ISDSR：IDベースシーケンシャルアグリゲート署名のDynamic Source Routingへの適用

村中 謙太[1]　矢内 直人[1]　岡村 真吾[2]　藤原 融[1]

概要：無線センサネットワークで脅威となっている攻撃の一つとして経路情報の改ざんがあり，この攻撃の対策として，電子署名を用いたセキュアルーティングプロトコルが検討されている．しかし，現在提案されているものは経由する端末ごとに署名が必要となり，効率的ではない．また，公開鍵暗号方式が使用され，公開鍵証明書も必要となり，メモリサイズや検証の負荷がさらにかかる．そこで，本稿では複数の署名を一つに集約でき，かつ任意の ID を公開鍵として使用できる ID ベースシーケンシャルアグリゲート署名 (IBSAS) に注目し，より効率的なセキュアルーティングプロトコルである ISDSR を提案する．

キーワード：セキュアルーティングプロトコル，Dynamic Source Routing，電子署名，ID ベース署名，ID ベースアグリゲート署名，ID ベースシーケンシャルアグリゲート署名

An Application of ID-based Sequential Aggregate Signature to Dynamic Source Routing

Kenta Muranaka[1]　Naoto Yanai[1]　Shingo Okamura[2]　Toru Fujiwara[1]

Abstract: An adversary can attack wireless sensor networks by utilizing false route information. A countermeasure against the attack is a secure routing protocol with digital signatures. However, existing secure routing protocols are inefficient because the memory size and the computational overhead are heavy. To overcome these problems, we focus on ID-based sequential aggregate signatures (IBSAS) where users can aggregate individual signatures into a single signature, and certificates of public keys are unnecessary. We propose a secure dynamic source routing with IBSAS, named ISDSR.

Keywords: secure routing protocol, dynamic source routing, digital signature, ID-based signature, ID-based aggregate signature, ID-based sequential aggregate signature

1. はじめに

1.1 背景

セキュアルーティングプロトコルとは，電子署名を用いて経路情報の正当性を保証することができるルーティングプロトコルである．セキュアルーティングプロトコルは，偽装された経路情報を流す攻撃に有効であり，無線センサネットワークへの適用が期待されている [1], [2], [3], [4], [5], [6], [7], [8], [9]．本稿で言うセキュアルーティングプロトコルとは，経路情報の正当性を保証するためにルーティングプロトコルで交換される経路情報に電子署名を導入したものである．主なプロトコルの流れとしては，ノードが経路情報を含むパケットを送信するときにパケットに自身の電子署名を追加して送信し，パケットを受け取ったノードはパケットに付いている電子署名を検証し，そのパケットの経路情報の正当性を確認するというものである．しかし，このような方式では，ノードごとにパケットに署名を付加するため経由するノードが増えるにつれてパケットサイズが増える．そのためノードにおいて

[1] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University
[2] 奈良工業高等専門学校
National Institute of Technology, Nara Collage

そのパケットを格納しておくために多くのメモリが消費され，すべての署名を検証するため計算時間の負荷も大きくなる．他にも，公開鍵暗号を使用した場合公開鍵証明書が必要となり，その取得や検証も大きな負荷となる．また，センサネットワークの端末は小型で放電容量が小さい．そのため，省電力な負荷の小さい計算が望ましく，既存方式では負荷が大きいため実用的ではない．

著者らは電子署名による負荷を削減させた方式として，署名長が人数に対して定数長となる多人数署名を dynamic source routing （DSR） [10] に導入した Secure-DSR [11] を提案している．本稿は，この Secure-DSR に具体的な署名方式として ID ベースシーケンシャルアグリゲート署名（IBSAS） [12] を導入することで、さらなる効率化を図っている．

1.2 貢献

この論文の貢献は，複数の署名を一つに集約することができる多人数署名と任意の識別子を公開鍵と使用できる ID ベース署名を組み合わせた ID ベースシーケンシャルアグリゲート署名（IBSAS) を用いて，Secure-DSR を発展させた ISDSR を提案したことである．具体的には，IBSAS の ID ベース署名の性質により，公開鍵証明書が不要な初のセキュアルーティングプロトコルとして効率化を図った．また，Secure-DSR で言及されていなかった Route Maintenence を拡張させた Secure Route Maintenance と鍵管理を行う Key Revocation や Key Update を提案した．他にも，ISDSR のパケットサイズが既存方式と比べて最小であることを示し，暗号実装を通じて ISDSR の計算速度を見積もった．

1.3 関連研究

効率化を目指すため多人数署名を用いたセキュアルーティングプロトコルもいくつか提案されている．一つ目は，Kim と Tsudik が提案した the secure route discovery protocol（SRDP） [6] である．このプロトコルでは，シーケンシャルアグリゲート署名 [13] とアカウンタブルサブグループ多人数署名 [14] が使われている．また，Ghosh と Datta はシーケンシャルアグリゲート署名を用いた identity based secure AODV（IDSAODV）を提案している．IDSAODV は，名前に “ID” と付いているが ID ベース署名を使用していない．そのため，既存方式はすべて公開鍵証明書が必要である．本稿では，多人数署名と ID ベース暗号を組み合わせた IBSAS により更なる効率化を目指す．

2. 要素技術

2.1 Dynamic Source Routing protocl

Dynamic Source Routing protocl（DSR） [10] とは，アドホックネットワークにおいて機能するように作られたルーティングプロトコルで，ネットワークインフラが不要で，自律的な無線ネットワークの構築が可能である．DSR は主に “Route Discoverry” と “Route Maintenance” の二つの機能をもつ．Route Discovery では，転送が必要なデータが発生した際に経路を探索，確立を行う．また，Route Maintenance では，ノードの離脱などによって経路が破壊された場合の復旧などを行う．

2.2 ID ベースシーケンシャルアグリゲート署名

ID ベース多人数署名は複数の署名を一つの署名に集約できる性質と任意の ID を公開鍵として使用できる性質を併せ持つ．一つ目の性質より，署名者の数によらず署名サイズを一定にすることができる．二つ目の性質より，公開鍵証明書が不要である．現在，ID ベースアグリゲート署名（IBAS） [15] と ID ベースシーケンシャルアグリゲート署名（IBSAS） [12] の 2 種類の ID ベース多人数署名方式が提案されている．この二つの方式の違いは署名集約のタイミングである．IBAS では，任意の署名者が個別に生成された署名を集約することができる．IBSAS では，それぞれの署名者が署名生成と集約を同じアルゴリズム内で行う．つまり IBSAS は署名者間で一つの署名をやり取りする．このことから，ノード間で通信をする署名の数が少ないほうがよいセキュアルーティングプロトコルでは IBSAS の機能で十分であると言える．また他の違いとしては，IBAS ではタイムスタンプが必要であるが，IBSAS では不要である．今回拡張を考えている DSR はタイムスタンプを使用していないため，IBSAS のほうが適している．以上のことから，本稿では IBSAS に注目する．

2.2.1 準備

$\mathbb{G}, \mathbb{G}_T$ を素数位数 p の群とする．ペアリングとは効率的に計算ができる写像 $\mathbf{e}$: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ のことで，次の二つの性質を持っている．

- 双線形性: すべての $u, v \in \mathbb{G}$ と $a, b \in \mathbb{Z}_p$ に対して，$\mathbf{e}(au, bv) = \mathbf{e}(u, v)^{ab}$ が成り立つ．
- 非退化性: 任意の元 $g \in \mathbb{G}^*$ に対して，$\mathbf{e}(g, g) \neq 1_{\mathbb{G}_T}$ が成り立つ．すなわち $\mathbf{e}(g, g)$ は群 $\mathbb{G}_T$ を生成する．$1_{\mathbb{G}_T}$ は群 $\mathbb{G}_T$ の単位元を表す．

また，$(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e})$ を双線形写像パラメータと呼ぶ．

2.2.2 アルゴリズム

ID ベースシーケンシャルアグリゲート署名（IBSAS）は **Setup**，**KeyDerivation**，**Sign**，**Verification** の四つのアルゴリズムで構成される．それぞれのアルゴリズムの内容は **Algorithm** 1~4 で示す．

Algorithm 1 Setup

Ensure: マスター公開鍵 mpk, マスター秘密鍵 msk

1: 双線形写像パラメータ $(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e})$ を生成する．
2: $g_1, g_2 \xleftarrow{R} \mathbb{G}$
3: $\alpha_1, \alpha_2 \xleftarrow{R} \mathbb{Z}_p$
4: $\alpha_1 \cdot g, \alpha_2 \cdot g$ を計算する．
5: ハッシュ関数 $\mathbf{H}_1, \mathbf{H}_2 : 0,1^* \to \mathbb{G}, \mathbf{H}_3 : 0,1^* \to \mathbb{Z}_p^*$ を用意する．
6: **return** $(\mathbb{G}, \mathbb{G}_T, \mathbf{e}, g, \alpha_1 g, \alpha_2 g, \{\mathbf{H}_i\}_{i=1,\ldots,3})$ を mpk とし，(α_1, α_2) を msk とする．

Algorithm 2 KeyDerivation

Require: $msk, ID \in \{0,1\}^*$

Ensure: ID に対応した秘密鍵 sk_{ID}

1: **return** $(\alpha_1 \mathbf{H}_1(ID), \alpha_2 \mathbf{H}_2(ID))$ を sk_{ID} とする．

Algorithm 3 Signing

Require: $sk_{ID_i}, m \in \{0,1\}^*, \sigma',)$

Ensure: 署名 σ'

1: if $i = 1$, σ は $(1_{\mathbb{G}}, 1_{\mathbb{G}}, 1_{\mathbb{G}})$ で定義される．
2: σ を $(\sigma_1, \sigma_2, \sigma_3)$ に分解する．
3: $r, x \xleftarrow{R} \mathbb{Z}_p$
4: $\sigma_3' \leftarrow \sigma_3 + xg$
5: $\sigma_2' \leftarrow \sigma_2 + rg$
6: $\sigma_1' \leftarrow \sigma_1 + r\sigma_3 + x\sigma_2' + \alpha_2 \mathbf{H}_2(ID_i) + \mathbf{H}_3(ID_i||m_i)\alpha_1 \mathbf{H}_1(ID_i)$
7: **return** $(\sigma_1', \sigma_2', \sigma_3')$ を ID が生成した署名とする．

Algorithm 4 Verification

Require: $mpk, ((ID_1, m_1), \ldots, (ID_n, m_n)), \sigma$

Ensure: 1 または 0

1: $ID_1, \ldots, ID_n$ がそれぞれ異なっているか確認する．重複がある場合は 0 を出力する．
2: σ を $(\sigma_1, \sigma_2, \sigma_3)$ に分解
3: $\mathbf{e}(\sigma_1, g) \stackrel{?}{=} \mathbf{e}(\sigma_2, \sigma_3) \cdot \mathbf{e}(\Sigma_{i=1}^n \mathbf{H}_2(ID_i), \alpha_2 g) \cdot \mathbf{e}(\Sigma_{i=1}^n \mathbf{H}_3(ID_i||m)\mathbf{H}_1(ID_i), \alpha_1 g)$
4: **return** 一致するならば 1 を，しないならば 0 を出力する．

3. ISDSR のモデル

3.1 機能

セキュアルーティングプロトコルにおいて，論文 [5] では経路を確立する際には五つの安全性要件を満たすことが求められている．しかし，本稿ではその要件と同等の機能を以下の二つの要件にまとめることができると考えるため，ISDSR はこの二つの要件を満たすように設計する．

(1) 攻撃者は他のノードになりすますことができない

(2) 経路情報を改ざんすることができない

上記の二つの要件を達成することを「経路情報の正当性を保証する」と定義する．ISDSR では，経路を探索するパケットを受け取ったノードが経路情報に自身の電子署名を付加していくことで経路情報の正当性を保証する．

3.2 仮定

ノード仮定 ネットワークを構成するノードには，センサノードと管理サーバの二種類存在する．今後センサノードのことを単にノードと呼ぶこととする．管理サーバは Key Generation Center（KGC）の役割を果たしているため，IBSAS のアルゴリズムである **Setup** で生成されるマスター秘密鍵，マスター公開鍵を所持している．ノードは識別子とその識別子から生成された秘密鍵とマスター公開鍵を所持している．また各ノードは管理サーバと暗号化通信するための共通鍵を持っており，ID を更新する際に管理サーバと暗号化通信を行う．

ネットワーク仮定 本稿が仮定しているネットワークは双方向通信を行っている．

3.3 攻撃モデル

真ノードを管理サーバが管理しているノード，攻撃ノードを管理サーバが管理していないノードと定義する．真ノードはプロトコル通りに動作をする．攻撃者の目的は，真ノードのみで構成されてすでに機能しているネットワークに攻撃ノードを設置し，攻撃を行うことである．今回想定しているネットワークは通信が公開されており，攻撃者はデータを盗聴改ざんすることができる．無線ネットワークの物理層はジャミングのような DoS 攻撃に脆弱であるが本稿ではこのような攻撃は考えない．また，同様に Media Access Control（MAC）プロトコルでの攻撃も考えない．攻撃者は真ノードのコントロールを奪うことはできない．そして，真ノードを物理的に奪取することや破壊することは考えない．攻撃者はパケットが攻撃ノードを経由するように経路生成時に経路情報の偽造，改ざんを行う．この攻撃モデルは論文 [5] で定義されている “managed-open” というレベルに相当する．

4. ISDSR の構成

ISDSR は **Setup**，**Secure Route Discovery**，**Secure Route Maintenance**，**Key Management** の 4 つの機能を持っている．この章で使用する表記を表 1 にまとめる．

4.1 Setup

この機能はネットワークを構成する前に行われる．まず管理サーバがマスター秘密鍵とマスター公開鍵を IBSAS の **Setup** で生成する．次に，管理サーバはそれぞれのノードに ID を発行し，その ID に対応した秘密鍵を **KeyDerivation** で生成する．管理サーバは，それぞれのノードに対し共通鍵も生成する．この共通鍵は **Key Management** で使用する．そして，それぞれのノードは ID，秘密鍵，マスター公開鍵，共通鍵をプリインストールする．

表 1　表記とその説明

表記	説明
ϕ	空集合 t
$ID_a \| \| ID_b$	ID_a と ID_b の連結,
ID_s	送信元ノードの識別子
ID_d	宛先ノードの識別子
ID_i	第 i 経由ノードの識別子
ID_{MS}	管理サーバの識別子
ID_c	確認ノードの識別子
ID_e	切断したノードの識別子
ID_r	使用禁止にするノードの識別子
ID_u	鍵更新をするノードの識別子
msk	マスター秘密鍵
mpk	マスター公開鍵
sk_{ID_i}	ID_i の秘密鍵
σ_{ID_i}	ID_i が **Signing** で生成した署名
hd	$SRREQ$ のヘッダ
RI	経路情報
$SRREQ_{ID_i}$	ID_i によって生成された $SRREQ$
EI	エラー情報
ET	エラータイプ番号
k_{ID}	ノード ID と管理サーバで共有している共通鍵
$Enc(k, m)$	m を共通鍵 k を使用して生成した暗号文
$Dec(k, c)$	c を共通鍵 k で復号した平文

4.2　Secure Route Discovery

Secure Route Discovery では多人数署名が付加された Secure Route Request パケット（$SRREQ$）と Secure Route Reply パケット（$SRREP$）を使用し，経路の発見，確立を行う．

Secure Route Discovery の探索処理は各ノードが署名をパケットに付加することを除いて，DSR の Route Discovery の探索処理と同じである．すなわち，Fig 1 のように $SRREQ$ は RREQ に多人数署名を付加したものとなる．まず送信元ノードは自身の ID を経路情報に加えたのち，経路情報をメッセージとし IBSAS のアルゴリズム **Signing** で署名を生成し，$SRREQ$ に付加してブロードキャストを行う．以降，$SRREQ$ を受け取ったノードは自身の ID を経路情報に加えたのち，経路情報をメッセージとし，$SRREQ$ に付いている署名を使って **Signing** を行い新たな署名を生成する．その新たな署名を $SRREQ$ に付加してブロードキャストを行う．その後，$SRREQ$ が宛先ノードに到達した場合，宛先ノードは経由したノードの ID を用いて，**Verification** で $SRREQ$ に付加されている署名を検証する．署名の検証結果が 1 ならば応答処理に移り，0 ならば検証結果が 1 になるような署名が付加されている $SRREQ$ を受信するまで待つ．応答処理では，経路情報をメッセージとして宛先ノードが **Signing** で署名を生成し $SRREP$ に付加して送信元ノードに返信する．$SRREP$ を受け取った送信元ノードは宛先ノードの ID を用いて，付加されている署名を **Verification** で検証する．結果が 1 ならば経路を確立し，0 ならばその $SRREP$ を破棄する．

Secure Route Discovery のアルゴリズムの詳細は以下のとおりである．$SRREQ$ は図 1 のようにヘッダ hd と経路情報 RI と署名 Σ の三つの要素を持っている．$SRREP$ も同様にヘッダと経路情報 RI と署名 σ の三つの要素を持っている．ヘッダは DSR と同様に構成し，RI は，宛先ノードの識別子，送信元ノードの識別子，経由したノードの識別子を要素として持っている．$SRREP$ の宛先ノードは $SRREQ$ での送信元ノードで，送信元ノードは $SRREQ$ での宛先ノードである．経由したノードは $SREEQ$ と逆順で格納されている．また，**SRREQConstruct**，**SRREPConstruct**，**RoopCheck**, **SRREEPHeaderRenewal** をそれぞれ **Function**1∼4 で定義する．**SRREEQConstruct** はヘッダ，経路情報，署名を入力としてヘッダを更新してその三つを要素とする $SRREQ$ パケットを構築する関数である．**RoopCheck** は経路情報とノードの識別子を入力として，経路情報にすでにそのノードの識別子が含まれているかチェックする関数である．つまり，経路ループが存在するかチェックする．**SRREEPConstruct** は経路情報と署名を入力として，$SRREP$ を構築する関数である．**SRREEPHeaderRenewal** は $SRREP$ を入力とし，ヘッダを更新した $SRREP$ を構築する関数である．そして，**Algorithm**5∼10 が Secure Route Discovery のアルゴリズムである．

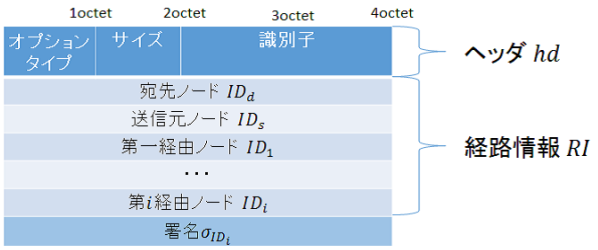

図 1　$SRREQ_{ID_i}$

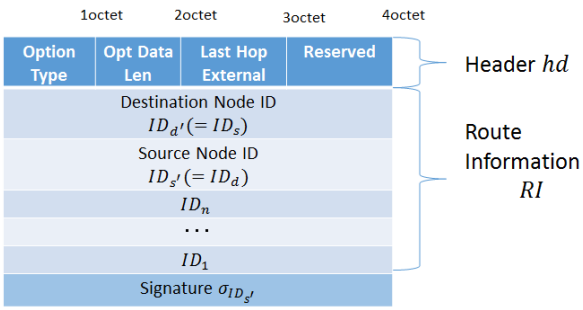

図 2　$SRREP$

Function 1 SRREQConstruct

Require: RI, σ, hd
Ensure: $SRREQ = (hd, RI, \sigma)$
1: hd を更新する
2: $SRREQ \leftarrow (hd,\ RI,\ \sigma)$
3: **return** $SRREQ$

Function 2 LoopCheck

Require: RI, ID
Ensure: 1 or 0
1: **if** $ID \in RI$ **then**
2: **return** 1
3: **else**
4: **return** 0
5: **end if**

Function 3 SRREPConstruct

Require: RI', σ
Ensure: $SRREP = (hd, RI, \sigma)$
1: hd を構成する．
2: $RI \leftarrow (RI'$ の逆順 $)$
3: $SRREP \leftarrow (hd, RI, \sigma)$
4: **return** $SRREP$

Function 4 SRREPHeaderUpdate

Require: $SRREP = (hd', RI, \sigma)$
Ensure: $SRREP = (hd, RI, \sigma)$
1: ヘッダを更新する．
2: **return** $SRREP$

Algorithm 5 送信元ノードの探索処理

Require: sk_{ID_s}, ID_s, ID_d
Ensure: $SSREQ_{ID_s} = (hd_0, RI_0, \sigma_{ID_s})$
1: $RI_0 \leftarrow \phi$
2: $hd \leftarrow \phi$
3: $RI_0 \leftarrow RI_0 || ID_d || ID_s$
4: $\sigma_{ID_s} \leftarrow$ **Signing**$(sk_{ID_s}, RI_0, 1)$
5: $SSREQ_{ID_s} \leftarrow$ **SRREQConstruct**(hd, RI_s, σ_s)
6: **return** $SSREQ_{ID_s}$

Algorithm 6 第 i 経由ノードの探索処理

Require: ID_i, sk_{ID_i},
$SRREQ_{ID_{i-1}} = (hd_{i-1}, RI_{i-1}, \sigma_{ID_{i-1}})$
Ensure: $SSREQ_{ID_i} = (hd_i, RI_i, \sigma_{ID_i})$ or 0
1: **if** **LoopCheck**$(RI_{i-1}, ID_i) = 1$ **then**
2: $SRREQ_{ID_{i-1}}$ を破棄する．
3: **return** 0
4: **else**
5: $RI_i \leftarrow RI_{i-1} || ID_i$
6: $\sigma_{ID_i} \leftarrow$ **Signing**$(sk_{ID_i}, RI_i, \sigma_{ID_{i-1}})$
7: $SSREQ_{ID_i}$
$\leftarrow$ **SRREQConstruct**$(hd_{i-1}, RI_i, \sigma_{ID_i})$
8: **return** $SSREQ_{ID_i}$
9: **end if**

4.3 Secure Route Maintenance

あるノードがデータ転送中に次のホップへのリンク切断を確認したとする．確認ノードは Secure Route Error

Algorithm 7 宛先ノードの探索処理

Require: $SRREQ_{ID_n} = (hd_n, RI_n, \sigma_{ID_n})$
Ensure: 1
1: **while** **Verification**$(mpk, RI_n, \sigma_{ID_n}) \neq 1$ **do**
2: 次の $SRREQ$ を待つ
3: **end while**
4: **return** 1

Algorithm 8 宛先ノードの応答処理

Require: sk_{ID_d}, $SRREQ_{ID_n} = (hd_n, RI_n, \sigma_{ID_n})$
Ensure: $SRREP = (hd, RI, \sigma_{ID_d})$
1: $\sigma_{ID_d} \leftarrow$ **Signing**$(sk_{ID_d}, RI_n, 1)$
2: $SRREP \leftarrow$ **SRREPConstruct**(RI_n, σ)
3: **return** $SRREP$

Algorithm 9 第 i 経由ノードの応答処理

Require: $SRREP = (hd', RI, \sigma_{ID_d})$
Ensure: $SRREP = (hd, RI, \sigma_{ID_d})$
1: $SRREP \leftarrow$ **SRREPHeaderUpdate**$(SRREP)$
2: **return** $SRREP$

Algorithm 10 送信元ノードの応答処理

Require: $SRREP = (hd, RI, \sigma_{ID_d})$
Ensure: 0 or 1
1: **if** **Verification**$(mpk, \sigma_{ID_d}, RI) = 1$ **then**
2: 経路確立
3: **return** 1
4: **else**
5: $SRREP$ を破棄する．
6: **return** 0
7: **end if**

パケット（SRERR）を送信元ノードに送信する．Secure Route Maintenance では，SRERR のエラータイプは 3 が設定される．SRERR の構成図は図 3 のようになっており，確認ノードが **Signing** で生成した署名が付加されている．確認ノードと送信元ノードの間にある経由ノードは何もせずに送信元ノードまで SRERR を転送する．送信元ノードは確認ノードの ID で署名を検証し，切断されたリンクを使わない経路を使ってデータ転送を再開する．**Algorithm**11〜12 が Secure Route Maintenance のアルゴリズムである．

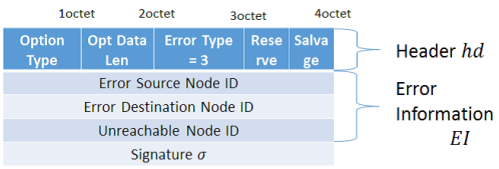

図 3 SRERR(Secure Route Maintenance)

Function 5 SRERRConstruct

Require: EI, σ, ET
Ensure: $SRERR$
1: エラータイプ番号が ET のヘッダ hd を構成する．
2: $SRERR \leftarrow (hd, EI, \sigma)$
3: **return** $SRERR$

Algorithm 11 確認ノード

Require: sk_{ID_c}, ID_c, ID_s, ID_e
Ensure: $SRERR$
1: $EI \leftarrow \phi$
2: $EI \leftarrow EI||ID_c||ID_s||ID_e$
3: $\sigma_{ID_d} \leftarrow$ **Signing**$(sk_{ID_c}, EI, 1)$
4: $SRERR \leftarrow$ **SRERRConstruct**$(EI, \sigma_{ID_c}, 3)$
5: **return** $SRERR$

Algorithm 12 送信元ノード

Require: $SRERR = (hd, EI, \sigma_{ID_c})$
Ensure: 1 or 0
1: **if** **Verification**$(mpk, \sigma_{ID_c}, ID_c) = 1$ **then**
2: 切断リンク確認
3: **return** 1
4: **else**
5: SRERR 破棄
6: **return** 0
7: **end if**

5. 鍵管理

ID の使用禁止と更新について鍵管理方法を新たに考えた．この鍵管理では，RERR を拡張した $SRERR$ を使用している．$SRERR$ では，エラータイプとして新たに Key Revocation タイプ（エラータイプ=4）と Key Update タイプ（エラータイプ=5）を付け加えた．

5.1 Key Revocation

あるノードの ID を公開鍵として使用禁止にする際，管理サーバはその ID と管理サーバの署名を付加してブロードキャストする．使用禁止にする ID を受け取ったノードは署名を検証したのち，その ID を使用している経路をキャッシュから削除する．またその ID をキャッシュしておき，経路生成する際その ID が含まれていないかチェックする．具体的には，RERR パケットを拡張した $SRERR$ パケットを使用する．$SRERR$ は図 4 のように構成されている．エラータイプは 4 に設定し，エラー送信元ノードには管理サーバの ID，エラー宛先ノードにはブロードキャスト，そして使用禁止にする ID を記述し，さらに管理サーバの署名を付加する．**Algorithm**13~14 が Key Revocation のアルゴリズムである．

5.2 Key Update

管理サーバは新たな ID を発行し，その ID に対応した秘密鍵を **KeyDerivation** で生成する．管理サーバは新た

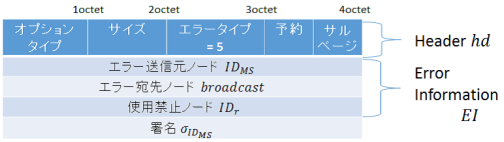

図 4 $SRERR$（Key Revocation）

Algorithm 13 管理サーバ

Require: $ID_{MS}, ID_r, sk_{ID_{MS}}$
Ensure: $SRERR$
1: $EI \leftarrow \phi$
2: $EI \leftarrow EI||ID_{MS}||Broadcast||ID_r$
3: $\sigma_{ID_{MS}} \leftarrow$ **Signing**$(sk_{ID_{MS}}, EI, 1)$
4: $SRERR \leftarrow$ **SRERRConstruct**$(EI, \sigma_{ID_{MS}}, 4)$
5: **return** $SRERR$

Algorithm 14 ノード

Require: $SRERR = (hd, EI, \sigma_{ID_{MS}})$
Ensure: $SRERR$ or 0
1: **if** **Verification**$(mpk, \sigma, ID_{MS}) = 1$ **then**
2: キャッシュから ID_r が含まれている経路を消去する．
3: 使用禁止 ID として ID_r をキャッシュに保存する．
4: $SRERR$ をブロードキャストする．
5: **return** $SRERR$
6: **else**
7: $SRERR$ を破棄する．
8: **return** 0
9: **end if**

な ID を発行したノードに $SRERR$ で新たな ID と秘密鍵，さらに管理サーバの署名を付加して送信する．このとき，秘密鍵はノードと管理サーバ間で共有されている共通鍵で暗号化し送信する．その後，更新される前の ID を **Key Revocation** で使用禁止にする．このときの $SRERR$ は図 5 のように構成されている．エラータイプは 5 に設定しエラー送信元ノードには管理サーバの ID，エラー宛先ノードには新たに ID を発行したノードの ID，そして新たな ID とその ID に対応した秘密鍵（暗号化）を記述し，さらに管理サーバの署名を付加する．**Algorithm**15~16 が Key Update のアルゴリズムである．

図 5 $SRERR$（Key Update）

Algorithm 15 管理サーバ

Require: ID_{MS}, $sk_{ID_{MS}}$
Ensure: $SRERR$
1: $EI \leftarrow \phi$
2: $EI \leftarrow EI||ID_{MS}||ID_u||ID_{new}||Enc(k_{ID_u}, sk_{ID_{New}})$
3: $\sigma_{ID_{MS}} \leftarrow$ **Signing**$(sk_{ID_{MS}}, EI, 1)$
4: $SRERR \leftarrow$ **SRERRConstruct**$(EI, \sigma_{ID_{MS}}, 5)$
5: revoke ID_u in **Key Revocation**
6: **return** $SRERR$

Algorithm 16 更新ノード

Require: $SRERR = (hd, EI, \sigma_{ID_{MS}})$
Ensure: 1 or 0
1: **if** **Verification**$(mpk, \sigma_{ID_{MS}}, ID_{MS}) = 1$ **then**
2: $ID_u \leftarrow ID_{New}$
3: $sk_u \leftarrow Dec(k_{ID_u}, Enc(k_{ID_u}, sk_{ID_{New}}))$
4: 使用禁止 ID として ID_u をキャッシュに保存する.
5: **return** 1
6: **else**
7: $SRERR$ を破棄する.
8: **return** 0
9: **end if**

6. 評価

6.1 性能評価

ISDSR は，IBSAS を導入することで本来のルーティングプロトコルに比べてどれほど追加で負荷が増加したか二つの視点から評価する．まず，一つ目は経路構築の際に従来のリクエストパケットやリプライパケットからどれほどパケットサイズが増加したか（パケットサイズ増加量）を評価する．二つ目は，経路構築の際に本来のルーティングプロトコルに比べて IBSAS を導入することによってどれほど処理時間が増加するのか（処理時間増加量）を評価する．

6.1.1 パケットサイズ増加量

経路構築時に i 番目（$i > 1$）のノードがリクエストパケット，リプライパケットを次のノードに送信する際，本来のルーティングプロトコルに比べてどれほどパケットサイズが増加しているか評価する．各セキュアルーティングプロトコルにおけるパケットサイズ増加量を表 2 に示す．ISDSR は楕円曲線ベースの署名をパケットに付加するだけであるので，他のセキュアルーティングプロトコルに比べて一番パケットサイズ増加量が小さい．ARAN や SAODV は公開鍵証明書が必要となるためパケットサイズ増加量は大きくなっている．また，IDSAODV は RSA ベースの署名方式を使用しているため，ISDSR よりパケットサイズ増加量が大きい．

6.1.2 処理時間増加量

経路構築時に，本来のルーティングプロトコルに比べてどれほど処理時間がかかるか評価する．具体的には，署名生成，検証や公開鍵証明書の検証でどれくらい処理時間がかかるのかを示す．ここでの評価はネットワークシミュレータを使用して実験したわけではなく，あくまでも各セキュアルーティングプロトコルで使用されている署名方式を実装し，経路構築までに必要な回数分だけ署名の生成検証や公開鍵証明書の検証を行い，そこでかかった処理時間を示す．つまり，署名方式による処理時間を評価する．実装環境は，表 3 の通りである．図 6 は，各セキュアルーティングプロトコルの処理時間増加量を比較したグラフである．グラフより，ARAN と SAODV が ISDSR より処理時間増加量が約 1/4 小さいことが分かる．今後はネットワークシミュレータなどより現実に近い環境にて，通信のオーバーヘッドも含めた処理時間増加量を計測する予定である．

表 2 パケットサイズ増加量（i 番目ノード）

ルーティングプロトコル	パケットサイズ増加量（バイト）
ISDSR	84
ARAN	2512
SAODV	1320
IDSAODV	$16i + 256$

表 3 実験環境

ホスト OS	Windows 8 Enterprise
仮想 OS	Ubuntu 16.04 LTS
コンパイラ	gcc version5.3.1
ライブラリ	pbc-0.5.14, openssl 1.02g
楕円曲線パラメータ	param-a
仮想環境	Virtual Box version 5.0.20
プロセッサ	Intel®Core^{TM}i7 CPU 870 @ 2.93GHz
メモリ	2GB

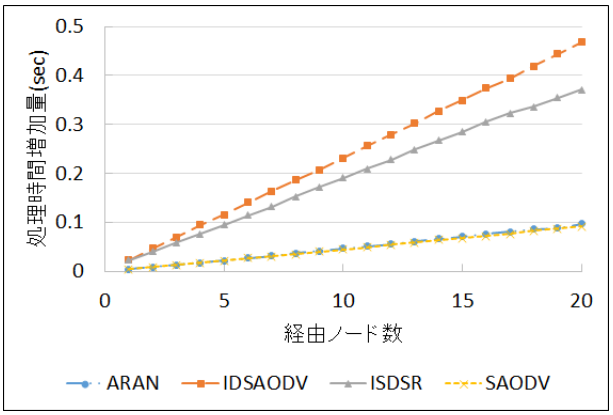

図 6 処理時間増加量

6.2 安全性解析

ISDSR では，攻撃者が物理的にノードを奪取することや真ノードのコントロールを奪うことは考えていないので攻撃ノードは他のノードの秘密鍵を知り得ない．したがって，攻撃ノードは署名が偽造不可能なものである限り，署名を偽造することができない．このことから，ISDSR は

3.1 節で述べた要件 1 を満たすことが分かる．また，管理サーバには真ノードの公開鍵しか登録されていない．そのため，経路情報が改ざんされた場合や経路情報に攻撃ノードが含まれている場合，管理サーバは署名の検証に失敗するため経路を確立できない．したがって，ISDSR では経路が確立された場合，その経路情報が改ざんされていないことや経路途中に攻撃ノードが含まれていないことを保証できる．このことから，SDSR は 3.1 節で述べた要件 2 を満たすことが分かる．したがって，ARAN [5] と同等の “managed-open”の安全性を持つと言える．

7. おわりに

本稿では，センサネットワークのセキュリティの重要性からセンサネットワークのルーティングプロトコルである DSR をセキュアルーティングプロトコルに拡張した ISDSR の安全性要件を整理し提案した．また，ISDSR は IBSAS の利用によりパケットサイズが既存方式に比べて最小であることを示した．他にも，ISDSR は悪意のあるノードが経路内に存在しないことを保証できることを示した．今後の課題としては，ネットワークシミュレータを用いた ISDSR の実装と性能評価を行うことが挙げられる．ネットワークシミュレータとしては NS-3[*1] の利用を考えている．また，Arnaud ら [16], [17] や Zhang ら [18] のような形式検証を使ったより厳密な ISDSR の安全性解析も考えている．他にも，ISDSR の速度改善も考えている．具体的には，IBSAS の計算の一部を事前に行っておくことで署名生成の時間を削減できると考えている．

8. 謝辞

本研究の一部は，JSPS 科研費 16K16065 の助成を受けている．

参考文献

[1] L. Zhou and Z.J. Haas. Securing ad hoc network. *IEEE Network Magazine*, Vol. 13, No. 6, pp. 24–30, 1999.

[2] Y.-C Hu, A. Perrig, and D.B. Johnson. Ariadne: a secure on demand routing protocol for ad hoc network. In *Proc. of MobiCom 2002*. ACM, 2002.

[3] Y.-C Hu, A. Perrig, and D.B. Johnson. Ariadne: a secure on demand routing protocol for ad hoc network. *Wireless Networks*, Vol. 11, pp. 21–38, 2005.

[4] M.G. Zapata and N. Asokan. Securing ad hoc routing protocols. In *Proc. of WISE*, pp. 1–10. ACM Press, 2002.

[5] K. Sanzgiri, D. LaFlamme, B. Dahill, B. Neil Levine, C. Shields, and E.M. Belding-Royer. Authenticated routing for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 3, pp. 598–610, 2005.

[6] J. Kim and G. Tsudik. Srdp: Secure route discovery for dynamic source routing in manets. *Ad Hoc Networks*, Vol. 7, No. 6, pp. 1097–1109, 2009.

[7] U. Ghosh and R. Datta. Identity based secure aodv and tcp for mobile ad hoc networks. In *Proc. of ACWR 2011*, pp. 339–346. ACM, 2011.

[8] U. Ghosh and R. Datta. Sdrp: Secure and dynamic routing protocol for mobile ad-hoc networks. *IET Network*, Vol. 3, No. 3, pp. 235–243, 2013.

[9] K. Muranaka, N. Yanai, S. Okamura, and T. Fujiwara. Secure routing protocols for sensor networks: Construction with signature schemes for multiple signers. In *Proc. of Trustcom 2015*, pp. 1329–1336. IEEE, 2015.

[10] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, Vol. 353, pp. 153–181, 1996.

[11] 村中謙太, 矢内直人, 岡村真吾, 藤原融. 多人数署名を用いた secure-dsr の提案. In *Proc. of SCIS2016*, 2016. 2E2-1.

[12] A. Boldyreva, C. Gentry, A. O’Neill, and D.H. Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing (extended abstract). In *Proc. of ACM CCS 2007*, pp. 276–285. ACM, 2007.

[13] A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In *Proc. of EUROCRYPT 2004*, Vol. 3027 of *LNCS*, pp. 74–90. Springer, 2004.

[14] S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures: extended abstract. In *Proc. of CCS 2001*, pp. 245–254. ACM, November 2001.

[15] C. Gentry and Z. Ramzan. Identity-based aggregate signatures. In *Proc. of PKC 2006*, Vol. 3958 of LNCS, pp. 257–273. Springer, 2006.

[16] M. Arnaud, V. Cortier, and S. Delaune. Modeling and verifying ad hoc routing protocols. In *Proc. of CSF 2010*, pp. 59–74. IEEE, 2010.

[17] M. Arnaud, V. Cortier, and S. Delaune. Modeling and verifying ad hoc routing protocols. *Information and Computation*, Vol. 238, pp. 30–67, 2014.

[18] F. Zhang, L. Jia, C. Basescu, T. Kim, Y. Hu, and A. Perrig. Mechanized network origin and path authenticity proofs. In *Proc. of ACM CCS 2014*, pp. 346–357. ACM, 2014.

[*1] `https://www.nsnam.org/`