修士論文

アドホックネットワークにおける (k,n) 閾値法を用いた
認証方式に関する研究

An Authorization Method with (k,n) Threshold Scheme
in Mobile Ad Hoc Networks

2005 年 1 月 31 日提出

指導教員 中山 雅哉 助教授

新領域創成科学研究科
基盤情報学専攻

47-36334 山口 健輔

目 次

概要

近年、コンピュータの高性能化に伴い、ノートＰＣや携帯電話などの携帯型移動端末による通信が普及してきている。それによって、いつでもどこでも携帯型移動端末を用いて他の端末との通信を行いたいという要求が高まっている。現在、こうした携帯型移動端末での通信には、無線ＬＡＮが広く用いられるようになってきている。無線ＬＡＮでは端末がケーブルに束縛されることがなく、電波が届く範囲内でなら自由に移動することができる。このため、無線ＬＡＮは携帯型移動端末での通信に適した通信方法だといえる。ところが、無線ＬＡＮに接続して通信を行うためにはアクセスポイントに接続する必要があり、アクセスポイントが存在しない場所では無線ＬＡＮによるネットワークを構成することはできない。

そこで、携帯型移動端末によってより柔軟にネットワークを構築する手法として、アドホックネットワークが注目されている。アドホックネットワークとは、個々の移動型携帯端末にルータのようなパケット中継機能を持たせることによって、アクセスポイントやルータを介さずに移動型携帯端末だけで構成するネットワークである。特定のインフラを必要としないため、いつでもどこでも複数の端末でネットワークを構成することができる。アドホックネットワークでは、ネットワークを構成する端末が移動することによってネットワークのトポロジが頻繁に変化する。このため、こうした変化に強い特殊なプロトコルの実装が不可欠であり、現在様々なプロトコルが提案されている。ただし、アドホックネットワークではしばしば見知らぬ他人同士が緊密なネットワークを構成することになるので、パケットの改ざんやなりすましを防ぐ、もしくは検出することができるような、強固なセキュリティ機構も必要である。

現在、このようなセキュリティを確立する方法として、一般的に公開鍵暗号が広く用いられている。公開鍵暗号を用いるときにはなりましを防ぐために送信先ノードに正しい公開鍵を入手させる必要があるが、アドホックネットワークのように様々

なノードが出入りするネットワークにおいて公開鍵暗号を用いる際には、オンラインで公開鍵の正しさを証明する必要がある。そのための方法として、CA(Certificate Authority / 認証局) に公開鍵の証明書を発行してもらい、その証明書を提示する方法が広く用いられている。ところが、ネットワークの外部に CA の存在を仮定してしまうと、外部との通信ができない環境においても携帯端末同士でネットワークを構成することができるというアドホックネットワークの利点を損なうことになる。よって、アドホックネットワークにおいてはネットワーク内部で公開鍵の正しさを保証する機構が望まれることになる。また、アドホックネットワークではノードの移動や故障によってリンクの切断やつなぎ替えが頻繁に起こると特定のノードと通信できない状況に陥りやすいという特徴がある。単一のノードに重要な役割を任せるべきではない。このため、単純にネットワークの内部に CA の役割を果たす中心的なノードを配置するのは好ましい方法とはいえない。

そこで本論文では、複数のリーダーノードが (k,n) 閾値法を用いて証明書発行用の秘密鍵のデータを分割して保持し、協力して公開鍵の証明書を発行する手法を提案し、その有効性を示す。さらに、セキュリティ、利便性の両面から、適切な閾値の検討を行う。

また、現在数多く提案されているアドホックネットワークのプロトコルの中でも最も普及すると見られてる DSR プロトコルでは、ネットワーク内の経路情報の改ざん、偽造を防ぐ機構を備えていないため、悪意のあるノードが簡単に経路情報を操作することができてしまう。そこで本論文では、パケットを中継していくノードが公開鍵の証明書を用いて経路情報を含むパケットに署名することにより、経路情報の改ざん、偽造を検知する手法を提案し、その有効性を示す。

1　はじめに

1.1　本研究の背景と目的

近年、コンピュータの高性能化に伴い、ノートＰＣや携帯電話などの携帯型移動端末による通信が普及してきている。それによって、いつでもどこでも携帯型移動端末を用いて他の端末との通信を行いたいという要求が高まっている。現在、こうした携帯型移動端末での通信には、無線ＬＡＮが広く用いられるようになってきている。無線ＬＡＮでは端末がケーブルに束縛されることがなく、電波が届く範囲内でなら自由に移動することができる。このため、無線ＬＡＮは携帯型移動端末での通信に適した通信方法だといえる。ところが、無線ＬＡＮに接続して通信を行うためにはアクセスポイントに接続する必要があり、アクセスポイントが存在しない場所では無線ＬＡＮによるネットワークを構成することはできない。そこで、携帯型移動端末によってより柔軟にネットワークを構築する手法として、アドホックネットワークが注目されている。アドホックネットワークとは、個々の移動型携帯端末にルータのようなパケット中継機能を持たせることによって、アクセスポイントやルータを介さずに移動型携帯端末だけで構成するネットワークである（図 1)。

特定のインフラを必要としないため、いつでもどこでも複数の端末でネットワークを構成することができる。アドホックネットワークでは、ネットワークを構成する端末が移動することによってネットワークのトポロジが頻繁に変化する。このため、こうした変化に強い特殊なプロトコルの実装が不可欠であり、現在様々なプロトコルが提案されている。ただ、しばしば見知らぬ他人同士が緊密なネットワークを構成することになるので、パケットの改ざんやなりすましを防ぐ、もしくは検出することができるような、強固なセキュリティ機構も必要である。

現在、このようなセキュリティを確立する方法として、一般的に公開鍵暗号が広

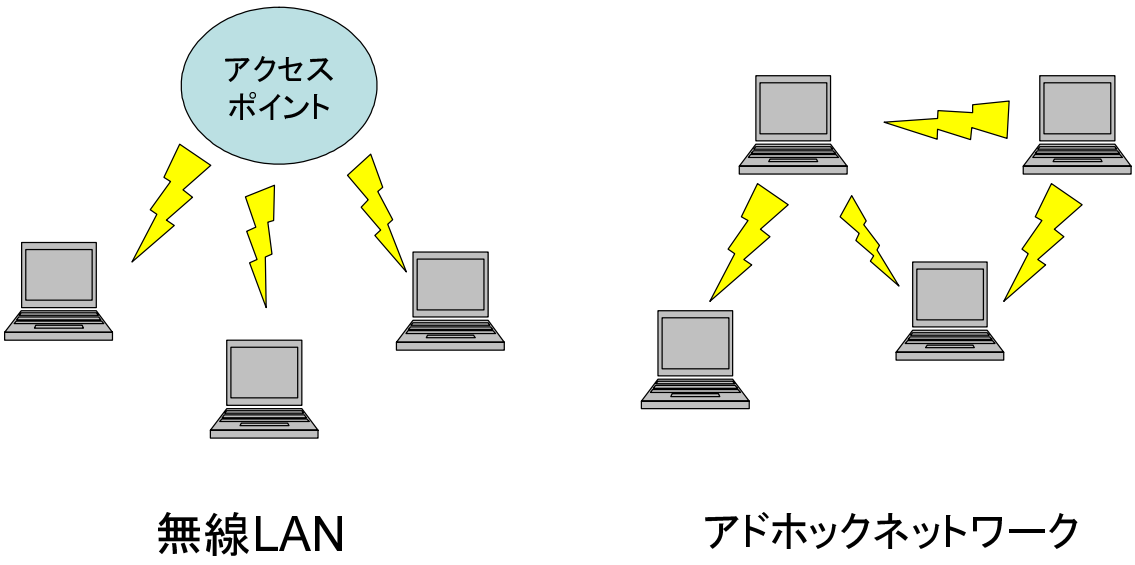

図 1: 無線 LAN とアドホックネットワーク

く用いられている。公開鍵暗号を用いるときにはなりましを防ぐために送信先ノードに正しい公開鍵を入手させる必要があるが、アドホックネットワークのように様々なノードが出入りするネットワークにおいて公開鍵暗号を用いる際には、オンラインで公開鍵の正しさを証明する必要がある。そのための方法として、CA(Certificate Authority / 認証局) に公開鍵の証明書を発行してもらい、その証明書を提示する方法が広く用いられている。ところが、ネットワークの外部に CA の存在を仮定してしまうと、外部との通信ができない環境においても携帯端末同士でネットワークを構成することができるというアドホックネットワークの利点を損なうことになる。よって、アドホックネットワークにおいてはネットワーク内部で公開鍵の正しさを保証する機構が望まれることになる。また、アドホックネットワークではノードの移動や故障によってリンクの切断やつなぎ替えが頻繁に起こると特定のノードと通信できない状況に陥りやすいという特徴がある。単一のノードに重要な役割を任せるべきではない。このため、単純にネットワークの内部に CA の役割を果たす中心的なノードを配置するのは好ましい方法とはいえない。

そこで本論文では、複数のリーダーノードが (k,n) 閾値法を用いて証明書発行用の秘密鍵のデータを分割して保持し、協力して公開鍵の証明書を発行する手法を提案し、その有効性を示す。さらに、セキュリティ、利便性の両面から、適切な閾値の検討を行う。

また、現在数多く提案されているアドホックネットワークのプロトコルの中でも最も普及すると見られてる DSR プロトコルでは、ネットワーク内の経路情報の改ざん、偽造を防ぐ機構を備えていないため、悪意のあるノードが簡単に経路情報を操作することができてしまう。そこで本論文では、パケットを中継していくノードが公開鍵の証明書を用いて経路情報を含むパケットに署名することにより、経路情報の改ざん、偽造を検知する手法を提案し、その有効性を示す。

1.2 本論文の構成

以下本論文では、第 2 章でアドホックネットワークの概略及びアドホックネットワークの代表的なルーティングプロトコルである DSR の概略を述べる。

第 3 章ではアドホックネットワークのセキュリティ上の問題点について述べ、具体的な攻撃例を示し、それらの攻撃への既存のネットワークにおける対策として公開鍵暗号方式及び CA について述べた。また、公開鍵暗号方式及び CA をアドホックネットワークに適用する際の問題点を述べる。

第 4 章では、関連研究として public key dividing model、及びそこで用いられている関連技術について述べる。

第 5 章では、アドホックネットワークにおいて複数のリーダーノードが (k,n) 閾値法を用いて証明書発行用の秘密鍵のデータを分割して保持し、協力して公開鍵の証明書を発行する手法および、パケットを中継していくノードが公開鍵の証明書を用いて経路情報を含むパケットに署名することにより、経路情報の改ざん、偽造を検知する手法を提案し、理論的評価を行って有効性を示す。

第６章では、提案手法を用いる際の適切な閾値について、セキュリティ、利便性の両面から理論値計算を行って検討した結果を示す。

2　アドホックネットワーク

本章では、近年新しいネットワークの構築方法として注目されているアドホックネットワークについて述べる。

2.1　アドホックネットワークの概要

アドホックネットワークとは、個々の移動型携帯端末にルータのようなパケット中継機能を持たせることによって移動型携帯端末だけで構成するネットワークである。各端末はアクセスポイントやルータを介さずに、ネットワークを構成する端末を経由してパケットの送受信を行う。このため、距離が遠いために直接通信できない端末とも近傍の端末を経由することによって通信することができる。

アドホックネットワークは以下のような特長を持つ。

- ネットワークの構築が容易

  特別なインフラを必要としないため、端末のネットワークへの参加、離脱が簡単に行える。

- インフラを使用できない環境下でのデータ伝送が可能

  インフラの整備がなされていない地域や、災害時などでインフラが使用できなくなった場合においてもネットワークを構築することができる。

- 端末の送信電力を抑えることができる

  個々の端末がネットワーク全域への通信能力を持つ必要がないため、端末の送信電力を低減し、端末のバッテリーの消費量を抑えることができる。また、端末の送信電力を小さくすることによって、電波の送出による周囲への影響を小さくすることができる。

2.2 ルーティング

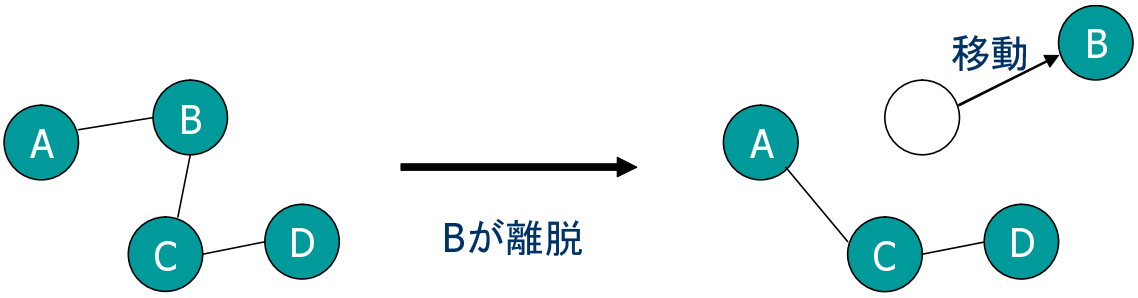

図 2: 端末の移動によるトポロジの変化

アドホックネットワークでは、ネットワークを構成する端末が移動することによってネットワークのトポロジが頻繁に変化する (図 2)。また、個々の端末の無線伝送距離には限界があるため、あまり遠くの端末とは直接通信することはできない。このため、こうした特徴に対応したルーティングプロトコルの実装が不可欠であり、現在 DSR[1, 5]、AODV[2, 6]、OLSR[3]、TBRPF[4] といった様々なプロトコルが提案されている。本節では、この中で最も普及すると見られている DSR[1, 5] の概要を述べる。

DSR は現在 Linux や Windows CE などで実装されており、パケットの転送にはソースルーティングというパケットの送信元があらかじめ転送経路を指定する方式を用いている。DSR では経路情報を得るたびにそれをキャッシュ (Route Cache) として蓄えており、転送の際にはそれを参照することによって頻繁に通信を行うノードに対して速やかにパケットの送信ができるようになっている。なお、通信を行うノード間のホップ数は数ホップの範囲を想定しており、数十ホップという長い経路は考えていない。

- DSR ヘッダ

DSR では、通常の IP パケットに DSR ヘッダを追加する。追加する位置は、IP ヘッダとそれに続く TCP、UDP といったトランスポート層のヘッダの間である。

DSR ヘッダにはオプション領域があり、そこには主に Route Request、Route Reply、Route Error、Acknowledgement Request、Acknowledgement、DSR Source Route といった種類のオプションが追加される。DSR を用いる際には経路探索や経路維持のためにいろいろな種類のパケットが飛び交うが、その種類はこれらのオプションで区別される。図 3 に DSR のパケットフォーマットを示す。

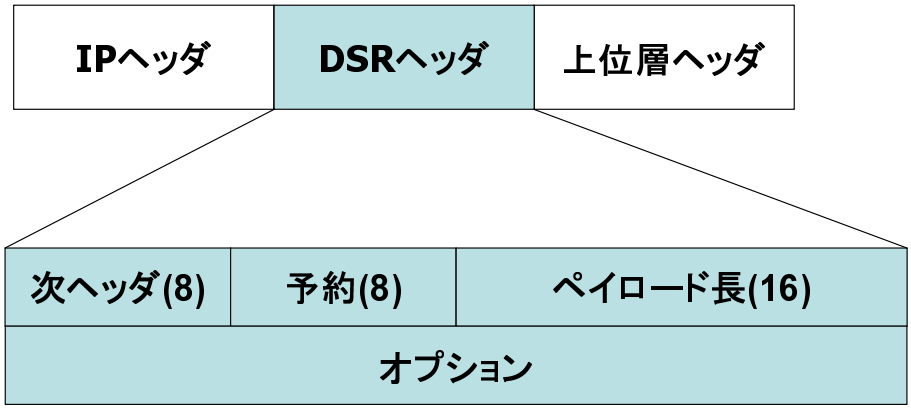

図 3: DSR のパケットフォーマット

- Route Discovery

  次に、DSR が実際にどのように動作するかを説明する。DSR は通信の要求があると現在利用可能な経路を探索する。これを Route Discovery と呼ぶ。

  以下で Route Discovery が行われる様子を図で示す。

  例では、開始ノード S と近傍のノード A,B,C,D,E の合計 6 つのノードが存在すると仮定する。ノード S はノード E とデータの通信をしようとしているものとする。最初にノード S は Route Request パケット（Route Request オプ

ションのついた DSR パケット、以下 RREQ）を周りのノードに送信する (図 4)。このとき、自分のノードをパケットに経路情報として追加する。

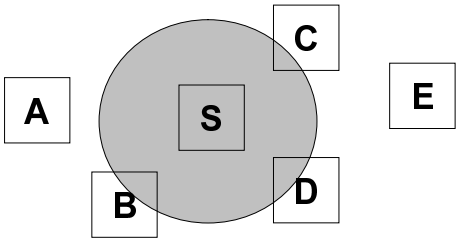

図 4: ノード S による Route Discovery の開始

すると、この RREQ パケットは電波を受信できる範囲にあるノード B,C,D で受信される。これらのノードは受信したパケットの宛先が自分でないことを確認すると、自分のノード ID をパケットの経路情報に追加して再び周りのノードに対してパケットを送信する (図 5)。

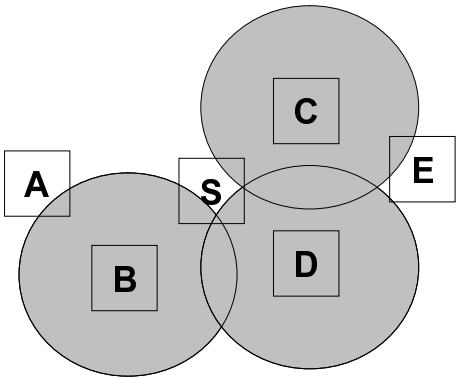

図 5: ノード B,C,D による Route Request の転送

すると、このパケットはノード A,S,E で受信されるが、このとき受信したパケットの経路情報に自分が含まれていたらそのパケットは破棄する。すなわち、ノード S はこのパケットを破棄することになる。ノード A は先ほどと同様に経路情報に自分のノード ID を追加して送信するが、ノード E はパケットの宛先が自分であることを確認すると、Route Reply パケット（以下 RREP）

を送信元ノードSに向けて返信する (図6)。ノードSはこのパケットを見てSからEへの経路情報を知ることになる。なお、各ノードは今までに受信した

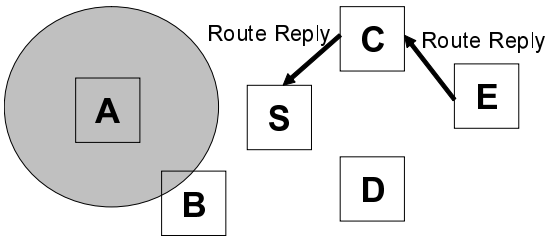

図 6: ノードEによるRoute Replyの転送

経路情報をRoute Cacheと呼ばれるキャッシュに蓄えており、パケットを送信する際にRoute Cacheを参照する。そしてキャッシュに経路情報が含まれていたらその経路情報に従ってデータを送信し、含まれていなかったらRREQを送信する。

- Route Maintenance

  Route Maintenanceとは、通信中に経路が何らかの原因で切断されてしまったときに、その経路を回避して通信を続けるための機構である。図7にその仕組みを示す。

  最初にノードAはA→B→C→Dという経路でDに対してパケットを送信しているとする (図7-(A))。

  この通信中にノードCが移動するなり電源を落とすなりしてB→Cのリンクが使えなくなると、それを知ったBはキャッシュからこのリンクを使う経路情報を削除し、Route Errorパケットを Aに返す (図7-(B))。

  これを受信したAは今までの経路が使えなくなったことを知り、自身のキャッシュから対応する経路情報を削除する。そして再びキャッシュからDへの経路を探し出し、なければRoute Discoveryを実行して新しい経路でDへのデータ

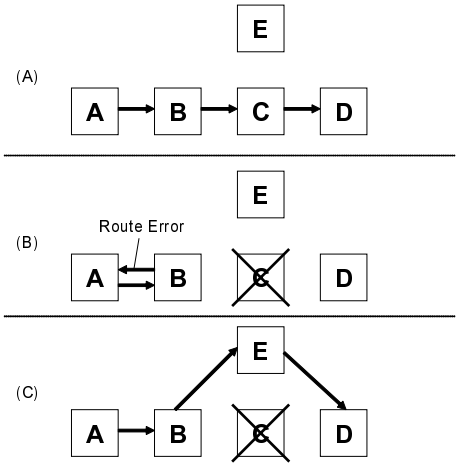

図 7: Route Maintenance の流れ

送信を再開する。なお、この間に失われたパケットについては TCP などの上位層プロトコルによって再送処理が行われる。

3　アドホックネットワークのセキュリティ

本章では、アドホックネットワークのセキュリティについて述べる。アドホックネットワークは、複数の携帯端末による一時的なネットワークであり、個々のノードは移動・消失しやすいといった特徴がある。このようなネットワークでは、セキュリティについて単一のリーダー的存在に頼るのは難しい。また、認証サーバなどのインフラを仮定してしまうと、いつでもどこでも即時にネットワークを構築できるというアドホックネットワークの利点を損なってしまう。アドホックネットワークのセキュリティについて考える場合には、これらの点に注意する必要がある。

3.1　アドホックネットワークに対する攻撃

アドホックネットワークに対する攻撃は、ネットワークの外部からの攻撃と内部からの攻撃に分類することができる。外部からの攻撃は主にパケットの盗聴、改ざん、消去を行ってネットワークを混雑させたりルーティングを混乱させたりするものである。外部からの攻撃は、ファイアウォールや暗号化技術によって比較的容易に防げることが多い。それに比べて、内部からの攻撃においては攻撃者はすでに認証を受けてネットワークの内部にいるので、攻撃者を特定しにくく、防ぎにくいことが多い。このため、内部からの攻撃に対して特に対策を練る必要がある。

ところが、MANET WG で提案されているプロトコルは、ネットワーク内のノードがすべて協調的に通信を行うことを仮定して設計されているため、こうした攻撃に対して何も対策が取られていない。次節では、アドホックネットワークにおける攻撃の例を挙げる。

3.2 アドホックネットワークに対する攻撃の例

本節では、アドホックネットワークに対する具体的な攻撃の例を挙げる。

- なりすまし

  なりすましとは、不正にネットワークに参加したり、ネットワークに参加してから権限を持ったユーザになりすましたりする攻撃である。これによって、内部の private な情報（ルーティング情報、位置情報など）を入手したり、内部のノードに攻撃を仕掛けやすくなるという効果がある。例えば、戦場で兵士達がネットワークを構成する場合には、敵にとって位置情報は重要な情報となるため、なりすましが特に脅威となる場合も考えられる。

- パケットの改ざん、偽情報の捏造

  DSR などのプロトコルは署名などの仕組みを備えていないため、攻撃者は容易に周囲のパケットを改ざんして流したり、偽の情報を持ったパケットを匿名で、もしくは偽名を使って流すことができる。また、アドホックネットワークではルーティングを行うための経路情報をノード間でのやり取りによって得るため、悪意のあるノードはネットワーク内を流れる経路情報を書き換えることによって、ネットワークを混乱させることもできる。

3.3 公開鍵暗号

パケットの改ざんやそれに伴うなりすましを防ぐ方法としては、一般的には公開鍵暗号方式 [10] が用いられる。公開鍵暗号方式とは、公開鍵、秘密鍵と呼ばれる 2 種類の鍵を用いてデータの暗号化及び復号を行う技術である。公開鍵を用いて暗号化したデータは、対になっている秘密鍵でしか復号できず、逆に秘密鍵で暗号化し

たデータは対になっている公開鍵でしか復号できない。個々のノードはそれぞれの公開鍵と秘密鍵の対を持っており、公開鍵だけをすべてのノードに公開する。

ノード A がノード B にパケットを送信する場合を考える。送信するパケットのデータの内容をノード B 以外に知られたくない場合、ノード A はデータをノード B の公開鍵で暗号化して送信する（図 8）。すると、そのデータは対になる秘密鍵を持っているノード B にしか復号できず、他のノードに内容を知られることはない。

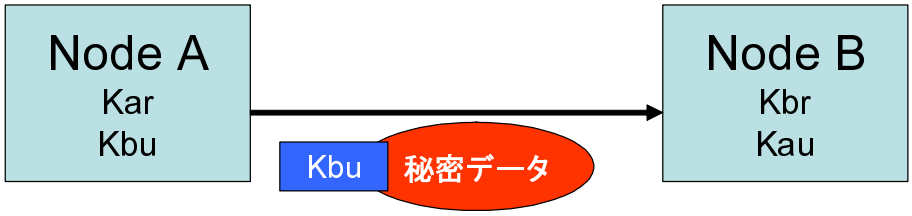

図 8: ノード A による秘密データの送信

また、データを作成したのがノード A であると証明したい場合、データをノード A の秘密鍵で暗号化して送信する（図 9）。ノード B がそのデータをノード A の公開鍵で復号できれば、そのデータは確かにノード A が送信したものであると証明される。これを、デジタル署名と呼ぶ。

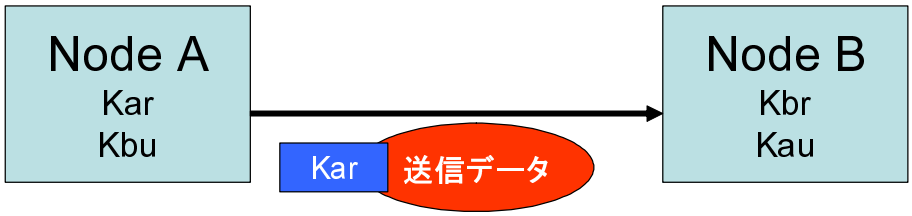

図 9: ノード A によるデジタル署名

公開鍵暗号方式では、通信相手に正しい公開鍵を入手させることが重要である。今、悪意のあるノード C が自らの公開鍵をノード A の公開鍵であると偽ってノード B に入手させた場合を考える。この場合ノード B は、ノード C が暗号化したデータをノード A から送られたデータであると認識してしまう。また、ノード A だけが復

号できるようにノードAの公開鍵で暗号化したつもりが、実際にはノードCに解読されてしまう。つまり、ノードCは自らの公開鍵をノードAの公開鍵であると偽って配布することにより、ノードAになりすますことができるのである。

3.4 CA(Cetrificate Authority)

公開鍵暗号方式で用いられる公開鍵を正しく入手するための手法として、一般にCA(Cetrificate Authority)[?] が広く利用されている。CAとは、電子的な身分証明書を発行し、管理する機関であり、証明書所有者の鍵ペア（秘密鍵と公開鍵）に対して公開鍵証明書を発行する。以下に、図10を用いてＣＡによる公開鍵証明書の発行とその使用手順を示す。

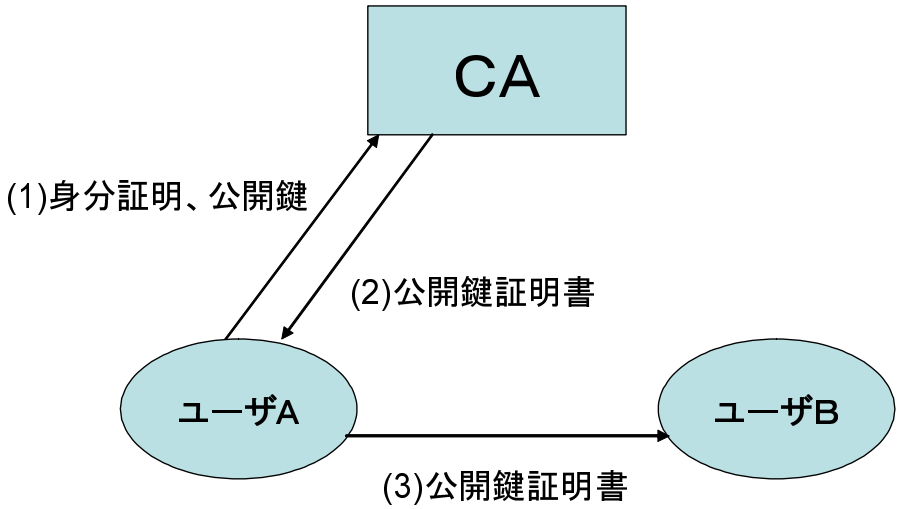

図 10: ＣＡによる公開鍵証明書の発行とその使用手順

まず、すべてのユーザはCAの秘密鍵に対応する公開鍵を持っているとする。最初にユーザAは自身の身分証明と公開鍵をCAに送り、証明書の発行申請をする（図10中(1)）。CAはユーザAからの発行申請を受理すると、自身の秘密鍵でユーザAの身分証明及び公開鍵を暗号化する。これが公開鍵証明書となる。CAは公開鍵証明

書を作成すると、これをユーザ A に送る（図 10 中 (2)）。こうしてユーザ A は公開鍵証明書を得る。次にユーザ B からユーザ A に対して通信要求があったとする。すると、ユーザ A は自身の公開鍵証明書をユーザ B に送る。ユーザ B は CA の公開鍵でこの証明書を復号することにより、ユーザ A の正しい公開鍵を得ることができる。

このシステムではセキュリティは CA の信頼性に依存するため、CA には高い堅牢性が要求され、CA では証明書の発行などに使用する署名生成システムにおいて、署名生成に使用する署名鍵およびデータに関し、その作成から廃棄までの全てのライフサイクルにわたり、十分な機密性を確保し、解読、漏えいなどによる証明書の偽造が行われないようにすることが必要である。

3.5 アドホックネットワークにおける CA の問題点

この CA をアドホックネットワークで適用することを考えると、いくつかの問題点が存在する。まず、アドホックネットワークは外部のネットワークと通信できない状況で構成されることも考えられるため、外部の CA の存在に頼ると公開鍵を更新した時などに証明書を更新できなくなる可能性があるので、アドホックネットワークでは外部に CA のようなネットワーク的に固定された存在を仮定するべきではない。また、アドホックネットワークではノードの移動や故障によってリンクの切断やつなぎ替えが頻繁に起こると特定のノードと通信できない状況に陥りやすいため、単一のノードに重要な役割を任せるべきではない。そこでネットワーク内に CA の役割を担うノードを複数配置する解決案が考えられるが、この手段では一つの CA が悪意のあるユーザに乗っ取られると、そのユーザに偽の証明書を作られてしまうため、脆弱性が増してしまう。そこで、一定数以上の CA に発行してもらった証明書だけをネットワーク内で使用できるような方式が必要となるが、そのためには証明書に一定数以上の CA によって発行されたという証明を付加する必要がある。そこで 5 章では、証明書を発行するための秘密鍵を (k, n) 閾値法 [8] を用いて複数のリー

ダーノードが分割して保持し、各ノードがネットワーク参加時に自らの証明書を発行してもらい、その証明書を他のノードとの通信時に通信相手に送ることによって自らの正しい公開鍵を入手させるという手法を提案する。

また、その証明書を用いてRoute Request、Route Replyパケットに署名することにより、経路情報の改ざん、偽装を検知する手法も同時に提案する。

4　関連研究

本論文で提案する手法では、証明書を発行するための秘密鍵を分割するのに [7] で提案されている private key dividing model を用いる。本章ではこの private key dividing model について述べる。

4.1　private key dividing model

private key dividing model は、ネットワーク内のノードが近隣の複数のノードに自らの信頼性を証明してもらってその証明書の分割データを受け取り、一定数以上のノードからの分割データが集まったら証明書を作成できるという手法である。正当な証明書を持たないノードはネットワーク内でのパケットのやり取りができないようになっている。

なお、以下では鍵をアルファベットを用いた略称で表記する（図 11）。略称の 1 文字目の K は Key の略である。2 文字目は鍵の所有者を表し、A ならばノード A、S ならばノード S、D ならばノード D の鍵であることを表す。3 文字目は秘密鍵であるか公開鍵であるかを表し、r ならば秘密鍵、u ならば公開鍵であることを表す。例えば、ノード A の秘密鍵は KAr、ノード D の公開鍵は KDu と表す。なお、2 文字目が小文字の c の場合、その鍵は certification key、すなわち認証用の鍵であることを意味するものとする。また、鍵 Kx を用いてデータ D を暗号化したデータを、Kx(D) と表記することにする。

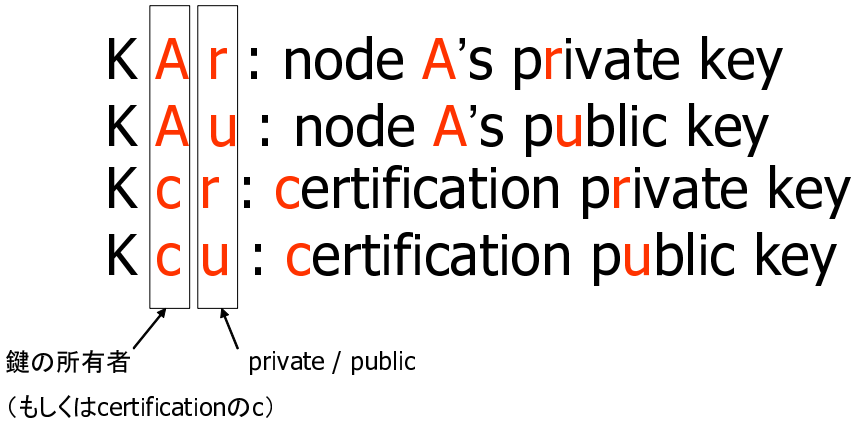

図 11: 本論文における鍵の表現

4.1.1 (k, n) 閾値法

(k, n) 閾値法 [8] とは、あるデータを n 個に分割し (図 12)、その分割データのうちの任意の k 個を集めると元のデータを復元できる (図 13) という手法である。またこのとき、どの k-1 個からも復元することはできない。この (k, n) 閾値法を用いると、秘密データを複数のノードで分割して管理し、そのうちの一定数以上のノードの承認によって元の秘密データを復元できる、という管理が可能となる。

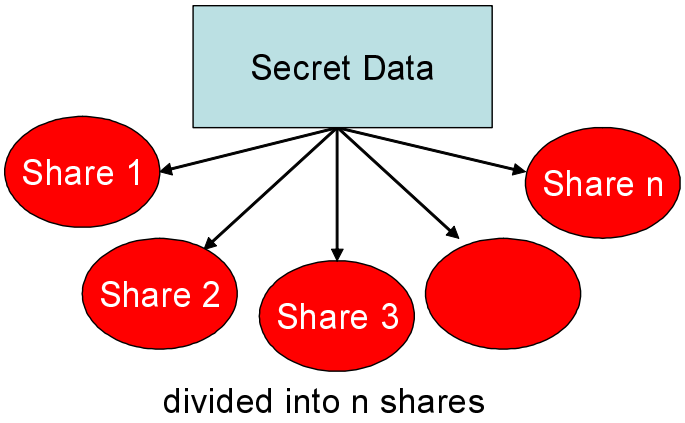

図 12: (k, n) 閾値法による秘密データの分割

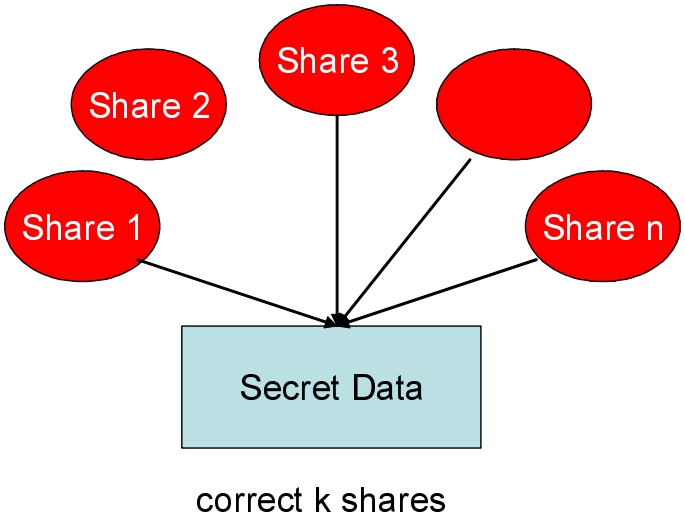

図 13: (k, n) 閾値法による秘密データの復元

4.1.2 ラグランジュ補間

次に、ラグランジュ補間 (Lagrange interpolation) の原理を説明する。今、

$$l_j(x) = \prod_{k=1,k\neq j}^{n} \frac{x - x_k}{x_j - x_k}$$

なる関数 $l_j(x)$ を定義すると、$x = x_j$ ならば $l_j(x) = 1$、$x \neq x_j$ ならば $l_j(x) = 0$ となる。よって、n 個の 2 次元座標 $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$ が与えられたときにそれらすべての点を通るような関数 $y = f(x)$ は、この $l_j(x)$ を用いて

$$f(x) = \sum_{j=1}^{n} y_j \cdot l_j(x)$$

と表すことができる。この公式をラグランジュの補間公式と呼ぶ。

4.2 証明書の発行方法

前提条件として、ネットワークに参加するノードはあらかじめ正しいKcuおよび、Kcrを(k, n)閾値法で分割した分割データを持っているものとする。

まず、ネットワークを構成する段階で、$Kcr = (d, N)$, $Kcu = (e, N)$の組を生成する。そして、K-1次の多項式$F(x) = d + f_1 \cdot x + f_2 \cdot x^2 + \cdots + f_{K-1} \cdot x^{K-1}$を決定する。$f_1, f_2, \cdots, f_{K-1}$はランダムに選ばれた有限な自然数とする。ここで、$F(0) = d$である。そして各ノード$N_i(i = 1, 2, \cdots, n)$に、$P_i = F(i) mod N$および$Kcu = (e, N)$をオフラインで配布する。なお、各ノードは自らのIDとしてiを知っているものとする。

ここで、ラグランジュの補間公式を用いると

$$d = F(0) = \sum_{j=1}^{K} P_j \cdot l_j(0) mod N$$

となり、$Kcr_i = P_i \cdot l_i(0) mod N$とすると、

$$d = F(0) = \sum_{j=1}^{K} Kcr_j (mod N)$$

と表せる。各ノードN_iはP_iからKcr_iを計算することができる。

ここでは、ノードAが証明書の発行を要求する際の例を示す。なお、分割データを復元するための閾値はkであるとする。まず、ノードAは発行要求のパケットをブロードキャストする。それを次々にブロードキャストしていき、JREQは最終的にリーダーノードに届けられる。このパケットを受け取った近隣ノードN_iは、証明書の分割データ$CERT_i = (cert)^{Kcr_i}$を生成し(certは証明書の平文)、ノードAに送る。ノードAは証明書の分割データがK個集まったら、$\prod_{j=1}^{K} CERT_j mod N = (cert)^{\sum_{j=1}^{K} Kcr_j mod N} = (cert)^{t \cdot N + d}$を計算する(tは0以上の整数)。ノードAは$Kcu = (e, N)$を持っているので、$(Add_A, KAu)^{t \cdot N + d}$に$(Add_A, KAu)^{-N}$をt回乗じることによって証明書$(cert)^d$を得ることができる。なお、$(Add_A, KAu)^{d \cdot e} = (Add_A, KAu)$となった時点でt回乗

じたことが分かる。なお、この証明書を Kcu で復号できれば、その証明書は正しいものであると保証される。

5　提案手法

本章では、前章で述べた private key dividing model を用いて、アドホックネットワークにおいて複数のリーダーノードが協力して公開鍵証明書を発行する手法および、その公開鍵証明書を用いた署名によって経路情報の改ざん、偽造を検知する手法を提案する。

5.1　公開鍵証明書の発行

前提条件として、ネットワークに参加するノードはあらかじめ正しい Kcu および全てのリーダーノードの公開鍵を持っているものとする。まず、ネットワークを構成する前にメンバー認証用の秘密鍵 Kcr を (k, n) 閾値法で分割し、それぞれの分割データをリーダーノードが持つ (図 14)。

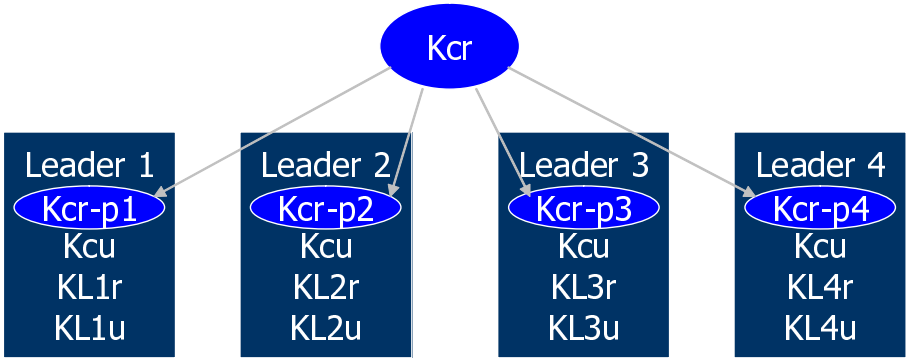

図 14: Kcr の分割

この場合、n はリーダーノードの数であり、n 個のリーダーノードのうち k 個分の分割データが集まれば Kcr が復元できることになる。

ここでは、ノード A がネットワークに参加する際の例を示す。なお、分割データ

を復元するための閾値を k とする。まず、ノード A は Join Request(JREQ) と呼ばれるパケットに自らの公開鍵 KAu を付加してブロードキャストする (図 15)。この JREQ を受け取ったノードはそれを次々にブロードキャストしていき、JREQ は最終的にリーダーノードに届けられる。これを受け取ったリーダーノードはノード A に対してアドレスを発行し、自身が持っている Kcr の分割データを用いてノード A の証明書の分割データを作成する。ここで、ノード A の証明書とはノード A のアドレス (addA) と KAu の組を Kcr で暗号化したもの（Kcr(addA, KAu)）とする。

そして、各リーダーノードは証明書の分割データをノード A の公開鍵で暗号化し、これをノード A に送る (図 16)。この分割データを k 個受け取ったら、ノード A は証明書を作成することができる。ノード A がこの証明書を Kcu で復号できれば、この証明書は正しい Kcr の分割データによって作成されたものであると確認できる。

5.1.1 本手法の理論的評価

本手法は公開鍵の証明書をネットワーク内部の複数のリーダーノードが発行するという点で、場所や環境を選ばずにいつでもネットワークを構成できるというアドホックネットワークの利点を損なわずに公開鍵の証明機構を実現していると言える。

また、アドホックネットワークではネットワークのトポロジが頻繁に変化し、リンクの切断や切り替えが頻繁に起こる事態も想定されるので単一の認証ノードに頼ることは望ましくないが、本手法では (k, n) 閾値法を用いて複数のリーダーノードに認証機能を分散することで、最大 k-1 個のリーダーノードが悪意のあるユーザに乗っ取られても偽の証明書を発行することはできず、またいくつかのリーダーノードがネットワークの障害やノードの故障などによって通信できない状態に陥っても、k 個のリーダーノードと通信できさえすれば証明書を発行してもらうことができる。

証明書は Kcr で暗号化されており、認証ノード以外には作ることができないので、あるノードの証明書を受け取ったノードは同時に送信ノードの本物の公開鍵を入手

することができる。よって、正しく Kcr で暗号化された証明書がパケットに付加されていて、その証明書に含まれている公開鍵でパケットを復号できれば、確かにそのパケットは証明書の中に記されているアドレスを持つノードが作成したことが証明できる。

なお、各ノードはあらかじめ全てのリーダーノードの公開鍵を持っているため、証明書発行の際にリーダーノードのなりすましや Join Request パケットの改ざんの恐れはない。

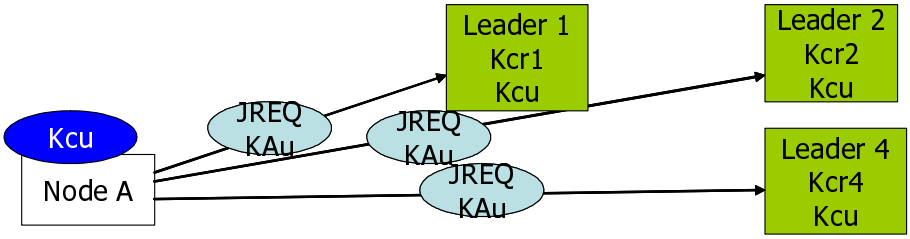

図 15: ノード A による Join Request のブロードキャスト

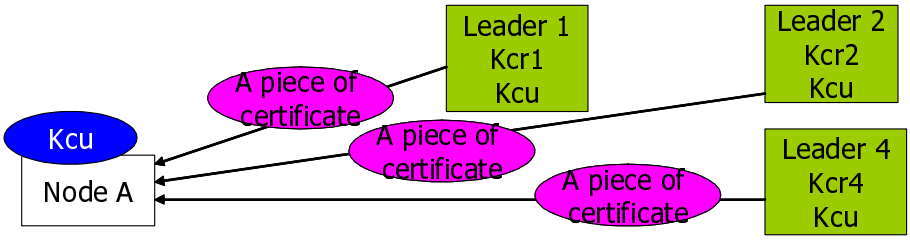

図 16: リーダーノードによる証明書の分割データの送信

5.2 パケットへの署名による経路情報の偽装の防止

本節では、前節の手法で発行した証明書を用いて Route Request、Route Reply パケットに署名することにより、経路情報の改ざん、偽装を検知する手法を提案する。

以下、ネットワーク内で共通のハッシュ関数 H を定義し、パケット P を H にかけたものを H(P) と表現する。本署名方式では、Route Request を送信するノードはパケットを送信する際にまず自らの証明書を付加する。そして証明書を付加したパケットをハッシュ関数にかけたものを自らの秘密鍵で暗号化したものを付加してブロードキャストする。それを受け取った中間ノードも同様の処理を行ってブロードキャストする。Route Request を受け取った目的ノードは、新しく付加された順に付加されている証明書を Kcu を用いて復号して中継ノードおよび送信元ノードのアドレスと証明書内のアドレスを比較し、さらに得られた公開鍵を使って H(P) を取り出して正しい H(P) と照合する。

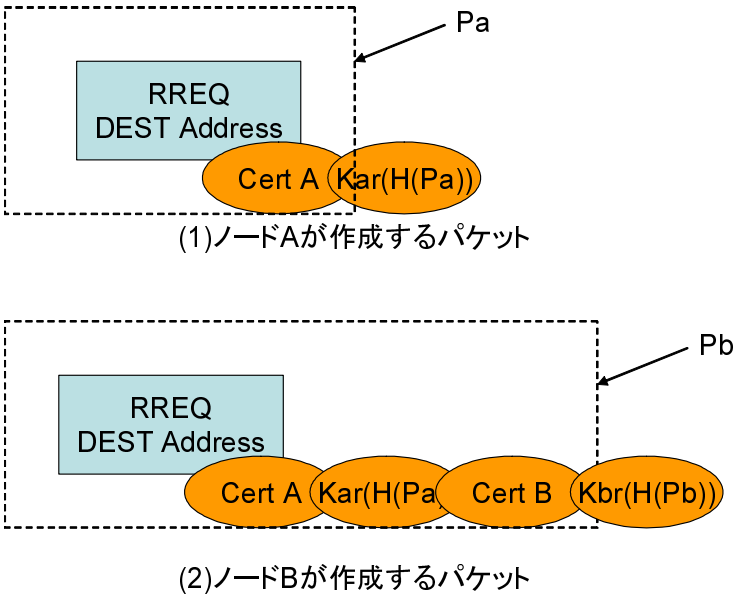

図 17: ノード A、ノード B の署名処理

例として、ノード A がノード D への経路を得る手順を示す。最初に、図 17(1) にノード A が Route Request パケットに署名する手順を示す。まずノード A は Route Request に自らの証明書を付加する。これを Pa とする。次に Pa をハッシュ関数にかけた H(Pa) を自らの秘密鍵で暗号化し、Route Request に付加する。そしてこれ

をブロードキャストする。次に、これを受け取った中継ノード B の処理を図 17(2) に示す。ノード B は、受け取ったパケットに自らの証明書を付加する。これを Pb とする。そして Pb をハッシュ関数にかけた H(Pb) を自らの秘密鍵で暗号化し、Route Request に付加し、さらにブロードキャストする。すべての中継ノードはこの処理を繰り返して Route Request をブロードキャストしていく。

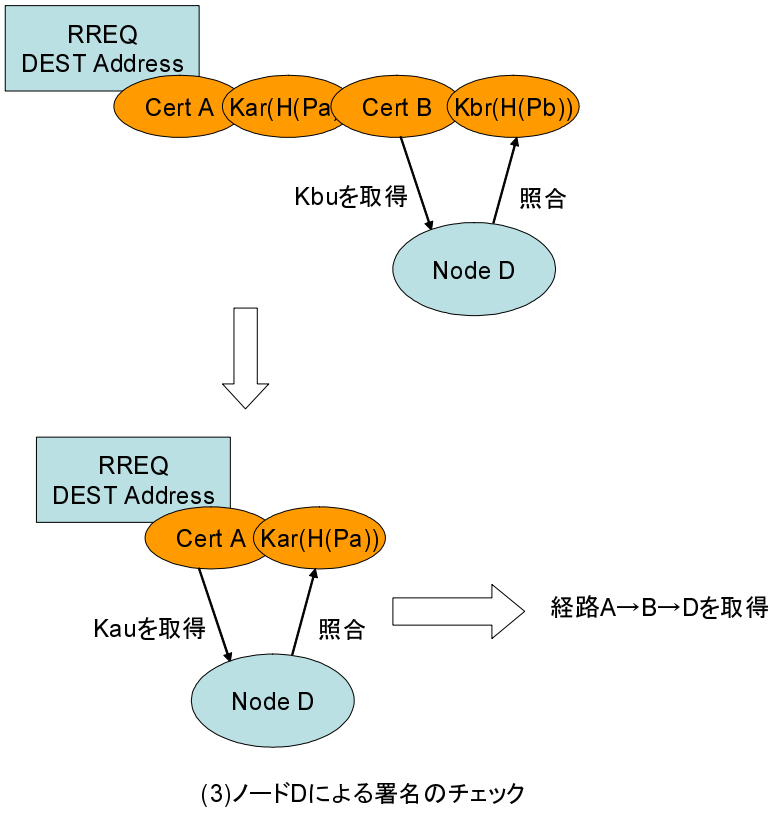

図 18: ノード D の署名チェック

次に、自分宛の RREQ を受け取ったノード D が署名をチェックする手順を図 18 に示す。ノード D は、新しく付加された順に付加されている証明書を Kcu を用いて復号して中継ノードおよび送信元ノードのアドレスと証明書内のアドレスを比較する。さらに得られた公開鍵を使って H(P) を取り出し、自ら H(P) を生成して照合する。すべて一致していればこのパケットが A、B、D の順に転送されてきたことが

分かるので、ノード A への経路を基に Route Reply を作成し、KDr で暗号化してからノード A へユニキャストで送信する。このようにして、ノード A はノード D への経路を得ることができる。

5.2.1 署名方式の理論的評価

本方式で署名する際にハッシュ関数を用いた処理を行っているのは、証明書を付加したのが確かにその証明書の持ち主であると確認できるようにするためである。ネットワーク内のノードは他のノードの証明書を入手することができるため、単純に証明書を付加しただけでは本当にその証明書を持ち主が付加したのかどうか確認できない。そこで、パケットを受け取ったノードが証明書を付加した後にハッシュ関数にかけてそれを自らの秘密鍵で暗号化することにより、最後に Route Request を受け取った目的ノードは証明書を付加したのがその証明書に含まれている公開鍵に対応する秘密鍵を持っているノードであることを確認することができる。

本方式によって、Route Request パケットを受け取った目的ノードはそのパケットが通ってきた経路を正しく知ることができ、正しい署名によって付加されなかった経路情報は破棄することができる。よって、開始ノードは悪意のあるノードによって偽装、改変された経路情報を入手してしまうことがなくなる。

6　適切な閾値の検討

提案手法において、閾値をどのように設定するかということは重要な検討課題である。閾値を大きくすると、悪意のあるユーザは偽の証明書を発行するためにより多くのリーダーノードを乗っ取る必要があるためセキュリティは強固になるが、証明書の発行を要求する際により多くのリーダーノードからの分割データを受け取る必要があるため、発行の待ち時間が長くなり、発行自体が失敗する可能性も高くなる。逆に閾値を小さくすると、証明書の発行待ち時間が短くなって発行の成功率も高くなるが、悪意のあるユーザに偽の証明書を発行される恐れも大きくなる。このため、本手法を用いる際にはセキュリティ、利便性の両面から適切な閾値を設定する必要がある。

そこで本章では、まずセキュリティ面を優先的に考慮してリーダーノード数に対する閾値の下限を設定し、その条件の下で利便性を考慮して適切な閾値を検討する。

6.1　セキュリティ面からの検討

本節では、まずセキュリティ面を考慮して適切な閾値を検討する。なお、本論文では、提案手法をあまり大規模なネットワークに適用することを想定していないため、本章ではリーダーノード数 10 以下の環境で検討する。

以下では各リーダーノードが悪意のあるユーザに攻撃を受けた際に 5%の確率で乗っ取られる状況を想定し、閾値数以上のリーダーノードが乗っ取られて悪意のあるユーザが偽の証明書を発行できてしまう状態に陥る確率を Pp とする。図 19〜図 26 に、それぞれ n = 3 〜 10 の時に閾値 k を変化させた時の Pp の理論値を示した。Pp は、

$$Pp = \sum_{i=k}^{n} ({}_nC_i \cdot 0.05^i \cdot 0.95^{n-i})$$

として計算した。k が小さい時は Pp は高い値を示しており、リーダーノード数が増加していくにしたがって Pp も増加していくが、k を大きくしていくと Pp は急激に下がっていく。このことから、単純に CA の機能を持ったリーダーノードを複数配置すると危険性が増していくことがわかり、(k, n) 閾値法を用いて複数のリーダーノードが協力して証明書を発行する本手法の有効性が示された。

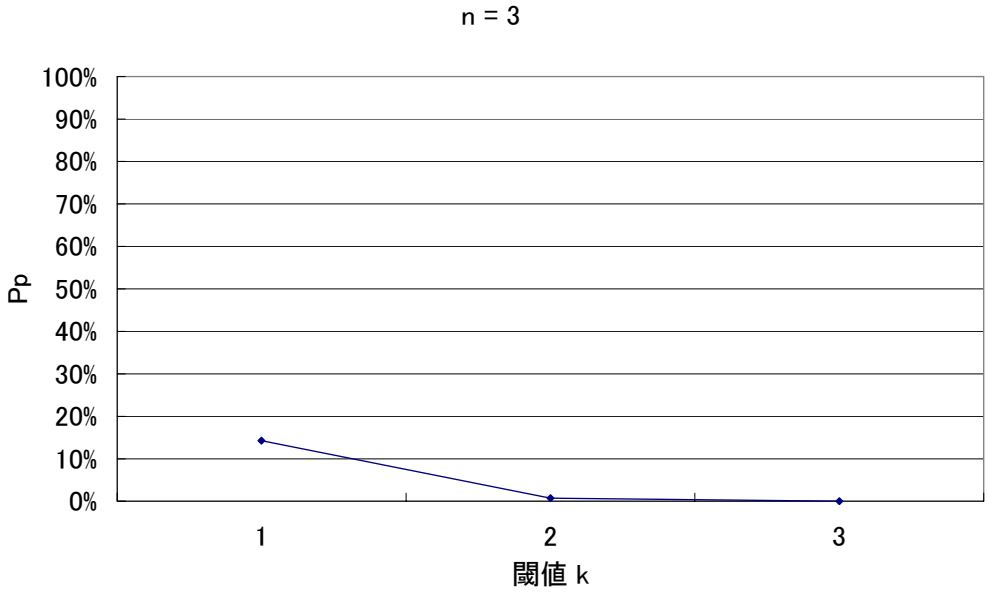

図 19: n = 3 の時の Pp

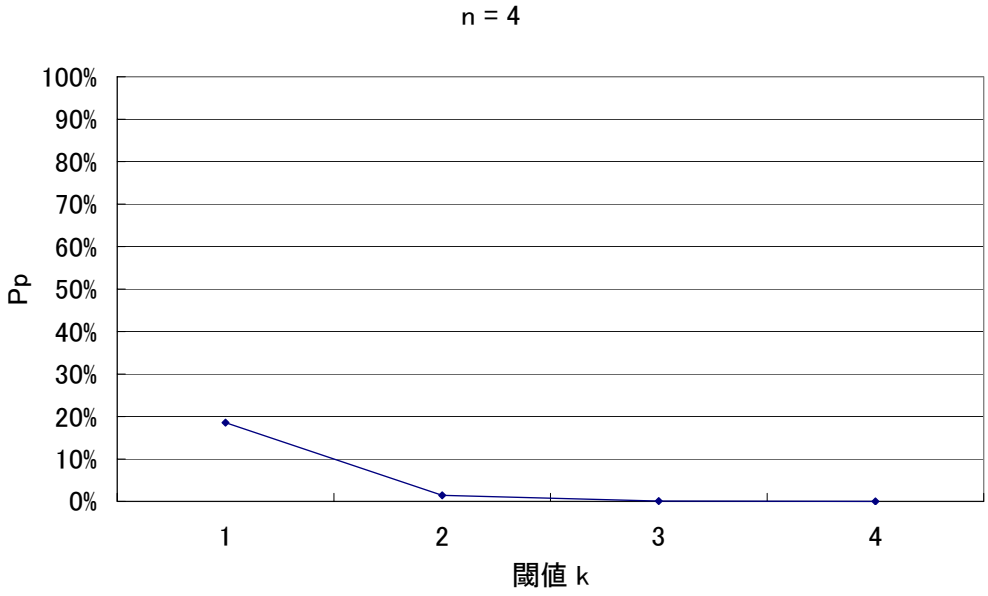

図 20: n = 4 の時の Pp

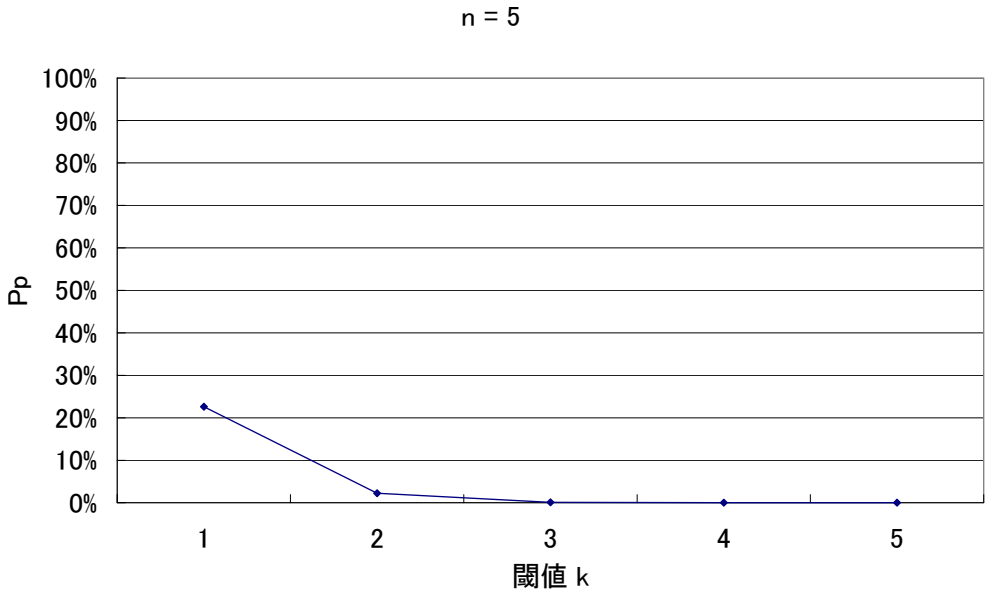

図 21: n = 5 の時の Pp

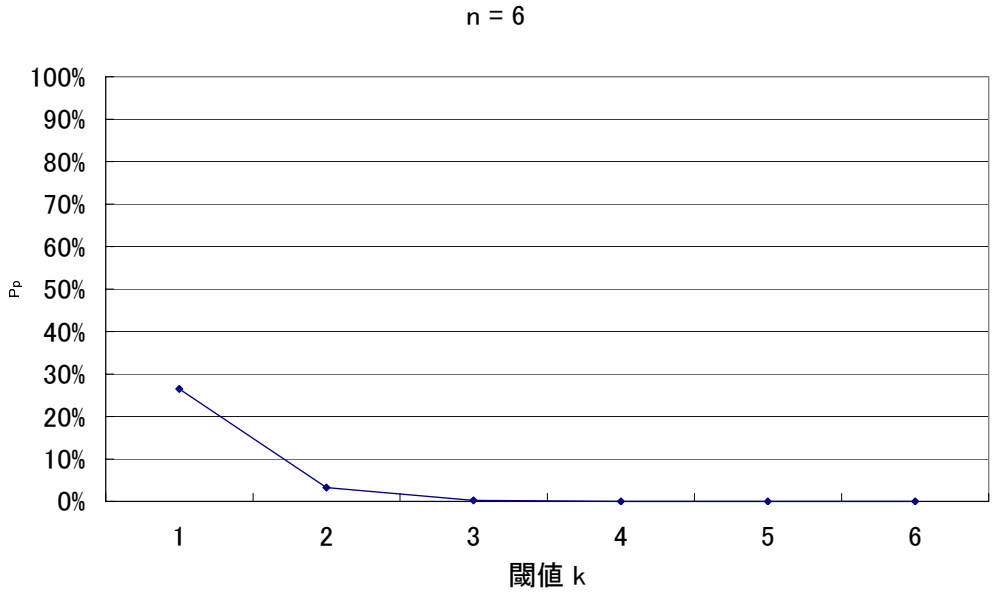

図 22: n = 6 の時の Pp

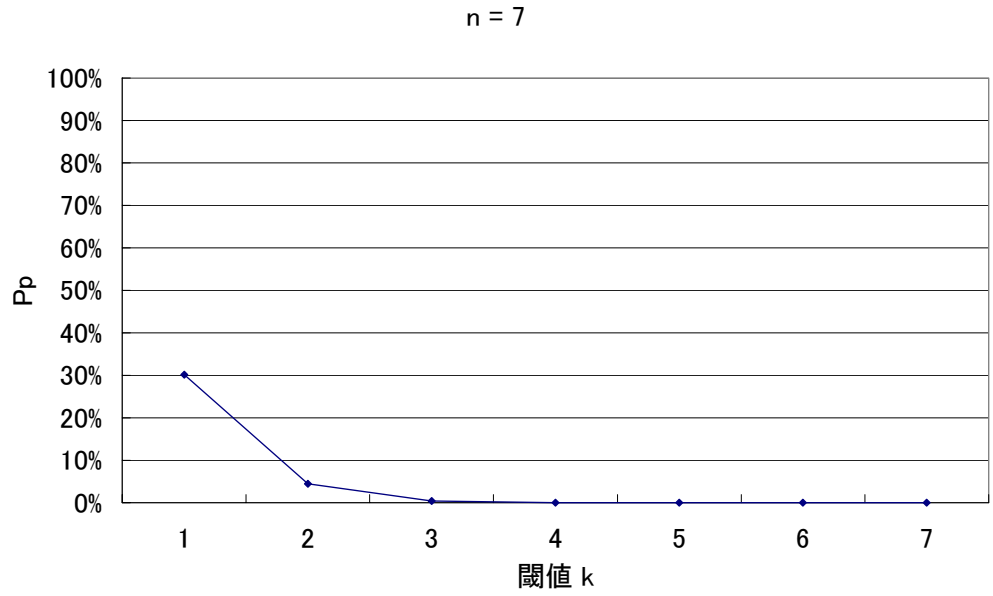

図 23: n = 7 の時の Pp

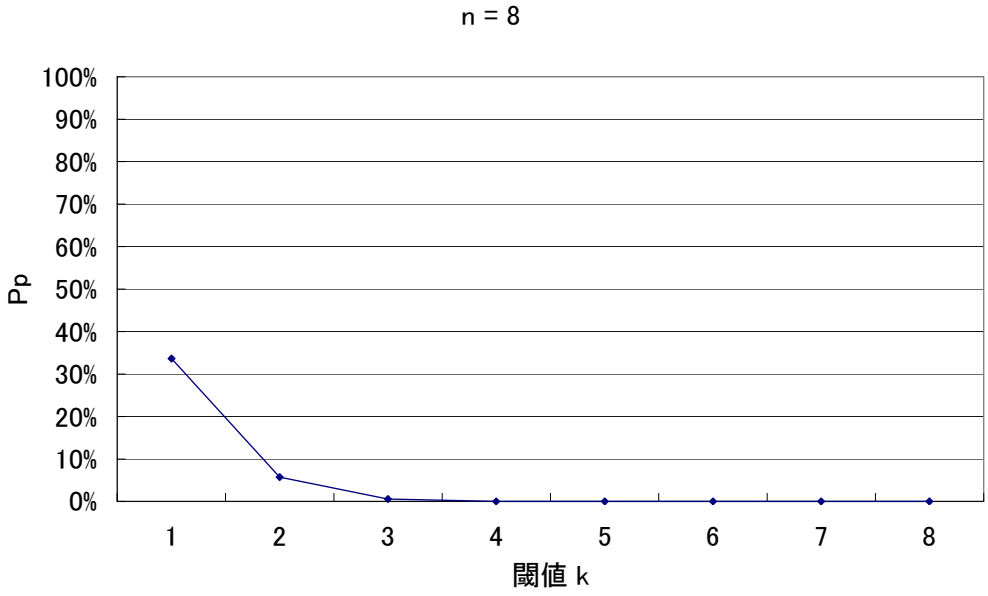

図 24: n = 8 の時の Pp

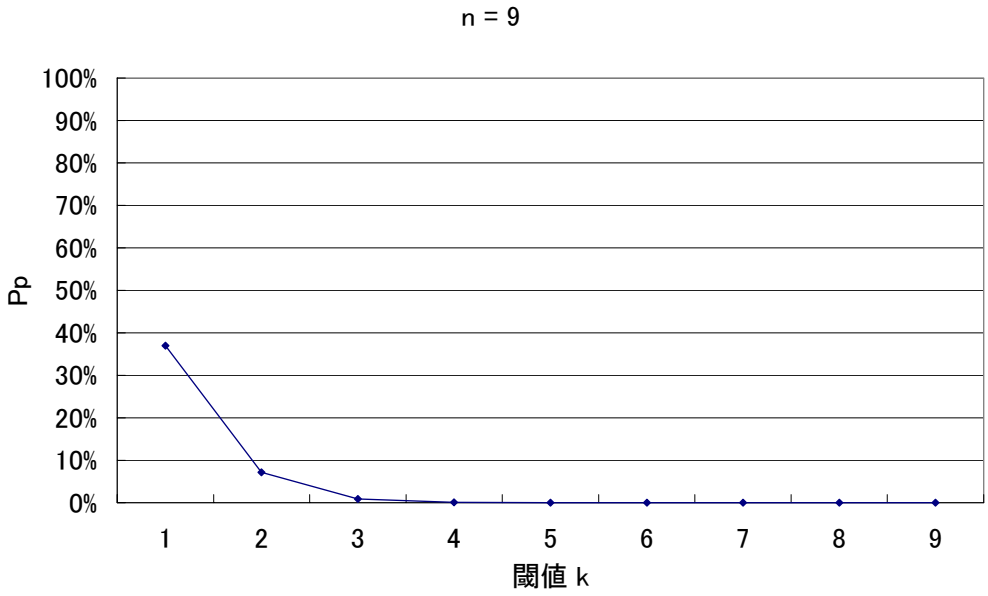

図 25: n = 9 の時の Pp

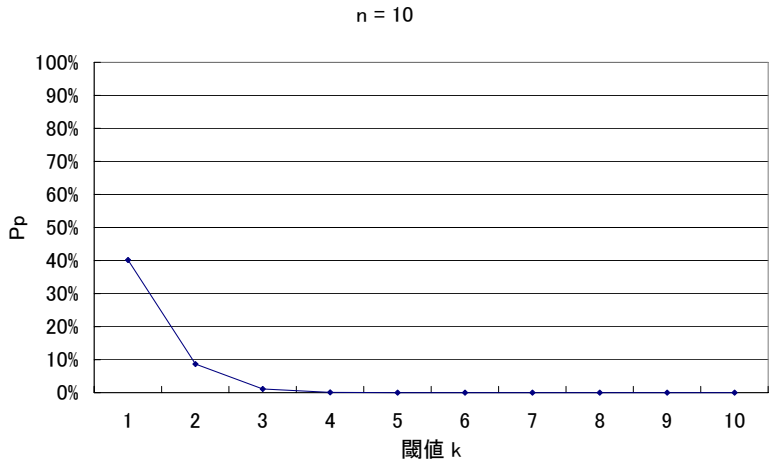

図 26: n = 10 の時の Pp

また、表 1 に、1 から 10 までのリーダーノード数 n に対して閾値 k を変化させた時の Pp の理論値を計算した結果を示した。

k＼n	1	2	3	4	5	6	7	8	9	10
1	5.000%	9.750%	14.263%	18.549%	22.622%	26.491%	30.166%	33.658%	36.975%	40.126%
2	–	0.250%	0.725%	1.402%	2.259%	3.277%	4.438%	5.725%	7.121%	8.614%
3	–	–	0.013%	0.048%	0.116%	0.223%	0.376%	0.579%	0.836%	1.150%
4	–	–	–	0.001%	0.003%	0.009%	0.019%	0.037%	0.064%	0.103%
5	–	–	–	–	0.000%	0.000%	0.001%	0.002%	0.003%	0.006%
6	–	–	–	–	–	0.000%	0.000%	0.000%	0.000%	0.000%
7	–	–	–	–	–	–	0.000%	0.000%	0.000%	0.000%
8	–	–	–	–	–	–	–	0.000%	0.000%	0.000%
9	–	–	–	–	–	–	–	–	0.000%	0.000%
10	–	–	–	–	–	–	–	–	–	0.000%

表 1: Pp の理論値

今 Pp の許容範囲を 5%未満と定めると、表中のグレーで示した部分が閾値としてふさわしくない範囲に当たることになる。

6.2 利便性からの検討

本節では、証明書取得の成功率という利便性の側面から適切な閾値を検討する。閾値を高くすればセキュリティは強固になるが、それだけ多くのリーダーノードと通信する必要が出てくるので、証明書発行までの待ち時間は長くなる。そればかりか、ネットワーク内でリンクエラーが起きる環境ではリーダーノードと通信できない状況が起こりうるので、あまり閾値を高く設定すると閾値数以上のリーダーノードと通信することができずに証明書の取得が失敗する可能性が高くなる。

アドホックネットワークの各リンクが一定確率でエラーを起こす状況で、各リーダーノードとの通信が成功する確率を Ps とする。各リンクがエラーを起こす確率を er、各リーダーノードへのホップ数を h とした時、Ps は、

$$Ps = \sum_{i=k}^{n} ({}_nC_i \cdot (1-er)^{h*i} \cdot (1-(1-er)^h)^{n-i})$$

として計算できる。図 27〜図 34 に、それぞれ k = 1 〜 8 の時のリーダーノード数に対する Ps の理論値を示した。閾値を上げていくと Ps が下がっていくが、リンクエラー率が高いときには特に急激に下がっていくことがわかる。

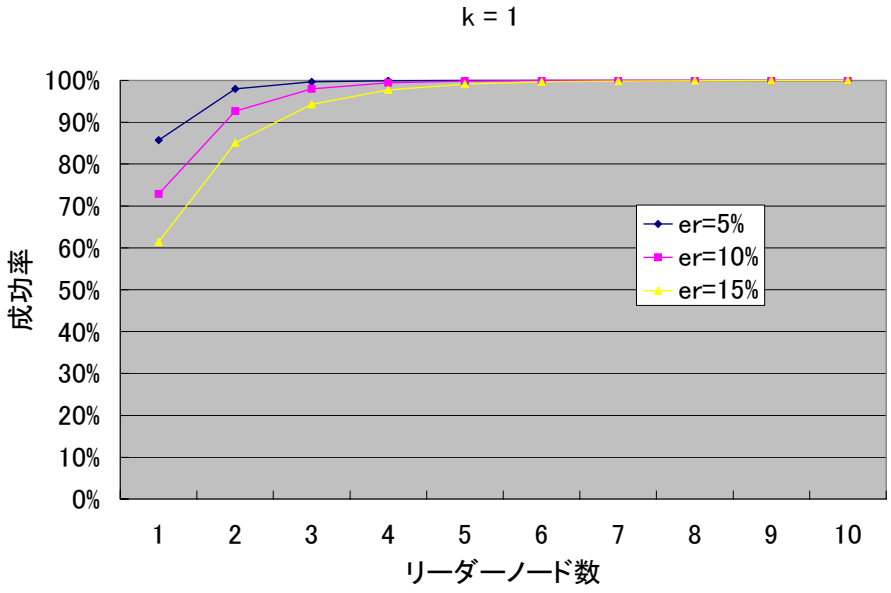

図 27: k = 1 の時の成功率

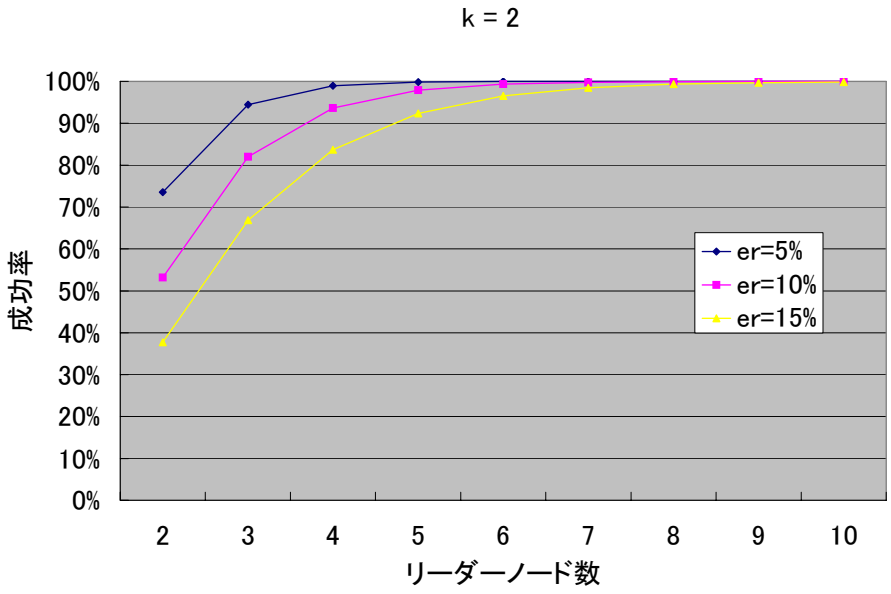

図 28: k = 2 の時の成功率

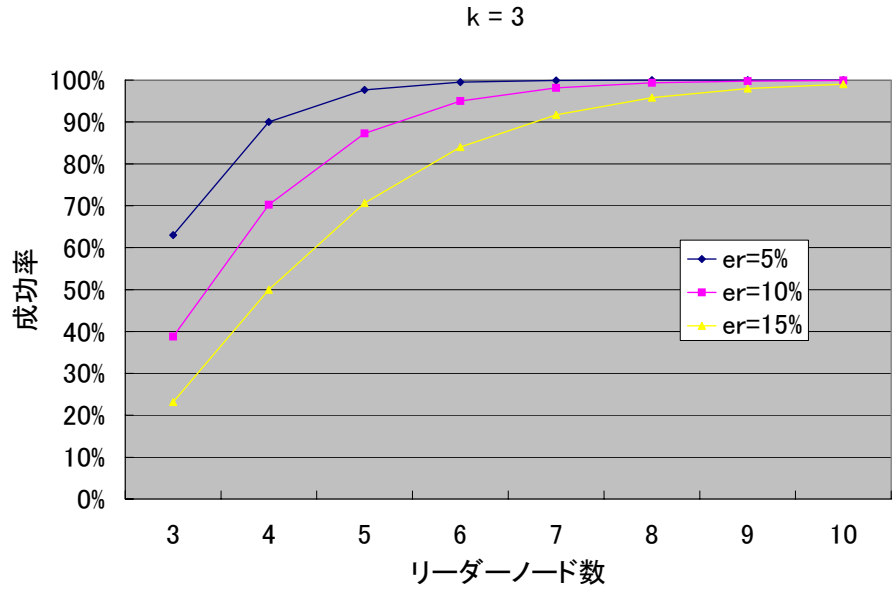

図 29: k = 3 の時の成功率

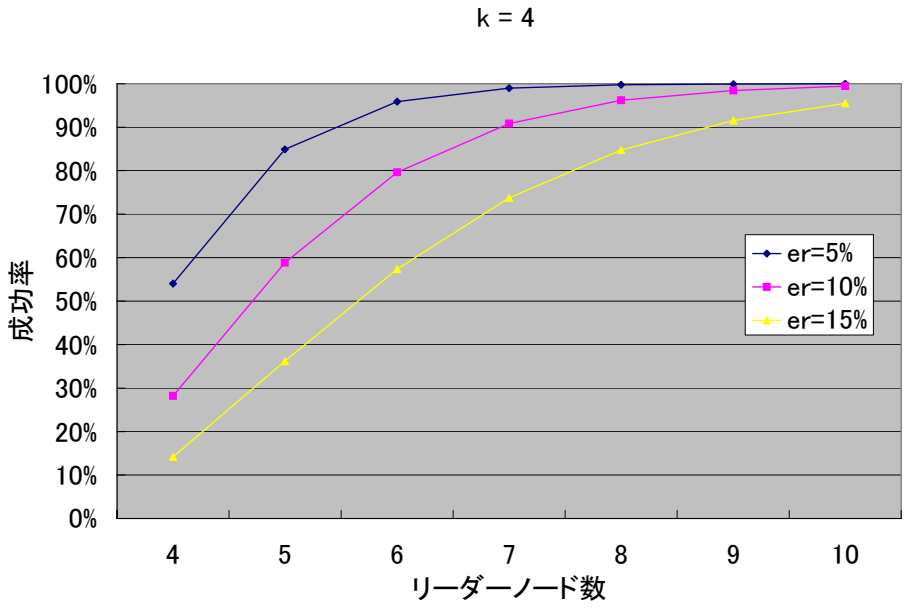

図 30: k = 4 の時の成功率

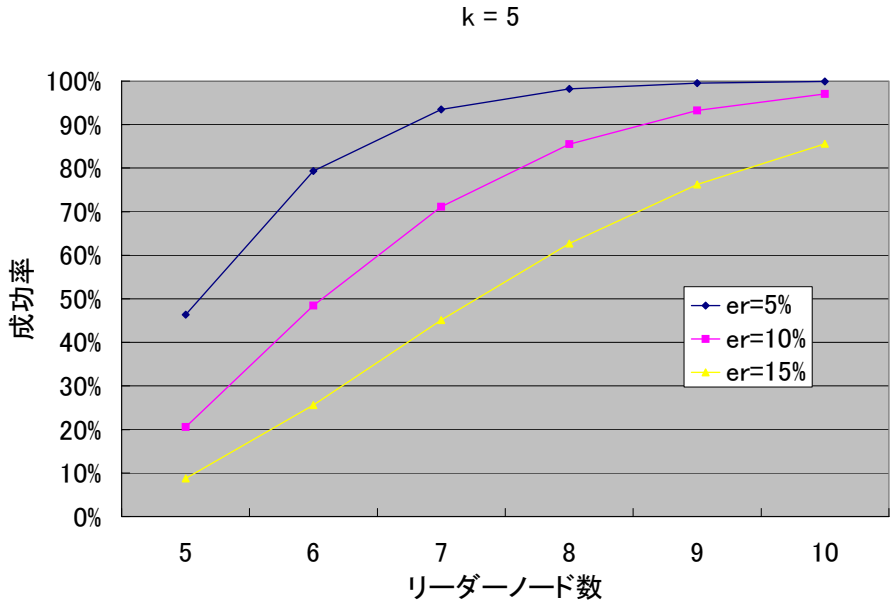

図 31: k = 5 の時の成功率

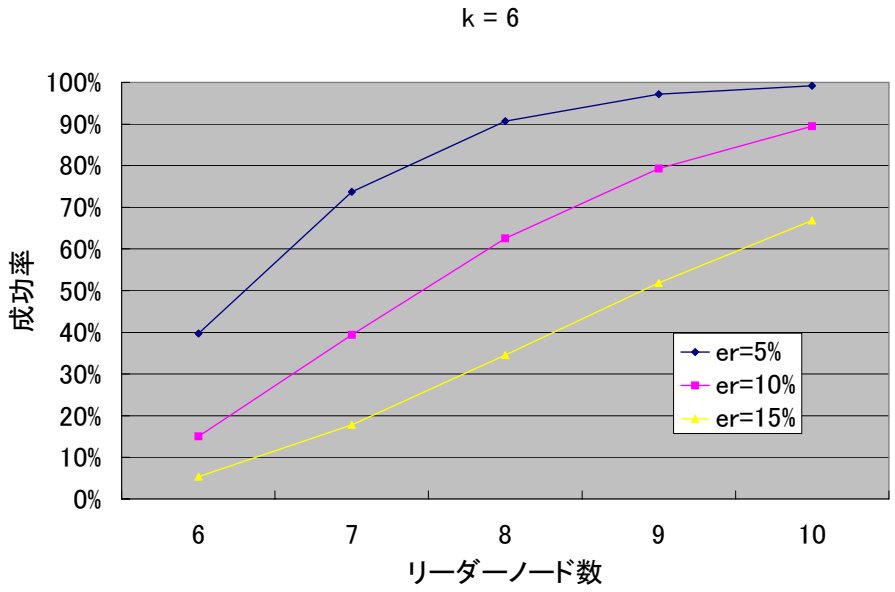

図 32: k = 6 の時の成功率

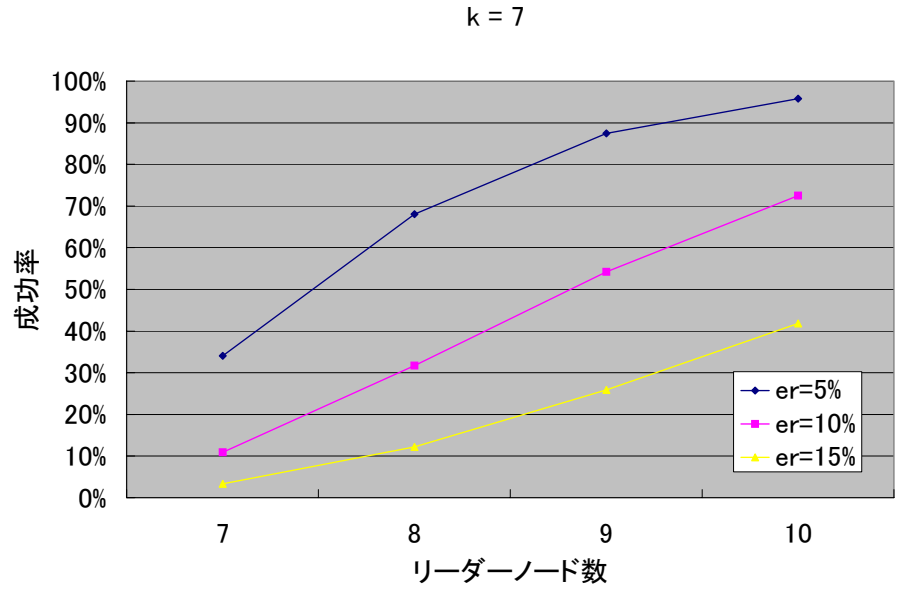

図 33: k = 7 の時の成功率

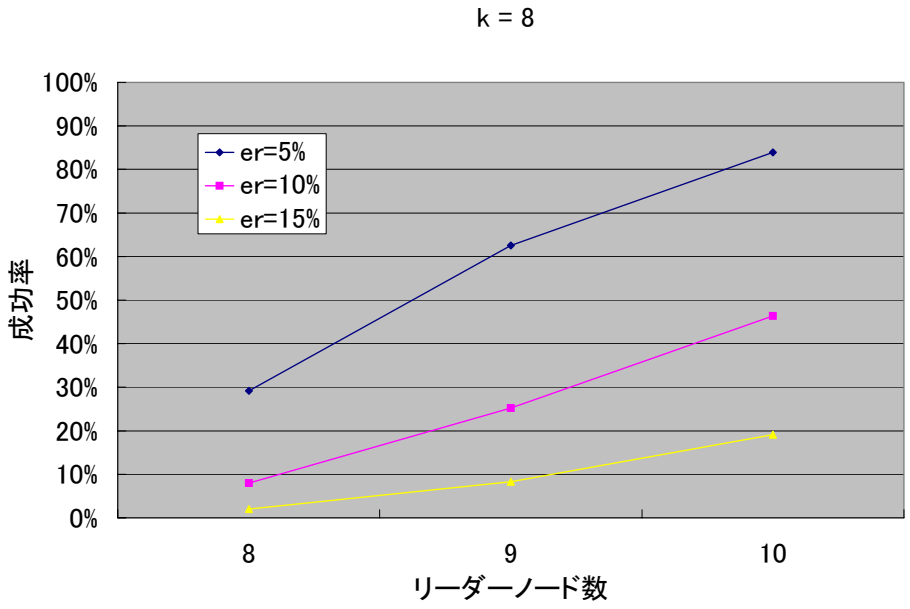

図 34: k = 8 の時の成功率

また、表2、3、4に、それぞれer=5%、10%、15%の時に1から10までのリーダーノード数nに対して閾値kを変化させた時のPsの理論値を計算した結果を示した。

成功率の許容範囲を90%と定め、前節で示したセキュリティ面での許容範囲と合わせると、表中のグレーで示した部分が閾値としてふさわしくない範囲に当たることになる。リンクエラー率が低い環境では、適切な閾値の値はリーダーノード数の30〜60%程度であることが分かる。リンクエラー率が15%と高い環境では証明書の取得成功率がかなり低い値になるが、それでもリーダーノード数の30%程度の値に設定すればセキュリティを保ったまま十分な成功率を達成することができるといえる。

k＼n	1	2	3	4	5	6	7	8	9	10
1	85.738%	97.966%	99.710%	99.959%	99.994%	99.999%	100.000%	100.000%	100.000%	100.000%
2	–	73.509%	94.478%	98.964%	99.817%	99.969%	99.995%	99.999%	100.000%	100.000%
3	–	–	63.025%	89.992%	97.684%	99.513%	99.904%	99.982%	99.997%	99.999%
4	–	–	–	54.036%	84.864%	95.856%	98.991%	99.774%	99.952%	99.990%
5	–	–	–	–	46.329%	79.368%	93.504%	98.208%	99.550%	99.895%
6	–	–	–	–	–	39.721%	73.713%	90.681%	97.135%	99.206%
7	–	–	–	–	–	–	34.056%	68.057%	87.455%	95.754%
8	–	–	–	–	–	–	–	29.199%	62.515%	83.897%
9	–	–	–	–	–	–	–	–	25.034%	57.169%
10	–	–	–	–	–	–	–	–	–	21.464%

表 2: Psの理論値 (er=5%)

k＼n	1	2	3	4	5	6	7	8	9	10
1	72.900%	92.656%	98.010%	99.461%	99.854%	99.960%	99.989%	99.997%	99.999%	100.000%
2	–	53.144%	81.948%	93.657%	97.888%	99.321%	99.787%	99.935%	99.980%	99.994%
3	–	–	38.742%	70.239%	87.311%	95.022%	98.156%	99.345%	99.775%	99.925%
4	–	–	–	28.243%	58.858%	79.600%	90.842%	96.174%	98.486%	99.425%
5	–	–	–	–	20.589%	48.487%	71.169%	85.511%	93.284%	97.076%
6	–	–	–	–	–	15.010%	39.415%	62.563%	79.292%	89.492%
7	–	–	–	–	–	–	10.942%	31.699%	54.199%	72.492%
8	–	–	–	–	–	–	–	7.977%	25.270%	46.359%
9	–	–	–	–	–	–	–	–	5.815%	19.998%
10	–	–	–	–	–	–	–	–	–	4.239%

表 3: Psの理論値 (er=10%)

k＼n	1	2	3	4	5	6	7	8	9	10
1	61.413%	85.110%	94.254%	97.783%	99.145%	99.670%	99.873%	99.951%	99.981%	99.993%
2	–	37.715%	66.822%	83.669%	92.337%	96.518%	98.453%	99.325%	99.709%	99.876%
3	–	–	23.162%	49.974%	70.667%	83.975%	91.678%	95.839%	97.980%	99.042%
4	–	–	–	14.224%	36.179%	57.359%	73.704%	84.742%	91.557%	95.501%
5	–	–	–	–	8.735%	25.589%	45.100%	62.667%	76.224%	85.640%
6	–	–	–	–	–	5.365%	17.785%	34.560%	51.821%	66.807%
7	–	–	–	–	–	–	3.295%	12.194%	25.929%	41.830%
8	–	–	–	–	–	–	–	2.023%	8.269%	19.115%
9	–	–	–	–	–	–	–	–	1.243%	5.558%
10	–	–	–	–	–	–	–	–	–	0.763%

表 4: Ps の理論値 (er=15%)

7　終わりに

本論文では、アドホックネットワークにおいて複数のリーダーノードが (k,n) 閾値法を用いて証明書発行用の秘密鍵のデータを分割して保持し、協力して公開鍵の証明書を発行する手法を提案した。また、パケットを中継していくノードが公開鍵の証明書を用いて経路情報を含むパケットに署名することにより、経路情報の改ざん、偽造を検知する手法を提案した。

第２章では、アドホックネットワークの概略及びアドホックネットワークの代表的なルーティングプロトコルである DSR の概略を述べた。

第３章では、アドホックネットワークのセキュリティ上の問題点について述べ、具体的な攻撃例を示し、それらの攻撃への既存のネットワークにおける対策として公開鍵暗号方式及び CA について述べた。また、公開鍵暗号方式及び CA をアドホックネットワークに適用する際の問題点を述べた。

第４章では、関連研究として public key dividing model、及びそこで用いられている関連技術について述べた。

第５章では、アドホックネットワークにおいて複数のリーダーノードが (k,n) 閾値

法を用いて証明書発行用の秘密鍵のデータを分割して保持し、協力して公開鍵の証明書を発行する手法および、パケットを中継していくノードが公開鍵の証明書を用いて経路情報を含むパケットに署名することにより、経路情報の改ざん、偽造を検知する手法を提案し、理論的評価を行って有効性を示した。

第６章では、提案手法を用いる際の適切な閾値について、セキュリティ、利便性の両面から理論値計算を行って検討した結果を示した。

謝辞

本研究を進めるに当たり、指導教員である中山雅哉助教授には研究の方針や内容について多大な御助言、御指導をいただき、非常にお世話になりました。また若原恭教授、中村文隆助手、関谷勇司助手にもミーティング等を通じて適切な御助言、御指導をいただき、非常にお世話になりました。心から感謝しております。

また、研究室の学生の皆様にも、ミーティングでの助言、ネットワークや計算機管理等の研究環境の構築、整備などさまざまな面でサポートしていただきました。大変感謝しております。

また、秘書の皆様方にも事務的な面で御尽力いただき、常に安心して研究を進めることができました。非常に感謝しております。

研究室の皆様の御指導、御尽力のおかげで無事に本論文を書き終えることができましたことを、心より感謝いたします。

参考文献

[1] David B. Johnson, David A. Maltz, Yih-Chun Hu. "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)". Internet Draft(http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt).(Apr. 2003)

[2] Charles E. parkins, Elizabeth M. Belding-Royer, Samir R.Das. "Ad hoc On-Demand Distance Vector (AODV) Routing". Internet Draft(http://vesuvio.ipv6.cselt.it/internet-drafts/draft-ietf-manet-aodv-13.txt).(Feb. 2003)

[3] Jacquet P, Muhlethaler P, Qayyum A. "Optimized link state routing(OLSR) protocol". Internet Draft(http://www.supelec-rennes.fr/ren/perso/bjouga/documents/rfcs/olsr/draft-ietf-manet-olsr-07.txt).(Jul. 2002)

[4] R. Ogier, M. Lewis, F.Templin. "Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)". Internet Draft(http://vesuvio.ipv6.cselt.it/internet-drafts/draft-ietf-manet-tbrpf-08.txt).(Apr. 2003)

[5] David B. Johnson, David A. Maltz. "Dynamic Source Routing in Ad Hoc Wireless Networks". In Imielinski and Korth, editors, Mobile Computing, volume 353. Kluwer Academic Publishers. (1996)

[6] C. perkins, E. Royer. "Ad-hoc on-demand distance vector routing(AODV).In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pages 90-100. (1999)

[7] Haiyun Luo, Petros Zerfos, Jiejun Kong, Songwu Lu, Lixia Zhang. "Self-securing Ad Hoc Wireless Networks". Seventh IEEE Symposium on Computers and Communications (ISCC '02) . (2002)

[8] Web site "Nick Szabo's Papers and Concise Tutorials" http://szabo.best.vwh.net/secret.html

[9] Web site "暗号入門:公開鍵暗号方式" http://c4t.jp/introduction/cryptography/cryptography04.html

[10] Web site "暗号入門:CA,PKI, 鍵共有とは?" http://c4t.jp/introduction/cryptography/cryptography07.html

発表文献

[1] 山口健輔, 中山雅哉
アドホックネットワークにおけるセキュアな通信方法の提案
第 3 回情報科学技術フォーラム (FIT2004) M-069