{ai}-1))C_{(H_{ai}-1)}}$$

また，$\overline{P(R_i)} = 1 - P(R_i)$は，i本目の経路が危険である(中継ノードのうち少なくとも一つが攻撃ノードである)確率である．

今，エンド間で(k,n)閾値法を用いたとすると，攻撃ノードはk～n本の経路に入り込むことができれば，秘密情報を復元することができる．経路をn本作成するとき，各々の経路は安全であるか(安全:=○)，または危険(危険:=●)であるかの2状態({○, ●})に分類できる．n本

---

[a] F_qは有限体であり$F_q = \{0, 1, \ldots, q-1\}$でqは$\mathbb{S} < q$を満たす素数である．通常qを法とする集合を用い，演算に対しては，$(F_q; + \bmod q)$が加群(結合律が成立し，単位元，逆元が存在する)であり，$(F_q; \cdot \bmod q)$が可換モノイド(結合律が成立し，単位元が存在する)でかつ非零の逆元が存在し，さらに$(F_q; + \bmod q, \cdot \bmod q)$に対して左右の分配律が成立する．減算は，$(F_q; + \bmod q)$上の逆元の加算，除算は$(F_q; \cdot \bmod q)$上の逆元の乗算とすることができ，四則演算が可能である．

の経路が取りうる状態のパターンは，{○，●}から取り出す要素に重複を許す順列(重複順列)で表すことができる．取り出す要素は n 個である．この場合の重複順列の総数は，$_2\Pi_n=2^n$である．例えば k=2,n=4 の場合，図 26 のような状態が存在する．

	経路数		状態パターン			
	#{ℝ|$R_{1\le\le n}$が危険}	#{ℝ|$R_{1\le\le n}$が安全}	R_1	R_2	R_3	R_4
(1)	0	4	○	○	○	○
(2)	1	3	●	○	○	○
(3)			○	●	○	○
(4)			○	○	●	○
(5)			○	○	○	●
(6)	2	2	●	●	○	○
(7)			●	○	●	○
(8)			●	○	○	●
(9)			○	●	●	○
(10)			○	●	○	●
(11)			○	○	●	●
(12)	3	1	●	●	●	○
(13)			●	●	○	●
(14)			●	○	●	●
(15)			○	●	●	●
(16)	4	0	●	●	●	●

復元される

図 26 k=2,n=4 ((2,4)閾値法)の場合の重複順列

n 本の経路のうち危険である経路数を#{ℝ|$R_{1\le\le n}$が危険}とする．(k,n)閾値法を用いる場合，攻撃者によって秘密情報を復元される確率$P(D(A))$は，#{ℝ|$R_{1\le\le n}$が危険}≥ kに対して，

$$P(D(A)) = \sum \prod_{i=1}^{n} \begin{cases} P(R_i), & R_i\text{が安全} \\ \overline{P(R_i)}, & R_i\text{が危険} \end{cases}$$

で与えられる．

(1,4)，(2,4)，(3,4)，(4,4)，(1,1)閾値法を用いる場合，グリッド/ショートにおける攻撃ノード数(A)による$P(D(A))$の変化は，図 27(N=100，$(H_{a1},H_{a2},H_{a3},H_{a4})$ = (3,3,4,5)，ERS オン)のようになる．図 27 からわかるように(2,4)閾値法では，攻撃ノードがネットワーク上に 20%存在すると約 65%の確率で情報が復元される．(3,4)閾値法では，約 30%前後と低いがネットワークエラーには弱くなる(1 パケットのロスしか許容しない)．ネットワークエラーと攻撃耐性はトレードオフの関係にある．攻撃ノードがネットワーク上に 5～10%以内であれば，(2,4)閾値法は(1,1)閾値法より若干，攻撃耐性がある．必要経路数 n が増大すると，トポロジにも依存するが経路構築成功確率が低下する．また，復元に必要なパケット数 k が増加すると，ネットワークエラーに弱くなる．最適な n,k は，トポロジとネットワークエラーレートに依存すると考えられる．

5.2.2. ブラックホールノード存在時の宛先のデータ復元確率

分割情報から共通鍵の復元が不可能な場合(暗号化された共通鍵に対して閾値法を適用した場合)は，攻撃者は通信の妨害を行う可能性が高く，ブラックホールノードへの対処は重要な課題である．ブラックホールノードは，自身が経路に組み込まれた際にデータパケットの大多数，あるいは全てを意図的に破棄する．(k,n)閾値法においてブラックホールノードが存在すると，分割情報が宛先へ k 個配送することが困難になる．本稿では，より効率的に攻撃を行うためにブラックホールノードは，自身に届く分割情報の全てを破棄すると定義する．ネットワーク上にブラックホールノードが存在した場合の宛先がデータを復元できる確率$P(D(D))$は，n 本の経路のうち安全である経路数を#{ℝ|$R_{1\le\le n}$が安全}とすると，#{ℝ|$R_{1\le\le n}$が安全}≥ kに対して，

$$P(D(D)) = \sum \prod_{i=1}^{n} \begin{cases} P(R_i), & R_i\text{が安全} \\ \overline{P(R_i)}, & R_i\text{が危険} \end{cases}$$

で与えられる．

(1,4)，(2,4)，(3,4)，(4,4)，(1,1)閾値法を用いる場合，グリッド/ショートにおけるブラックホールノード数(A)による$P(D(D))$の変化は，図 27(N=100，$(H_{a1},H_{a2},H_{a3},H_{a4})$ = (3,3,4,5)，ERS オン)のようになる．

$P(D(A))$とは逆に，(1,4)閾値法がブラックホールノードの存在に対して強い耐性をもち，ネットワーク上に 50%ブラックホールノードが存在した場合でも，宛先は約 50%の確率で復元可能である．(2,4)閾値法でも，ブラックホールノードが 30%程度であれば(1,1)閾値法(H_a = 3,4)より復元率が高い．一方，(3,4)，(4,4)閾値法は，攻撃ノードへの情報漏洩には強いが，ブラックホールノードの存在やネットワークエラーによるパケットロストに弱い．ブラックホールノードへの対処と攻撃者への情報の秘匿はトレードオフの関係にある．

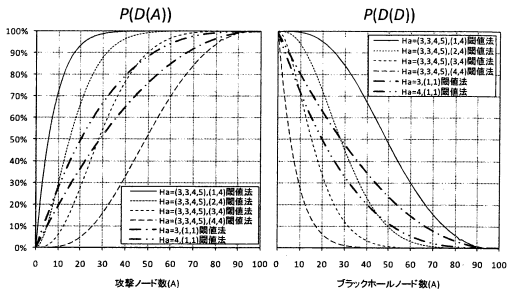

図 27 N=100，H_a =(3,3,4,5)，ERS オンにおける$P(D(A))$, $P(D(D))$

参考文献

[1] 森郁海，森拓海，小野良司，撫中達司，高橋修，" アドホックネットワーク(VANET，MANET)におけるセキュリティの定性評価とセキュアルーティングプロトコルの現状と課題"， DICOMO2008 2008.

[2] SBI Net Systems Co.,Ltd., "暗号入門", http://dev.sbins.co.jp/cryptography/cryptography02.html,2003

[3] S. Corson, J. Macker., "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations (RFC2501)", Jan 1999.

[4] 宇根正志，神田雅透，"暗号アルゴリズムにおける 2010 年問題について"，pp45，2006.

[5] Shamir, A., "How to share a secret", Comm. Assoc. Comput. Mach., vol.22, no.11, pp.612-613, Nov. 1979.

[6] ITpro, "ハイブリッド暗号方式", http://itpro.nikkeibp.co.jp/article/COLUMN/20060620/241303/, 2006.

[7] 上野裕介，撫中達司，小野良司，渡辺尚，"ルートの独立性を考慮したマルチパスルーチングプロトコルの提案とその評価"，情報処理学会論文誌 Vol45 No12, Dec. 2004.

[8] IETF Mobile Ad Hoc Networking Working Group, "Secure Ad hoc On-Demand Distance Vector (SAODV) Routing" , Internet Draft, 2006

[9] Ad hoc On-Demand Distance Vector (AODV) Routing(RFC3561) ,http://www.ietf.org/rfc/rfc3561.txt.

[10] W. Diffie and M. E. Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol.IT-22, No.6, pp.644-654, Nov, 1976.

[11] N.Koblits, "Elliptic curve cryptosystems", Mathematics of Computation 48, pp203-209, 1987.

[12] V.Miller, "Use of elliptic curves in cryptography", Advances in Cryptology-CRYPTO'85, Lecture Notes in Computer Science, 210, 1986, Springer-Verlag, pp.417-426.

[13] Alex Biryukov and Eyal Kushilevitz, "From Differential Cryptanalysis to Ciphertext-Only Attacks", CRYPTO 1998, pp72–88.

[14] AODV-UU, http://core.it.uu.se/adhoc/AodvUUImpl..

[15] A-SAODV (Adaptive SAODV), http://saodv.cefriel.it/.

[16] Network Simulator version 2(ns-2),http://www. isi .edu/nsnam/ns.

[17] libgcrypt, http://www.gnupg.org/.

[18] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," Wireless Communications and Mobile Computing, vol. 2, no. 5, pp. 483–502, 2002.

[19] M. K. Marina, S. R. DAS, "On-demand Multipath Distance Vector Routing in Ad Hoc Networks", IEEE ICNP 2001.

[20] "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4 (RFC 4728)", http://www.ietf.org/rfc/rfc4728.txt.

[21] 森拓海，森 郁海，高橋 修， "AAAr: Anti Attack Ad-hoc routing protocol の提案と実装・評価", DICOMO2007, 2007.

[22] G. R. Blakley and C. Meadows, "Security of ramp schemes," Advances in Cryptology-CRYPTO'84, LNCS 196, Springer-Verlag, pp. 242–269, 1985.

[23] A. Barbir , S. Murphy , Y. Yang , "Generic Threats to Routing Protocols (RFC 4593)", http://www.ietf.org/rfc/rfc4593.txt, pp.8, October 2006.