BGPSEC におけるアグリゲート署名の導入

田中 和磨 ∗　矢内 直人 †　岡田 雅之 ‡　金山 直樹 §　西出 隆志 §
岡本 栄司 §

∗ 筑波大学大学院 システム情報工学研究科
305-8573 茨城県つくば市天王台 1-1-1
† 大阪大学大学院 情報科学研究科
565-0871 大阪府吹田市山田丘 1-5
‡ 一般社団法人 日本ネットワークインフォメーションセンター
101-0047 東京都千代田区内神田 3-6-2 アーバンネット神田ビル 4F
§ 筑波大学 システム情報系
305-8573 茨城県つくば市天王台 1-1-1

あらまし 現在 BGP で交換される経路情報の正当性は保証されていない．このため，第三者が経路情報を乗っ取り，不正なサーバにパケットを吸い込むなどの経路のハイジャックが大きな問題となっている．経路情報の正当性を保証するには，経路情報中の ORIGIN AS 情報と AS PATH 属性の 2 つの正当性を保証する必要がある．本稿では AS PATH 属性を保証する点に注目し，現在提案されている BGPSEC にアグリゲート署名を適用する新たな手法を提案する．アグリゲート署名を適用することで署名サイズを小さくすることが可能であるが，その導入手法及び具体的な効率性についてはこれまで不明瞭であった．本稿では実装も行い，性能も評価する．

An Application of Aggregate Signatures to BGPSEC

Kazuma Tanaka∗　Naoto Yanai†　Masayuki Okada‡　Naoki Kanayama§
Takashi Nishide§　Eiji Okamoto§

∗ Graduate School of Systems and Information Engineering, University of Tsukuba,
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan.
†Graduate School of Information Science and Technology, Osaka University,
1-5 Yamadaoka, Suita, Osaka, 565-0871, Japan.
‡Japan Network Information Center,
urbanne-Kanda Bldg 4F, 3-6-2 Uchi-Kanda, Chiyoda, Tokyo, 101-0047, Japan.
§Faculty of Engineering, Information and Systems, University of Tsukuba,
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan.

Abstract The validity of a route advertisement that BGP speakers propagate is not guaranteed now. Therefore, the adversary can overwrite the advertisement to make it an illegal one and attract victim packets to a malicious server, thus causing a big issue. To prevent this, we need to guarantee the validity of both ORIGIN AS Information in a route advertisement and AS PATH attributes. Our study focuses attention on AS PATH attributes validity and we propose a new method that applies an aggregate signature scheme to a traditional BGPSEC. It is known that it is possible to reduce a size of signatures by applying the aggregate signature scheme, but its specification and efficiency is unclear. Thus, we implement the proposed method and give an evaluation about our method.

1 はじめに

(背景) ISP や企業，研究所等の組織は Autonomous System(AS) と呼ばれる各組織が運用する自立したネットワークを保有しており，個々の AS を識別する一意な AS 番号が割り当てられている．この AS 番号を利用して，ネットワーク上で組織間の経路情報をやり取りすることでインターネットは運営されている．現在，唯一の AS 間経路制御プロトコルは Boarder Gateway Protocol (BGP)[23] であり，この BGP でやり取りされている経路情報の正当性は保証されていない．このため，ある AS が不正な経路情報を他の AS に転送することで，あるネットワーク宛のパケットを自分のネットワークに吸い込み，情報を盗み取ったり，パケットを盗聴，改ざんする Man In The Middle(MITM) 攻撃が可能である．具体的な事例としては，2008 年のパキスタンテレコムによる YouTube の経路情報の乗っ取り [19] や bitcoin の換金システムにおける経路情報の乗っ取り [16] が発生している．

BGP でやり取りされる経路情報の正当性を保証するためには，経路情報中の ORIGIN AS 情報と AS PATH 属性の両方の正当性を保証する必要がある．経路情報中の ORIGIN AS 情報の保証とは，ORIGIN AS の AS 番号とこの AS が広告する IP アドレスの組み合わせに対して，これが正しい組み合わせであることを保証することである．これに対して AS PATH 属性の保証とは，経路情報を交換する BGP アップデートメッセージが通ってきた経路の情報を保証することである．前者の ORIGIN AS 情報の保証については，Route Origin Authorization (ROA)[14] と呼ばれる AS 番号と IP アドレスの組み合わせに電子署名を施したデータを利用することで実現でき，既にこの仕組が活用され始めている．一方で，AS PATH 属性の保証については実現手法の検討が不十分であり，考察の余地が多くある．また，最近では DDoS 攻撃対策として経路情報を意図的に書き換えることで，攻撃パケットを吸い込み，正常なパケットのみを攻撃対象のホストに受け渡すサービスが複数の事業者により提供されている [1]．このような経路のハイジャックは運用者の正当な意思の基行われるものであり，現状の仕組みでは正当なハイジャックと不正なハイジャックの区別は難しい．このような理由からも ORIGIN AS 情報だけでなく AS PATH 属性も保証する必要がある．

AS PATH 属性を保証する手法として，BGPSEC の標準化 [12] が Internet Engineering Task Force (IETF) によって行われている．一見するとこの手法を取り入れることで経路情報の正当性確認の達成が期待できる．しかしながら，現在使用されている ECDSA 署名などの従来暗号技術による負荷は膨大であり，実際はこの導入によりルータのメモリ量が不足するという新たな問題が発生しうる．現状の見積では，暗号技術の導入によりルータのメモリ量は数 10G バイトも必要となる [21]．このため，世界中のネットワークルータを高額なメモリを持つものに置き換える必要が生じる．

本稿では，複数の署名を 1 つの小さな署名に集約する機能を持ったアグリゲート署名に注目し，この署名を BGPSEC に適用する新たな手法を提案する．アグリゲート署名を導入することで，署名とメモリサイズの削減が可能となるがこれを導入することで新たな問題が発生する可能性がある．このため，実装及び評価を行い提案手法の有用性と顕在化する問題を明らかにする．

(本稿の貢献) アグリゲート署名を利用して，BGP に AS の経路情報の正当性を保証することができる機能を追加したプロトコルを提案する．技術的な貢献として，署名の事前計算及び不正な署名を発見するトレーシング手法を提案する．現時点でのインターネット上の全ての経路情報は 50 万行以上と言われおり，この数は増加の一途を辿っている．このため，このような莫大な数の経路情報の正当性を保証することが，実用に耐えうる効率性で実現できるか確かめる必要がある．よって，提案したプロトコルを実装し，仮想的なネットワーク環境で検証し，実用性も保証する．

2 関連研究

BGP セキュリティとアグリゲート署名の 2 つ観点から関連研究について言及する．

(BGP セキュリティ) Vervier らの調査 [22] によると経路のハイジャックは 1 日あたり 4.8 件も発生しており，この対策は必須である．BGP セキュリティは 2000 年に Kent らが経路情報に署名を付け正当性を保証する S-BGP[10] の提

案に端を発している．その後，BGP の安全性要件の具体化や実現方法の考察，S-BGP の導入により生ずる問題についての考察を 2003 年に Goodell ら [7] が，2004 年に Hu ら [9] が行っている．顕在化した問題の 1 つに署名付与による BGP アップデートメッセージサイズの増加とメモリ容量不足があり，2005 年に Zhao らはアグリゲート署名を利用してこの問題を解決する手法 [25] を提案した．しかしながら，アグリゲート署名の導入による効率性の変化や問題点などの評価はなされておらず，明らかにする必要がある．これに加え，BGP で DoS 攻撃をする手法について考察を 2011 年に Schuchard ら [20] が，安全な BGP の導入戦略に関する考察を 2011 年に Gill ら [6] が行っている．近年では，証明可能安全性の観点からの議論を 2012 年に Boldyreva ら [2] が行い，提案されている安全な BGP が本当に安全であるのかの考察を 2014 年に Li ら [15] が改めて行っている．更には安全な BGP を実際に部分的に導入したときの評価や問題点を 2013 年に Lychev ら [17] が議論している．また，これまで提案されてきた RPKI を利用した AS PAHT 属性の保証を行う方法とは大きく異なり，MAC を利用して AS PAHT 属性の保証を行う手法を 2014 年に Zhang ら [24] が提案している．

BGP セキュリティを取り巻く環境としては，BGPSEC などを運用する上で欠かすことができない公開鍵基盤である Resource Public Key Infrastructure (RPKI)[13] の環境が活用されはじめ，経路情報中の ORIGIN AS 情報を保証する時に使用される認証データである Route Origin Authorization (ROA)[14] を発行する仕組みも活用されつつある．

(署名方式) アグリゲート署名とは，複数の署名を 1 つのサイズの小さな署名に集約することができる署名方式である．署名作成処理と集約処理を同時に行う順次的なもの [18] と署名作成処理と集約処理を分離可能な一般的なもの [3] がある．2003 年に Boneh らが一般的なアグリゲート署名の提案 [3] を行い，2004 年に Lysyanskaya らが順次的なアグリゲート署名の提案 [18] を行った．本研究では Boneh らの一般的なアグリゲート署名を対象としている．2006 年には Gentry らが ID ベース暗号とアグリゲート署名を組み合わせた方式 [5] や 2015 年には Hohenberger らが任意の署名方式に対して署名の集約が可能な方式 [8] を提案している．

3 BGPSEC

BGP とは，Exterior Gateway Protocol (EGP) の 1 つで，AS 間の標準的な経路制御プロトコルのことである．BGP でやり取りされる経路情報の正当性は保証されておらず，BGP の経路広告が乗っ取られることにより，経路のハイジャックや中間者攻撃などの問題が発生することが予測された．このような背景から，前述の攻撃を防ぐ新たな経路制御プロトコルとして BGPSEC[12] の標準化が IETF によって進められている．

BGPSEC では，経路情報の正当性を保証するために必要とされる経路情報中の ORIGIN AS 情報の保証と AS PATH 属性の保証を提供している．ORIGIN AS 情報の保証に関しては，既に提案されている RPKI を利用して ROA を発行する手法を組み込んでいる．次に，AS PATH 属性の保証に関しては経路交換を行うための BGP アップデートメッセージに電子署名を付加することで，アップデートメッセージが交換されてきた AS PATH 属性の正当性を保証している．

本研究では，BGPSEC の AS PATH 属性を保証する方法について注目するため，この方法について詳しく説明する．

3.1 BGPSEC の AS PATH 属性保証

BGPSEC における AS PATH 属性の保証は，経路情報を含んだ BGPSEC アップデートメッセージを交換する際に各々の BGP ルータ (BGP スピーカ) が電子署名を付与することで実現されている．また，AS 内部のピア (internal peer) 間での交換では署名の付与は行わず，AS 外部のピア (external peer) と交換する際にのみ署名を付与する．前提条件として，RPKI により AS 番号と所持している IP アドレスのプレフィックス，署名に使用するアルゴリズムスイートや公開鍵などが登録されているものとする．以下に具体的な署名生成方法を記述する．経路広告を作成する ORIGIN AS とこの経路広告を交換する AS とでは署名を付与するデータが異なるため注意が必要である．また，表 1 と表 2 のパラメータの詳細については [12] を参照されたい．

(ORIGIN AS の場合)

1. Algorithm Suite Identifier の値を利用して，使用するダイジェストアルゴリズム及び署名アルゴリズムを確認する．
2. 表 1 のデータ列にダイジェストアルゴリズムを適用し，ダイジェスト値を得る．
3. ダイジェスト値に署名アルゴリズムを適用して，署名データを得る．
4. 署名及び署名長を適当なデータフィールドに格納する．

表 1: ORIGIN AS が署名つけるデータ

Target AS Number	(4 octets)
Origin AS Number	(4 octets)
pCount	(1 octets)
Flags	(1 octets)
Algorithm Suite Id.	(1 octets)
NLRI Length	(1 octets)
NLRI Prefix	(variable)

(交換を行う AS の場合) ORIGIN AS の場合と同様に計算する．ただし，署名を付与するデータ列は表 2 であることに注意が必要．

表 2: 交換を行う AS が署名つけるデータ

Target AS Number	(4 octets)
Signer's AS Number	(4 octets)
pCount	(1 octets)
Flags	(1 octets)
Most Recent Sig Field.	(variable)

このように署名を生成することで，BGPSEC のアップデートメッセージが通ってきた経路を保証する．検証は基本的に署名生成と逆の操作を行い，1 つでも検証に失敗した署名があると無効と判断する．また，これらの署名の数はアップデートメッセージが経由した BGPSEC ルータの数に依存するため，経由するルータの数が多くなるとパケットサイズを圧迫する．

3.2 BGPSEC の脅威モデル

先に述べたように BGPSEC では，経路情報中の ORIGIN AS 情報の保証と AS PATH 属性の保証のみを提供している．一方で，アップデートメッセージの正当性や通信パケットがアップデートメッセージが伝搬されてきた経路を通ることは保証していない．また，正当な AS PATH 中に悪意のある AS がおり，アップデートメッセージを意図的に廃棄しているようなことを検知することもできない．

4 アグリゲート署名導入について

アグリゲート署名の最大の利点は複数の署名を，署名の数に関わらず，固定サイズの小さな 1 つの署名に集約できることである．一方でこの署名の欠点は，複数の署名を集約してしまうため集約した署名の検証に失敗した場合，集約した複数の署名のうちどの署名が検証に失敗したか特定できないことである．また，順次的なアグリゲート署名では集約処理と署名処理を分離することができないため，集約処理を後でまとめて行う場合などは適用が難しい．

以下に本研究で対象とする Boneh らのアグリゲート署名 [3] のアルゴリズムを示す．この方式では，任意のそれぞれの値が異なるメッセージ $M_i \in \{0,1\}^*$ のみにしか適用できない．この方式は 5 つのアルゴリズム：$Setup$ (Algorithm 1), $KeyGeneration$ (Algorithm 2), $Signing$ (Algorithm 3), $Aggregation$ (Algorithm 4), $Aggregate\ Verification$ (Algorithm 5) から構成される．

Algorithm 1 Setup

Ensure: The computable isomorphism ψ from $\mathbb{G}_2$ to $\mathbb{G}_1$, and the bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, with target group $\mathbb{G}_T$, are system parameters. The scheme employs a full-domain hash function $H : \{0,1\}^* \to \mathbb{G}_1$, viewed as a random oracle.

1: generate generators g_1, g_2
2: generate the base groups $\mathbb{G}_1, \mathbb{G}_2$ from their respective g_1, g_2
3: **return** $g_1, g_2, \mathbb{G}_1, \mathbb{G}_2$

Algorithm 2 Key Generation

Ensure: a secret key $x \in \mathbb{Z}_p^*$, a public key $v \in \mathbb{G}_2$

1: $x \xleftarrow{R} \mathbb{Z}_p^*$
2: $v \leftarrow xg_2$
3: **return** x, v

Algorithm 3 Signing

Require: x, a message $M \in \{0,1\}^*$
Ensure: signature $\sigma \in \mathbb{G}_1$

1: $h \leftarrow H(M)$
2: $\sigma \leftarrow xh$
3: **return** σ

5 提案手法

基本的には BGPSEC[12] に準拠して，プロトコルを構成する．このため，BGPSEC との差分について詳しく言及する．本稿では BGPSEC の署名方式を一般的なアグリゲート署名にすることによって，経路情報に占める署名サイズの削減を図る．これに加え，署名時間を削減する

Algorithm 4 Aggregation

Require: the aggregating subset of users $U \subseteq \mathbb{U}$, assign to each user an index i, ranging from 1 to $k = |U|$, each user $u_i \in U$ provides a signature $\sigma_i \in \mathbb{G}_1$ on a message $M_i \in \{0,1\}^*$.
Ensure: signature $\sigma \in \mathbb{G}_1$
1: $\sigma \leftarrow \sum_{i=1}^{k} \sigma_i$
2: **return** σ

Algorithm 5 Aggregate Verification

Require: an aggregate signature $\sigma \in \mathbb{G}_1$ for an aggregating subset of users U, the original messages $M_i \in \{0,1\}^*$, public keys $v_i \in \mathbb{G}_2$ for all users $u_i \in U$.
Ensure: the messages M_i are all distinct, and *reject* otherwise.
1: **for** $i = 1$ to $k = |U|$ **do** $h_i \leftarrow H(M_i)$ **end for**
2: **if** $e(\sigma, g_2) = \prod_{i=1}^{k} e(h_i, v_i)$ **then**
3: **return** *accept*
4: **end if**

方法として事前に署名を生成しておくことを提案する．また，アグリゲート署名の署名集約により，どの BGPSEC スピーカが不正な署名を生成したか特定ができなくなる．このため，本稿ではアグリゲート署名を導入した場合でも不正な署名の特定が可能な手法，トレーシングについても考察を行った．

5.1 事前の署名生成

先に述べたように，アグリゲート署名の署名時間は世の中で一般的に使用されている ECDSA 署名よりも計算コストが大きく，署名生成に時間がかかる．このため，この署名時間を短縮するために事前に署名を生成することを考える．BGPSEC[12] において，ORIGIN AS でないルータは表 2 のデータ列に対して署名をつける．ただし，pCount, Flags, Most Recent Sig Field の値を事前に入手することはできない．しかし，AS PATH 属性を保証するためには Target AS Number と Signer's AS Number に対して署名をつければ十分である．このため，提案手法ではこの 2 つの値に対してのみ署名をつける．これによって，署名を予め作成することが可能となる．具体的な手順を以下に示す．

1. Target AS Number(接続している各々ピアの AS 番号) と Signer's AS Number(自身の AS 番号) に対する署名を作成する．
2. アップデートメッセージを受信する．
3. アップデートメッセージの集約された署名に予め作成した適切な署名を集約する．
4. 更新したそれぞれのアップデートメッセージをピアに送信する．

この事前の署名生成は ECDSA 署名でも適用可能であるため，従来の安全な BGP の構成手法でも署名時間を短縮することは可能である．

5.2 トレーシング

検証に失敗した無効な署名の特定手法の直観は，受理状態にある署名が無視可能な連立方程式を構成することである．署名は受理状態にある限り，ある種の等式 (検証式) が成立する．このため，一般に検証式の左辺は右辺で割り切れる．アグリゲート署名においてもこれは同様であり，アグリゲート署名の中の受理状態にある個々の署名はそれぞれ同様に割り切れる．一方で，無効な署名は割り切ることができない．この事実を利用し検証式の辺々を割ることで，無効な署名だけが残る式が構成できる．あとは実際に残った署名がどの値のものであるかを確認するために，独立な連立方程式を構成すればよい．具体的にはアグリゲート署名の集約処理において各署名に固有の乱数を作成し，本来の作成法で用意したアグリゲート署名と個別の乱数を導入したアグリゲート署名を作成する．連立方程式がそれぞれのアグリゲート署名，その解が無効な署名となる．一方，この手法では無効な署名の特定数が予め集約処理に限られることとなる．その数は連立方程式の数，すなわちアグリゲート署名の個数に依存する．このため，より多くの無効な署名を特定したい場合は，アグリゲート署名の数が増加する．実際の運用では発生しうる無効な署名の発生頻度と運用負荷の見積もりを行う必要がある．

以下にトレーシング機能を追加したアグリゲート署名のアルゴリズムを示す．この署名方式は *Setup* (Algorithm 6), *Key Generation* (Algorithm 2), *Signing* (Algorithm 3), *Aggregation* (Algorithm 7), *Verification* (Algorithm 8), *Tracing* (Algorithm 9) の 6 つのアルゴリズムから成り立つ．ただし，*Key Generation* と *Signing* は Boneh らのアグリゲート署名と同じものである．

5.3 提案手法

署名方式に Boneh らのアグリゲート署名を利用する．この署名を利用することで，署名サイズの削減とルーティングテーブルのサイズの削

Algorithm 6 Setup(Tracing Ver.)

Require: security parameters 1^k, 1^τ, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \{0,1\}^* \to \mathbb{Z}_p^*$

$g_1 \xleftarrow{R} \mathbb{G}_1$, $g_2 \xleftarrow{R} \mathbb{G}_2$ // generator
$para \leftarrow (g_1, g_2, H_1, H_2)$
return $para$

Algorithm 7 Aggregation(Tracing Ver.)

Require: $para$, i-tuples $(\{pk_j, m_j, \sigma_j\}_{j=1}^i)$, a secret key sk_{agg} of an aggregator

$R \xleftarrow{R} \{0,1\}^k$ // generate a random number
$\sigma_{agg} \leftarrow$ **Signing**$(para, sk_{agg}, R)$
for $j = 1$ to i **do** $\delta_j = H_2(pk_j \parallel R)$ **end for**
for $k = 1$ to τ **do** $\rho_k = \sum_{j=1}^i \delta_j^k \sigma_j$ **end for**
$\rho_{\tau+1} = \sum_{j=1}^i \sigma_j$
return $\sigma = (\sigma_{agg}, R, \rho_1, \cdots, \rho_{\tau+1})$

減が可能となる．また，署名を事前に生成することで署名生成処理時間の削減を図る．これに加え，本稿の上記に示したトレーシングを利用することで，無効な署名を生成した署名者を検証時に見つけることが可能となる．このトレーシング機能は AS の運用者が機能を有効にするかしないか任意に設定するものとする．ここで示した署名生成や検証以外に関しては BGPSEC に準拠するものとする．

6 評価

本稿では，ペアリングライブラリの TEPLA[11] を使用して ECDSA 署名と Boneh らのアグリゲート署名を実装して速度測定を行った．なお，TEPLA では 128 bit セキュリティを採用している．これは簡易的な評価であるが，従来の BGPSEC で運用した場合と本稿で提案するアグリゲート署名を利用した手法との相対的な比較となり得る．より詳細な評価として The BIRD Internet Routing Daemon や ExaBGP などシミュレーションツールを使用した実装評価も考えられるが，これは今後の課題とする．以下に測定環境及び測定結果を記載する．

OS OS X 10.10.1

Algorithm 8 Verification(Tracing Ver.)

Require: $para, \{pk_j, m_j\}_{j=1}^i$, $\sigma = (\sigma_{agg}, R, \rho_1, \cdots, \rho_{\tau+1})$

if $\boldsymbol{e}(\rho_{\tau+1}, g_2) = \prod_{j=1}^i \boldsymbol{e}(h_j, pk_j)$ where $h_j = H_1(m_j)$ **then**
  return $accept$
else
  return $reject$
end if

Algorithm 9 Tracing(Tracing Ver.)

Require: $para, \{pk_j, m_j\}_{j=1}^i$, $\sigma = (\sigma_{agg}, R, \rho_1, \cdots, \rho_{\tau+1})$

the p_k is the kth polynomial of signatures.
if Verification$(para, pk_{agg}, R, \sigma_{agg}) = reject$ or **Verification**$(para, \{pk_j, m_j\}_{j=1}^i, \rho_{\tau+1}) = accept$ **then**
  return $\emptyset$
else
  for $k = 0$ to τ **do**
    $\alpha_k = \frac{\boldsymbol{e}(\rho_k, g_1)}{\prod_{j=1}^i \boldsymbol{e}(h_j, \delta_j^k pk_i)}$ where $h_j = H_1(m_j)$
  end for
  if δ_j's for all $j \in [1, i]$ such that $\alpha_\tau = \prod_{k=1}^\tau (\alpha_k)^{(-1)^{k-1} p_k}$ holds for all $k \in [1, \tau]$ **then**
    return a set $\mathcal{I} \subset [1, i]$ of indexes corresponding to δ_j's
  else
    return $\emptyset$
  end if
end if

CPU 2.9 GHz Intel Core i7
Memory 8 GB
Library TEPLA 1.0, GMP 6.0.0a

表 3 に平文のサイズを 100 文字分 (100 byte) に固定して 10000 個の平文に対して署名の生成から検証処理までを行った際の平文 1 個あたりのそれぞれの計算時間をまとめている．GDH Sig, Agg Sig, ECDSA Sig はそれぞれ GDH 署名 [4], Boneh らのアグリゲート署名，ECDSA 署名のことであり，初期化は計算するために必要な楕円曲線やペアリングなどの初期化とメモリ確保など初期化処理に要する時間，署名は署名計算処理に要した時間，集約はアグリゲート署名の集約処理に要した時間，検証は検証処理に要した時間，終了処理はメモリ解放など終了処理に要した時間，全体は初期化，署名，集約，検証，終了処理を合算した合計時間を表している．

表 3: 平文 1 個あたりの速度評価 (単位: m sec)

署名方式	初期化	署名	集約	検証	終了処理	全体
GDH	1.1730	1.1478	なし	11.1128	0.0059	13.4397
Agg	1.1602	1.1679	0.0040	5.8286	0.0057	8.16658
ECDSA	0.5971	0.6170	なし	1.2376	0.0012	2.45309

次に 5 個の BGPSEC ルータを経由してアップデートメッセージを交換する際のそれぞれのルータでの署名生成や検証に要する時間を簡単に導出した結果を表 4 にまとめている．アップデートメッセージが経由するルータの数は平均 5〜6 個程度あり，非常に多い場合でも 10 個にも達しない．このため，今回は 5 個の場合で速度を導出した．

署名方式の提案 (ECDSA) と提案 (Agg) は署名を事前に計算した際のそれぞれの署名方式のことである．また，R1 ∼ R5 は BGPSEC スピー

カを表しており，アップデートメッセージがR1からR2に伝播して最終的にR5に届くような状況を想定している．つまり，R1はORIGIN ASであり一番はじめの経路広告を生成するルータである．それぞれのルータにおける処理時間について言及する．全て処理は初期化と終了処理の時間が必要である．R1はこれらの処理時間に加え，署名処理時間を加えたものになる．R2以降では署名処理時間に加え，検証処理時間も必要となり，アップデートメッセージが1つのルータを経由する度に署名の数も増加するため検証処理時間は署名の数分だけ増加する．また，アグリゲート署名ではこれに加えて集約処理の時間も必要となる．さらに，提案方式では事前に署名処理を行っているため，署名処理時間はかからないものとしている．

また，BGPの運用方法によっては検証処理は基本的に時間がかかるため，先に署名を作成してこの署名を付与したアップデートメッセージを交換した後の空き時間に署名の検証を行う場合もある．このように運用した場合は，アップデートメッセージを生成する段階では検証処理を行わないため，より高速にアップデートメッセージを生成することが可能となる．

表 4: それぞれのBGPSECルータでの計算速度 (単位: m sec)

署名方式	R1	R2	R3	R4	R5
GDH	2.3268	13.4396	24.5524	35.6652	46.7780
Agg	2.3338	8.1665	13.9951	19.8237	25.6523
ECDSA	1.2154	2.4530	3.6906	4.9282	6.1658
提案 (ECDSA)	1.2154	1.8359	3.0735	4.3111	5.5487
提案 (Agg)	2.3338	6.9986	12.8272	18.6558	24.4844

実装したプログラムは公開しているため，詳細はこちらを参照されたい[1]．

次にBGPアップデートメッセージのサイズが提案手法により，どの程度削減するかの見積もり計算の結果を示す．以下の条件の基で計算を行った．ただし，署名サイズは実装プログラムの値を参考にしている．

1. Withdrawn Routes はなし
2. NLRI の値は 172.16.0.0/16
3. 署名1つのサイズは ECDSA とアグリゲート署名どちらも 64 byte (512 bit)
4. 提案手法のトレーシングは使用しない
5. Signature_Block は 1 つだけ
6. Origin AS を含めて 5 つの AS を経由した状態

また，BGPSECのデータフレームは大きく，ヘッダとアップデートメッセージ，この中のBGPSEC PATH属性から構成されており，以下の様なサイズの内訳になっている．

- BGP メッセージヘッダ (20 byte)
- UPDATE メッセージ
  - Unfeasible Routes Length (2 byte)
  - Withdrawn Routes (variable, 0 byte)
  - Total Path Attribute Length (2 byte)
  - BGPsec_Path Attribute (variable)
  - Network Layer Reachability Information, NLRI (variable)
    - Length (1 byte)
    - Prefix (variable)
- BGPsec_Path Attribute
  - Secure_Path (variable)
    - Secure_Path Length (2 byte)
    - Secure_Path Segments (variable, path の個数によって変動)
      - AS Number (4 byte)
      - pCount (1 byte)
      - Flags (1 byte)
  - Signature_Block (variable)
    - Signature_Block Length (2 byte)
    - Algorithm Suite Identifier (1 byte)
    - Sequence of Signature Segments (variable, path の個数によって変動)
      - Subject Key Identifier (20 byte)
      - Signature Length (2 byte)
      - Signature (64 byte, 実装プログラムの場合)

上記のデータフレームの内訳と条件にそいデータサイズを計算した結果が表5である．提案手法のアグリゲート署名を利用した方式の方がサイズが小さいことが確認できる．ECDSA署名を利用した一般的なBGPSECは経由するASが増加すればするほど署名の数が増え，結果，アップデートメッセージのサイズが増加することになる．また，一般化式はこのアップデートメッセージのサイズを一般化した式であり，nは経由するASの個数である．

表 5: アップデートメッセージサイズ

単位: byte	署名サイズ	全体サイズ	一般化式
BGPSEC	320	493	$92n + 33$
提案手法	64	237	$28n + 97$

7 まとめ

BGPの経路情報の正当性を保証するために，BGPSECのAS PATH属性の保証が抱えている問題に注目して，アグリゲート署名を適用した新たな手法について考察を行った．また，AS

[1]ECDSA) https://github.com/natsus/ecdsa
アグリゲート) https://github.com/natsus/agg_sig

PATH 属性を保証する際の処理効率を向上させるために署名の事前計算及び，アグリゲート署名を導入していても不正な署名を発見することが可能なトレーシング手法の 2 つを提案した．また，ECDSA 署名及びアグリゲート署名を実装し，速度評価とアップデートメッセージサイズの見積もりを行い，提案手法の効率性を簡易的に推察した．今後の課題として，より厳密な評価に向けて，The BIRD Internet Routing Daemon や ExaBGP などシミュレーションツールを使用して提案手法を実装，速度やメモリに関する評価を行う．また，シミュレーションにより，アグリゲート署名の導入により新たに発生する問題点も明らかにする．

謝辞　本研究の一部は公益財団法人 KDDI 財団調査研究助成「ペアリング暗号の高速化についての研究」，JSPS 科研費 26330151, 26880012，公益財団法人倉田記念日立科学技術財団倉田奨励金，JSPS A3 Foresight Program の助成により実施されている．

参考文献

[1] Akamai. Prolexic Routed. https://www.akamai.com/us/en/solutions/products/cloud-security/prolexic-routed.jsp.

[2] A. Boldyreva and R. Lychev. Provable security of S-BGP and other path vector protocols: model, analysis and extensions. In *ACM, CCS 2012*, pp. 541–552, 2012.

[3] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *EUROCRYPT 2003*, pp. 416–432, 2003.

[4] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In *ASIACRYPT 2001*, pp. 514–532, 2001.

[5] C. Gentry and Z. Ramzan. Identity-based aggregate signatures. In *PKC 2006*, pp. 257–273, 2006.

[6] P. Gill, M. Schapira, and S. Goldberg. Let the market drive deployment: a strategy for transitioning to BGP security. In *ACM SIGCOMM 2011*, pp. 14–25, 2011.

[7] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. D. McDaniel, and A. D. Rubin. Working around BGP: an incremental approach to improving security and accuracy in interdomain routing. In *NDSS 2003*, 2003.

[8] S. Hohenberger, V. Koppula, and B. Waters. Universal signature aggregators. In *EUROCRYPT 2015*, pp. 3–34, 2015.

[9] Y. Hu, A. Perrig, and M. A. Sirbu. SPV: secure path vector routing for securing BGP. In *ACM SIGCOMM 2004*, pp. 179–192, 2004.

[10] S. T. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (S-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, 2000.

[11] Laboratory of Cryptography and Information Security, University of Tsukuba. TEPLA, 2013. http://www.cipher.risk.tsukuba.ac.jp/tepla/.

[12] M. Lepinski. BPSEC Protocol Specification. draft-ietf-sidr-bgpsec-protocol (work in progress), 2015.

[13] M. Lepinski and S. Kent. An Infrastructure to Support Secure Internet Routing. RFC 6480, 2012.

[14] M. Lepinski, S. Kent, and D. Kong. A Profile for Route Origin Authorizations (ROAs). RFC 6482, 2012.

[15] Q. Li, Y.-C. Hu, and X. Zhang. Even rockets cannot make pigs fly sustainably: Can bgp be secured with bgpsec? In *NDSS 2014 Workshop on SENT*, 2014.

[16] P. Litke and D. S. C. T. U. Joe Stewart. BGP Hijacking for Cryptocurrency Profit, 2014. http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit/dt.

[17] R. Lychev, S. Goldberg, and M. Schapira. BGP security in partial deployment: is the juice worth the squeeze? In *ACM SIGCOMM 2013*, pp. 171–182, 2013.

[18] A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In *EUROCRYPT 2004*, pp. 74–90, 2004.

[19] RIPE. Youtube Hijacking: A Ripe NCC RIS Case Study, 2008. http://www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study.

[20] M. Schuchard, A. Mohaisen, D. F. Kune, N. Hopper, Y. Kim, and E. Y. Vasserman. Losing control of the internet: Using the data plane to attack the control plane. In *NDSS 2011*, 2011.

[21] K. Sriram, O. Borchert, O. Kim, D. Cooper, and D. Montgomery. RIB Size Estimation for BGPSEC, 2011. http://www.nist.gov/itl/antd/upload/BGPSEC_RIB_Estimation.pdf.

[22] P. Vervier, O. Thonnard, and M. Dacier. Mind your blocks: On the stealthiness of malicious BGP hijacks. In *NDSS 2015*, 2015.

[23] E. Y. Rekhter, E. T. Li, and E. S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271, 2006.

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

[25] M. Zhao, S. W. Smith, and D. M. Nicol. Aggregated path authentication for efficient BGP security. In *ACM, CCS 2005*, pp. 128–138, 2005.