情報処理学会論文誌 Vol.58 No.2 544–556 (Feb. 2017)

推薦論文

APAT：BGPSECにおけるアグリゲート署名の導入

田中 和磨[1,a)] 矢内 直人[2,b)] 岡田 雅之[3,c)] 西出 隆志[4,d)] 岡本 栄司[4]

受付日 2016年5月5日, 採録日 2016年11月1日

概要：BGPSECはBGPに電子署名を導入することで第三者による経路のハイジャックを防ぐ技術である．一方で，電子署名の導入により，ルータのメモリが肥大化するという問題が指摘されている．本稿ではこの問題に対し，アグリゲート署名と呼ばれる電子署名を圧縮する技術をBGPSECに導入することで効率性を実現するプロトコル，Aggregated Path Authentication with Tracing（APAT）を提案する．アグリゲート署名は署名が圧縮されるため，1個の不正な署名を挿入することで全体の情報が不受理になるという問題があるが，本稿ではこの問題を解決するトレーシング技術も新たに提案・導入している．また，アグリゲート署名の実装を通じた性能評価も行う．ECDSAを導入したBGPSECと比べて，APATはたかだか0.5ミリ秒の計算時間の遅延によりメモリサイズを従来の4割程度に削減することが可能である．

キーワード：BGP，BGPSEC，経路ハイジャッキング，アグリゲート署名，トレーシング

APAT: An Application of Aggregate Signatures to BGPSEC

Kazuma Tanaka[1,a)] Naoto Yanai[2,b)] Masayuki Okada[3,c)] Takashi Nishide[4,d)] Eiji Okamoto[4]

Received: May 5, 2016, Accepted: November 1, 2016

Abstract: BGPSEC is a protocol which prevents route hijacking by introducing digital signatures to BGP, but it is pointed out that there is a problem that memory size becomes bloated. In this paper, we propose a new protocol called *aggregated path authentication with tracing (APAT)* which introduces aggregate signatures where individually generated signatures can be combined into a single short signature. We mention that aggregate signatures have some inherent problem that verification fails if invalid signatures are inserted to aggregated signatures. That is, the original signatures are information-theoretically hidden and thus the aggregate signatures are forced to be invalid in the standpoint of a verifier. We also note that APAT overcomes this problem by a virtue of our proposed tracing method which can identify invalid signatures from aggregate signatures. Moreover, we implement aggregated signatures and evaluate their performances. Then, we show that APAT enables to reduce the memory size to one third with only five-millisecond overhead compared with the conventional BGPSEC with ECDSA.

Keywords: BGP, BGPSEC, route hijacking, aggregate signature, tracing

[1] 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305–8577, Japan
[2] 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565–0871, Japan
[3] 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
Japan Network Information Center, Chiyoda, Tokyo 101–0047, Japan
[4] 筑波大学大学院システム情報系
Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305–8577, Japan
a) tanaka@cipher.risk.tsukuba.ac.jp
b) yanai@ist.osaka-u.ac.jp
c) okadams@nic.ad.jp
d) nishide@risk.tsukuba.ac.jp

1. はじめに

1.1 背景

インターネットサービスプロバイダ（ISP）や企業，研究所などの組織は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

本稿の内容は2015年10月のコンピュータセキュリティシンポジウム2015で報告され，同プログラム委員長より情報処理学会論文誌ジャーナルへの掲載が推薦された論文である．

経路情報を交換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）[30] であるが，BGP で交換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，改ざんしたりする Man In The Middle（MITM）攻撃が可能である．具体的な事例としては，2008 年のアジア地域の国営通信事業者による YouTube の経路情報の乗っ取り [24] や Bitcoin の換金システムにおいて経路情報のハイジャックが発生して 1,000 万円近くの被害 [21] が発生している．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，必要なパケットのみを攻撃対象のネットワークへ受け渡すサービスが提供されている [2]．このような正当な経路広告は運用者の意思のもとに行われるものであり，現状の仕組みでは正当な経路広告と不正な経路広告の区別は難しい*1．このような理由からも経路情報の正当性の保証は重要である．

BGP で交換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，経路の発信元である ORIGIN AS の AS 番号とその AS が広告する IP アドレスの組合せに対して，正しい組合せであることを保証するものである．この ORIGIN AS 情報の保証については，Route Origin Authorization（ROA）[19] と呼ばれる AS 番号と IP アドレスの組合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る．国内では日本ネットワークインフォメーションセンターとインターネットマルチフィードによって，専用の公開鍵基盤である Resource Public Key Infrastructure（RPKI）[18] の登録情報の公開検索機能が提供されており，ROA の仕組が活用され始めている．

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ものである．その具体的な手法として，BGP に電子署名を導入することで経路の情報を保証する BGP Security Extension（BGPSEC）の標準化 [17] が Internet Engineering Task Force（IETF）によって行われている．しかしながら，現在使用されている Elliptic Curve Digital Signature Algorithm（ECDSA）署名 [14] 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．将来的には，暗号技術の導入によりルータのメモリ量は数 10G バイトも必要となり [27]，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．

*1 このような正当な動機による経路のハイジャックを “正当な経路のハイジャック” と呼ぶ．

このような背景から，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．本稿では低いコストで経路情報の正当性を保証可能な新たな AS PATH 属性保証機能を持ったプロトコルを提案する．

1.2 本稿の貢献

本稿では AS PATH 属性の保証に注目し，署名サイズが削減可能なアグリゲート署名 [4] を利用した方式を提案する．アグリゲート署名は独立に生成された電子署名を 1 個の署名に集約することで，任意の運用規模に対し署名サイズがつねに定数となることを目的とした暗号技術である．一見するとこの署名方式の導入により BGPSEC がかかえていた問題が解決されるように見えるが，本稿ではアグリゲート署名の導入に際して新たな 2 つの問題が生じることを指摘する．まず，“巻き込み署名不受理” 問題である．アグリゲート署名の検証において結果が不受理になったとき，すべての署名を不受理なものと扱わざるをえない．これはアグリゲート署名では個々の署名を集約しているため，各々の署名のうちどの署名が無効であるか特定することができないことに起因する．第 2 の問題はアグリゲート署名の実装評価である．アグリゲート署名はペアリングと呼ばれる楕円曲線上で定義される特殊な関数を用いるが，ペアリングは ECDSA など従来の電子署名技術と比べて処理負荷が高い．現時点での経路数は 50 万件以上といわれており，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．また，上記 2 つの問題とは独立に，アグリゲート署名に適した新たな BGPSEC の仕様の模索も必要となる．

これら一連の問題を改善する提案方式を，Aggregated Path Authentication with Tracing（APAT）と呼ぶ．APAT の特徴はアグリゲート署名による署名の集約と，検証結果が不受理の際に不受理な署名を発見するトレーシング機能をあわせ持つことである．この手法では不受理な署名が生じうる個数をあらかじめ指定する必要があるが，不受理な署名の数が指定した数以下の場合は任意の署名を特定することが可能である．これにより “巻き込み署名不受理” 問題を解決する．また，署名方式を実装し，ECDSA 署名およびアグリゲート署名の実行速度の比較に加えて，メモリサイズの見積りを導出する．この結果，アグリゲート署名を導入した BGPSEC では従来の ECDSA 署名を使用しているものよりも，たかだか 0.5 ミリ秒の計算時間の増加に対しメモリサイズを従来の 4 割程度まで削減することを示す．詳細は 5 章で述べる．

2. BGP Security Extension（BGPSEC）

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

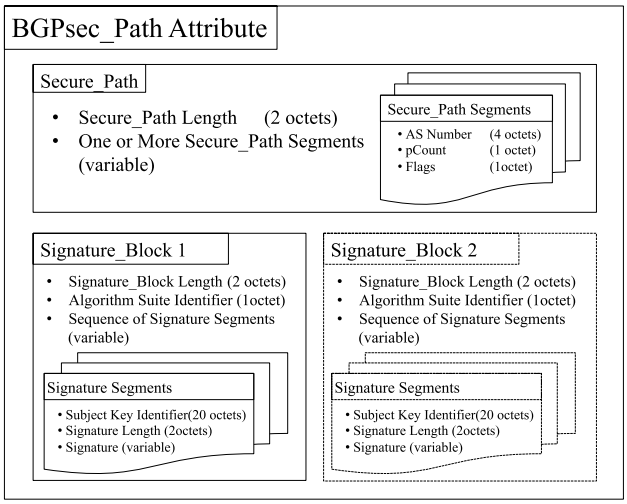

図 1 BGPSEC_PATH 属性の構成図
Fig. 1 Framework of BGPSEC_PATH attribute.

証に関しては，すでに提案されている 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．

2.1 BGPSEC_PATH 属性のフレームワーク

BGPSEC_PATH 属性は BGP アップデートメッセージが交換された AS PATH に関する情報を安全に伝達するものである．AS PATH の情報はディジタル署名によって保護されている．BGPSEC_PATH 属性のフレームワーク図を**図 1** に示す．

BGPSEC_PATH 属性は大きく Secure_Path と Signature_Blocks から構成される．Secure_Path は BGP アップデートメッセージに必要な AS PATH の情報を格納する属性であり，アップデートメッセージが通過した BGP ルータの AS 番号などの情報を含む Secure_Path Segment を格納している．また，Signature_Block は署名に関する属性であり，署名アルゴリズムの識別子に加えて，Secure_Path の中の AS 番号に対応した署名データ（Signature Segment）が格納されている．BGPSEC のフレームワークの詳細については，文献 [17] を参照されたい．

2.2 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 ルータが電子署名を付与することで実現されている．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．

表 1 ORIGIN AS が署名つけるデータ列
Table 1 Sequence of octets to be signed by ORIGIN AS.

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)

2.2.1 ORIGIN AS の署名生成

(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およ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) 署名および署名長を適当なデータフィールドに格納する．

表 1 のそれぞれのパラメータについて説明する．Target AS Number は次にアップデートメッセージを交換する AS の AS 番号を示す．Origin AS Number は経路情報を作成した AS の AS 番号，すなわち，この署名を生成している AS 自身の AS 番号を示す．pCount は同じ AS が AS PATH に複数回現れるかを表すパラメータである．Flags は複数の AS を同一と見なす Confed_Segment Flag のみが存在する．Algorithm Suite Id. は署名生成において使用するダイジェストアルゴリズムや署名アルゴリズムを特定するための識別子である．NLRI Length は経路交換するネットワークのネットワーク部の長さ，NLRI Prefix は経路交換するネットワークのネットワークアドレスに該当する値をそれぞれ意味する．

2.2.2 交換を行う AS の署名

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** であることに注意されたい．表 2 のそれぞれのパラメータについて説明する．Target AS Number は次にアップデートメッセージを交換する AS の AS 番号を，Signer's AS Number は署名を生成してアップデートメッセージにこの署名を追加する AS の AS 番号，つまり，この署名を生成している AS 自身の AS 番号をそれぞれ意味する．pCount と Flags は Origin AS の項で説明したものと同様の機能を有する．最後に，Most Recent Sig Field. は交換したアップデートメッセージに含まれて

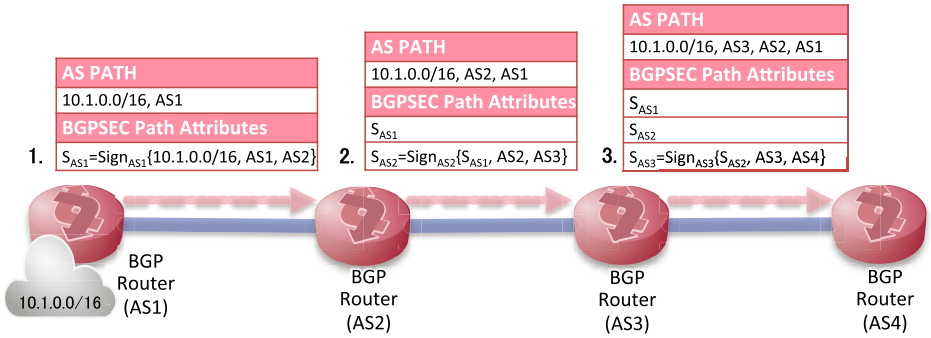

図 2 BGPSEC アップデートメッセージの交換
Fig. 2 Exchanging of a update message on BGPSEC.

表 2 交換を行う AS が署名つけるデータ列
Table 2 Sequence of octets to be signed by exchanging AS.

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

いる最も最近追加された署名の値となる．

2.2.3 検証

BGPSEC アップデートメッセージの検証概要について説明する．

(1) RPKI を通じて AS とその鍵の関係性を保証する RPKI ルータ証明書において，有効な証明書の中から有効な AS，Subject Key Identifier（SKI），公開鍵を見つけ出し，アップデートメッセージのそれぞれのセグメントのこれらの値に対応する属性と比較し一致するか確認する．もし一致する場合はここで得られた公開鍵で署名検証を行う．
(2) 与えられたアルゴリズムスイートから得られるダイジェスト関数を適切なデータ列（表 1，2）に適応して，ダイジェスト値を計算する．
(3) 与えられたアルゴリズムスイートから得られる署名検証アルゴリズムにダイジェスト値を適応して署名の検証を行う．
(4) (1)–(3) のステップをすべてのセグメントに対して行う．
(5) すべてのセグメントの検証結果が受理である場合は 'valid'，いずれかのセグメントの検証結果が不受理である場合は 'invalid' を出力する．

2.3 BGPSEC の未解決問題

図 2 は BGPSEC のアップデートメッセージがどのように交換されるかを表している．なお，Sign$_{ASn}\{A, B, \ldots\}$ は ASn が自身の秘密鍵を利用してデータ列 $A, B, \ldots$ に対して署名することを表しており，$AS1$ は ORIGIN AS である．BGPSEC の主な問題は 1.1 節で述べたとおり，ルータのメモリサイズである．その原因は，アップデートメッセージが経由した BGPSEC ルータ数が多ければ多いほど署名数（セグメント数）が増加することに起因する．より厳密には，これは署名サイズが増加することを意味しており，パケットサイズが増加することを意味する．BGPSEC ルータはその受信パケットに記載された BGPSEC_PATH 属性をルーティングテーブルとしてメモリに保存する必要があることから，結果としてメモリサイズが増加してしまう．また，検証処理の計算時間も，BGPSEC ルータ数に依存して長くなる．

3. BGPSEC へのアグリゲート署名の導入

本稿では 2.3 節で述べた問題解決に対し，アグリゲート署名の導入を検討する．アグリゲート署名の最大の利点は複数の署名を，署名の数にかかわらず，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．これは署名によるデータサイズを定数オーダとすることから，BGPSEC の問題を本質的に解決する可能性を秘めている．一方で，アグリゲート署名の導入は容易ではない．本章ではアグリゲート署名の詳細およびその導入に関する問題点を述べる．これらの問題点は 2.3 節における従来の問題点とは別の，本稿で取り組む技術的な課題といえる．

3.1 アグリゲート署名

3.1.1 アグリゲート署名の選定

アグリゲート署名の特徴は複数の署名を 1 つの署名に集約する処理にある．その構成には独立して生成された署名を第三者が任意の時点で集約できる汎用的アグリゲート署名 [4] と集約処理と署名処理を同時に行う順次的アグリゲート署名 [23] の 2 つがある．このうち，著者らは文献 [31] において順次的アグリゲート署名は集約処理を後でまとめて行う場合などは適用が難しいことを示している．このため，本稿では以降，汎用的アグリゲート署名に着目して議論を進める．

一方，汎用的アグリゲート署名は，その構成が多くは知

られていない [1], [4], [10], [11]．このうち，文献 [1] の方式は同じ時刻において生成された署名しか集約できないという性質を持つ．これはネットワークが動的に変化する状況を考えると，集約ができない可能性を含む．また，文献 [10], [11] の方式は多重線形写像や暗号学的難読化と呼ばれる最新の暗号理論から構成されている．その詳細は割愛するが，これらの要素技術は実用的な実装が知られていない．このため，現実的に利用可能な方式は初の方式である Boneh らの方式 [4] に限られる．次項で Boneh らのアグリゲート署名 [4] のアルゴリズムを示す．

3.1.2 Boneh らのアグリゲート署名方式

Boneh らのアグリゲート署名は 6 つのアルゴリズム：*Setup*（Algorithm 1），*Key Generation*（Algorithm 2），*Signing*（Algorithm 3），*Verification*（Algorithm 4），*Aggregation*（Algorithm 5），*Aggregate Verification*（Algorithm 6）から構成される．最初の 4 つのアルゴリズムは一般的な Gap Diffie-Hellman（GDH）署名方式 [5] であり，最後の 2 つが集約機能である．これらのアルゴリズムは Algorithms 1–6 に記載している．

Algorithm 1 Setup

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

Algorithm 4 Verification

Require: v, a message M, a signature σ
1: $h \leftarrow H(M)$
2: **if** $e(\sigma, g_2) = e(h, v)$ **then**
3: **return** *accept*
4: **else**
5: **return** *reject*
6: **end if**

Algorithm 5 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

なお，アグリゲート署名はその数学的な構造としてペアリングから構成されており，以下にその性質を説明する．$\mathbb{G}_1$ と $\mathbb{G}_2$ はそれぞれ素数位数 p の（加法）巡回群であり，乗法群 $\mathbb{G}_T$ は $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = p$ を満足する．ランダムオラクルのように振る舞うフルドメインハッシュ関数の一種 $H\colon \{0,1\}^* \to \mathbb{G}_1$ を map-to-points と呼ぶ．ペアリング $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ は Bilinearity と呼ばれる性質を持つ．これは任意の $T \in \mathbb{G}_1$，$S \in \mathbb{G}_2$ および $a, b \in \mathbb{Z}_p$ において，$e(aS, bT) = e(S, T)^{ab}$ が成立することを意味する．他にも非退化，準同型性などの性質があるが，これらは提案方式の機能には影響しないため説明は省略する．また，このようなペアリングは一般的に有限時間内に効率的に計算可能であることを仮定する．

3.2 問題点

BGPSEC へアグリゲート署名を導入することで BGPSEC がかかえているメモリ量やパケットサイズ増加の問題は解決することができる．しかしながら，単に BGPSEC にアグリゲート署名を適応するだけでは，効率性や信頼性などにおいて新たな問題が発生することが考えられる．以下に予想される問題について言及する．

3.2.1 大きな計算コスト

Algorithm 3 の署名生成において，署名者はメッセージを群要素へエンコードするためにハッシュ関数の一種の map-to-point 関数の計算 1 回と 1 回のスカラー倍算を行う必要がある．ECDSA では 1 回だけのスカラー倍算とよく使用される SHA-256 のようなハッシュ関数を計算するだけでよいため，アグリゲート署名の方が計算コストが大きくなると考えられる．これに加えて，Algorithms 4，6 の検証では演算コストが大きなペアリングを計算する必要がある．一方で，ECDSA と RSA に基づいた署名方式では複雑なペアリングの計算は必要ない．この演算コストは選択する楕円曲線に依存しているが，多くの場合は計算時間は増加することが予測される．

3.2.2 巻き込み署名不受理

もしアグリゲート署名に 1 つでも検証が不受理となる署名が含まれていた場合，残りのすべての署名が受理される

Algorithm 6 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: **else**
5: **return** *reject*
6: **end if**

ものだとしてもアグリゲート署名全体が不受理であると判定され，署名は拒否される．この問題はアグリゲート署名の署名集約機能が仇となり発生する．個々の署名が集約された状態では，もとの各々の署名は情報理論的に隠れてしまい，各々の署名の秘密鍵を利用しない限り，アグリゲート署名からもとの署名を抽出することはできない．それゆえに，検証者においてはアグリゲート署名全体を不受理として扱わなければならない．n を署名者数としたとき，このアグリゲート署名からオリジナルの個々の署名を復元することの難しさは n-抽出問題 [6] として知られており，この問題は CDH 問題と同等の難しさである．このような不受理な署名の存在により引き起こされる問題を巻き込み署名不受理問題と呼ぶ．つまり，BGPSEC の AS PATH 属性の保証において，AS PATH 属性に 1 つでも不受理な署名を持つ AS が含まれていた場合はどの AS が正当であり，どの AS が不正であるか特定することができないことを意味している．これは運用上，重要な問題となる．

3.2.3 仕様の欠如と評価

アグリゲート署名は最近作られた暗号方式であるため，実装や応用などは依然として少ない現状である．また，アグリゲート署名のアルゴリズムは ECDSA など従来の署名方式のアルゴリズムとは異なり，署名集約アルゴリズムが含まれている．このため，BGPSEC の署名方式をアグリゲート署名に変更するためには新たな BGPSEC の仕様を考える必要がある．これに加えて，署名アルゴリズムが大きく変化するため，BGPSEC がかかえているメモリサイズ増加などの問題に関する評価を正確に行う必要がある．

4. Aggregated Path Authentication with Tracing（APAT）

Aggregated Path Authentication with Tracing（APAT）は，AS PATH 属性の正当性を保証するために本稿で提案する新たな手法である．APAT はアグリゲート署名を利用した経路情報に占める署名サイズの大幅な削減と不受理な署名を特定するトレーシングという 2 つの特徴を持つ．基本的なプロトコルは BGPSEC [17] に準拠して構成されるため，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．BGPSEC において，アップデートメッセージには経由したルータ数の署名が追加される．このため，アップデートメッセージが 5 個のルータを経由したときの署名数は 5 つになるが，アグリゲート署名を利用している場合は 1 つの署名だけですみ，署名サイズについてもおおよそ 1/5 になる．詳細な評価は次章に述べる．本方式では，署名サイズと個数が経由したルータ数に依存せず一定であることが大きな特長である．トレーシングについて以下で詳しく言及する．

4.1 トレーシング

トレーシング手法はアグリゲート署名の検証において検証結果が不受理である場合，アグリゲート署名の集約された署名に含まれている署名のうち，どの署名が不受理であるかを特定する手法である．検証結果が不受理となる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，不受理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，不受理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署名者による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，その解が不受理な署名となる．一方，この手法では不受理な署名の特定数があらかじめ制限されることとなる．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，より多くの不受理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不受理な署名の発生頻度と運用負荷の見積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 6 つのアルゴリズムから成り立つ：*Setup*（Algorithm 7），*Key Generation*（Algorithm 2），*Signing*（Algorithm 3），*Aggregation*（Algorithm 8），*Verification*（Algorithm 9），*Tracing*（Algorithm 10）．ただし，*Key Generation* と *Signing* は 3 章で説明した Boneh らのアグリゲート署名と同じものである．

具体例

無効な署名を見つける場合の例を示す．提案手法では *Setup*，*Key Generation*，*Signing* の処理は従来のアグリゲート署名と同様であるが，*Aggregation* の処理が以下のようになる．まず乱数 R を生成し，その後に n 個のハッシュ値 $\delta_j = H_2(pk_j \parallel R)$ を計算する．その後に以下を計算する．

Algorithm 7 Setup (Tracing Ver.)

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

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

Algorithm 8 Aggregation (Tracing Ver.)

Require: $para$, i-tuples $(\{pk_j, m_j, \sigma_j\}_{j=1}^{i})$, a secret key sk_{agg} of an aggregator
1: $R \xleftarrow{R} \{0,1\}^k$ // generate a random number
2: $\sigma_{agg} \leftarrow$ **Signing**$(para, sk_{agg}, R)$
3: **for** $j = 1$ to i **do** $\delta_j = H_2(pk_j \parallel R)$ **end for**
4: **for** $k = 0$ to τ **do** $\rho_k = \sum_{j=1}^{i} \delta_j^k \sigma_j$ **end for**
5: **return** $\sigma = (\sigma_{agg}, R, \rho_0, \cdots, \rho_\tau)$

Algorithm 9 Verification (Tracing Ver.)

Require: $para, \{pk_j, m_j\}_{j=1}^{i}, \sigma = (\sigma_{agg}, R, \rho_0, \cdots, \rho_\tau)$
Ensure: the messages m_j are all distinct, and *reject* otherwise.
1: **for** $j = 1$ to $k = i$ **do** $h_j \leftarrow H_1(m_j)$ **end for**
2: **if** $e(\rho_0, g_2) = \prod_{j=1}^{i} e(h_j, pk_j)$ **then**
3: **return** *accept*
4: **else**
5: **return** *reject*
6: **end if**

Algorithm 10 Tracing (Tracing Ver.)

Require: $para, \{pk_j, m_j\}_{j=1}^{i}, \sigma = (\sigma_{agg}, R, \rho_0, \cdots, \rho_\tau)$
Ensure: the p_k is the kth polynomial of signatures.
1: **if Verification**$(para, pk_{agg}, R, \sigma_{agg}) = reject$ or **Verification**$(para, \{pk_j, m_j\}_{j=1}^{i}, \rho_0) = accept$ **then**
2: **return** $\emptyset$
3: **else**
4: **for** $k = 0$ to τ **do**
5: $\alpha_k = \frac{\boldsymbol{e}(\rho_k, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j^k \cdot pk_j)}$ where $h_j = H_1(m_j)$
6: **end for**
7: **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**
8: **return** a set $\mathcal{I} \subset [1, i]$ of indexes corresponding to δ_j's
9: **else**
10: **return** $\emptyset$
11: **end if**
12: **end if**

$$\rho_0 = \sum_{j=1}^{i} \sigma_j, \qquad \rho_1 = \sum_{j=1}^{i} \delta_j \sigma_j.$$

アグリゲート署名として (ρ_0, ρ_1, R) を出力する．*Aggregate Verification Algorithm* に実質使う署名はこのうち ρ_0 だけであり，もし ρ_0 を用いて検証式 $\boldsymbol{e}(\rho_0, g_2) = \prod_{j=1}^{i} \boldsymbol{e}(h_j, y_j)$ が成り立つならば署名は受理される．検証結果が不受理の場合，*Tracing Algorithm* で以下が成り立つような乱数 δ_k を $[\delta_1, \delta_i]$ から探索する．

$$\frac{\boldsymbol{e}(\rho_1, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j y_j)} \stackrel{?}{=} \left(\frac{\boldsymbol{e}(\rho_0, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, pk_j)} \right)^{\delta_k}$$

上式において，不受理状態にある署名だけが割り切れずに，両辺それぞれに残る．右辺の要素 ρ_1 はそれぞれのユーザに対し個別の乱数を含むことから，左辺においてこの式が成り立つような乱数 δ_k を探すことで不受理な署名を特定できる．なお，この例で見つけられる不受理な署名は最大1個であり，2つ以上の署名を発見したい場合は集約処理が変わる．具体的には $\rho_2 = \sum_{j=1}^{i} \delta_j^2 \sigma_j$ となるアグリゲート署名を新たに出力する．このとき，トレーシングの処理は以下の等式を満たす2つの乱数 δ_{k1}，δ_{k2} を発見することとなる．

$$\left(\frac{\boldsymbol{e}(\rho_2, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j^2 pk_j)} \right) \stackrel{?}{=} \left(\frac{\boldsymbol{e}(\rho_1, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j y_j)} \right)^{\delta_{k1}+\delta_{k2}} \left(\frac{\boldsymbol{e}(\rho_0, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, y_j)} \right)^{-\delta_{k1}\delta_{k2}}$$

τ 個の無効な署名を特定する場合は *Aggregation* Algorithm で $\tau+1$ 個のアグリゲート署名を生成する．

$$\rho_0 = \sum_{j=1}^{i} \sigma_j, \qquad \rho_1 = \sum_{j=1}^{i} \delta_j \sigma_j,$$
$$\rho_2 = \sum_{j=1}^{i} \delta_j^2 \sigma_j, \qquad \cdots \qquad , \qquad \rho_\tau = \sum_{j=1}^{i} \delta_j^\tau \sigma_j.$$

そして，$\rho_0, \cdots, \rho_\tau$ の $\tau+1$ 個の変数からなる以下の等式を満足する乱数 δ の組合せを発見すればよい．

$$\left(\frac{\boldsymbol{e}(\rho_\tau, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j^\tau pk_j)} \right) \stackrel{?}{=} \left(\frac{\boldsymbol{e}(\rho_{\tau-1}, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j^{\tau-1} y_j)} \right)^{\alpha_1} \left(\frac{\boldsymbol{e}(\rho_{\tau-2}, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j^{\tau-2} y_j)} \right)^{-\alpha_2} \cdots \left(\frac{\boldsymbol{e}(\rho_1, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, \delta_j y_j)} \right)^{-\alpha_{\tau-1}} \left(\frac{\boldsymbol{e}(\rho_0, g_2)}{\prod_{j=1}^{i} \boldsymbol{e}(h_j, y_j)} \right)^{-\alpha_\tau}$$

ただし，$\alpha_1, \alpha_2, \cdots, \alpha_\tau$ はそれぞれ1次の τ 変数基本対称式，2次の τ 変数基本対称式，$\cdots$ τ 次の τ 変数基本対称式である．4変数の場合のそれぞれ基本対称式の具体的な値を以下に示す．

$$\alpha_1 = \delta_{k1} + \delta_{k2} + \delta_{k3} + \delta_{k4},$$
$$\alpha_2 = \delta_{k1}\delta_{k2} + \delta_{k1}\delta_{k3} + \delta_{k1}\delta_{k4} + \delta_{k2}\delta_{k3} + \delta_{k2}\delta_{k4} + \delta_{k3}\delta_{k4},$$
$$\alpha_3 = \delta_{k1}\delta_{k2}\delta_{k3} + \delta_{k1}\delta_{k2}\delta_{k4} + \delta_{k1}\delta_{k3}\delta_{k4} + \delta_{k2}\delta_{k3}\delta_{k4},$$
$$\alpha_4 = \delta_{k1}\delta_{k2}\delta_{k3}\delta_{k4}.$$

4.2 具体的な構成方法

3.1.1 項で述べたとおり，以下の構成では Boneh らのアグリゲート署名 [4] 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減が可能となる．厳密には，ORIGIN AS の署名生成では Algorithm 3 の処理のみを実施し，ダイジェスト値および署名データを得る．また，2.2.2 項で述べた交換を行う AS の場合では，Algorithm 3 を行い署名を生成した後，Algorithm 8 を通じて署名集約を行う．この際に生成されるアグリゲート署名の個数はあらかじめネットワーク運用者により決められている個数に従うものとする．なお，

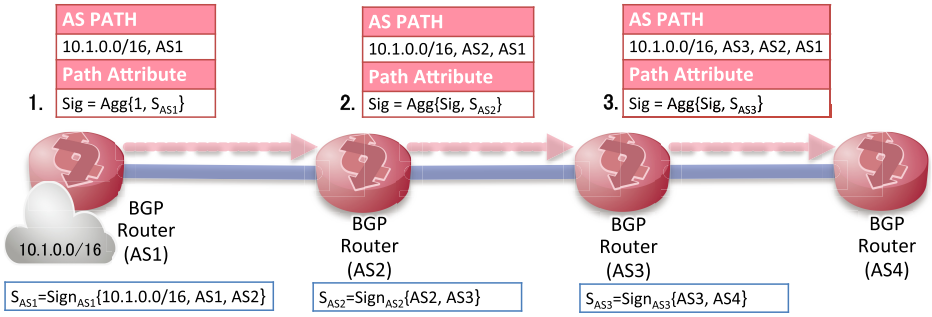

図 3 APAT におけるアップデートメッセージの交換
Fig. 3 Exchanging of a update message on APAT.

表 3 平文 1 個あたりの計算時間の評価（単位：ms）
Table 3 Computational time per message (Unit: ms).

Sig Scheme	初期化 [ms]	署名 [ms]	集約 [ms]	検証 [ms]	終了 [ms]	全体 [ms]
GDH Sig. [5]	1.8531	1.0834	none	11.0470	0.0054	13.9890
ECDSA Sig. [14]	0.6571	0.6298	none	1.2044	0.0020	2.4933
APAT	2.6806	1.1369	0.6283	6.0223	0.0051	10.4734

すべての BGP ルータでは Algorithm 7 の 2 を通じて鍵情報およびパラメータを登録しているものとする．検証処理では Algorithm 9 を用いて署名検証を行う．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の仕様に関しては BGPSEC [17] に準拠するものとする．APAT において，アップデートメッセージがどのように交換されるかを図 3 に示す．ここで，$\mathrm{Sign}_{ASn}\{A, B, \ldots\}$ は ASn が自身の秘密鍵を利用してデータ列 $A, B, \ldots$ に対して署名することを表しており，$AS1$ は ORIGIN AS，Agg は集約関数をそれぞれ意味する．

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．この場合は任意のタイミングで Algorithm 3 を行い署名を生成し，署名集約の処理が必要になった時点で Algorithm 8 を実行する．このような集約タイミングの分割が行える点も APAT の特徴の 1 つである．

5. 性能評価

本章では APAT の性能について，計算時間およびデータサイズの 2 つの観点から評価する．特にデータサイズの評価では，まずアップデートメッセージのデータサイズを評価し，その結果から APAT によりメモリサイズがどの程度削減できるか議論する．

5.1 実験環境

本稿では，ペアリングライブラリの TEPLA [16] 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 ビット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（APAT）との相対的な比較となりうる．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．以下に測定環境および測定結果を記載する．

OS OS X 10.10.1
CPU Intel Core i7-3520M CPU（2.90 GHz）
メモリ 8 GB
ペアリングライブラリ TEPLA 1.0，GMP 6.0.0a

実装したプログラムは公開しているため，詳細は実装コード[*2]を参照されたい．

5.2 計算時間の評価

表 3 に平文 1 個あたりの計算時間をまとめている．平文のサイズを百文字分（100 バイト）に固定して 100 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.，ECDSA Sig.，APAT はそれぞれ Gap Diffie-Hellman（GDH）署名 [5]，Elliptic Curve Digital Signature Algo-

[*2] ECDSA, https://github.com/natsus/ecdsa
アグリゲート署名，https://github.com/natsus/agg_sig

表 4 それぞれの BGPSEC ルータで要する計算時間の評価（単位：ms）
Table 4 Computational time for each BGPSEC router (Unit: ms).

Sig Scheme	R1 [ms]	R2 [ms]	R3 [ms]	R4 [ms]	R5 [ms]
GDH Sig. [5]	2.9420	13.9890	25.0360	36.0830	47.1300
ECDSA Sig. [14]	1.28895	2.4933	3.6977	4.9021	6.1064
APAT（通常処理）	4.5489	10.2993	16.0498	21.8003	27.5507
APAT（トレーシング）	-	29.1368	35.1592	41.1816	47.2039

rithm（ECDSA）署名 [14]，提案方式のことであり，“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” 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．

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．R1, R2, ..., R5 は BGPSEC ルータを表しており，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 に届く状況を想定している．つまり，R1 は ORIGIN AS であり一番最初に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計算時間について述べる．すべての BGPSEC ルータは引数の初期化と終了処理が必要である．R1 はこれらの計算時間に加え，署名生成の計算時間を加えたものになる．R2 以降では署名生成の計算時間に加え，署名検証の計算時間も必要となる．アップデートメッセージが 1 つのルータを経由するたびに署名の数も増加するため，署名検証の計算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．これらの計算結果を表 4 において APAT（通常処理）として記載する．さらに，R2 以降では署名の検証が不受理だった場合を想定し，トレーシングの計算時間も測定する．その理由はトレーシングは検証が失敗した場合の例外処理に相当するため，これは上述した計算時間の測定とは独立に行う．BGPSEC では電子署名の検証に失敗した場合，“失敗” という事実を後続の BGPSEC ルータに伝達すること，また，トレーシングは次のアップデートメッセージの到着を待つ待機時間に計算できることから，この計算時間の測定の分割は現実の状況とかい離しないものと考えている．なお，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 個の場合で計算時間を評価した．なお，検証が不受理となる署名の数は最大 1 つだけとして測定を行う．その理由は，検証が不受理となる署名の発生頻度は一般的にはきわめて低いためである．また，R1 で検証が不受理だった場合，R1 の署名が不受理となることが一意に定まるため，トレーシングの時間を測定しない．このため，各 R2 から R5 において，最大 1 個の署名を見つける状況で，考えられるすべての計算を行った “最悪の状況” に要する計算時間を測定する．トレーシングの計算時間については APAT（トレーシング）として記載する．

表 3，表 4 によると，APAT は検証時間が ECDSA に比べて劣るため，全体の計算時間としては ECDSA を用いた構成と比較しておおむね 4 倍程度の時間を必要とする．この原因は負荷が重いペアリングの計算を要することである．しかしながら，この遅延は以下の理由から問題にならないといえる．まず，BGP における経路情報伝達の間隔は初期値として 30 秒が設定されている [30]．このため，各ルータにおいて計算が 30 秒以内に完了すれば，現実的に動作するといえる．文献 [28] によるとペアリングは，現在のルータにおける標準的な仕様である 1 GHz 程度の計算能力を持つ ARM アーキテクチャにおいても，1 回あたり 1 ミリ秒程度で計算可能である．これはルータのアーキテクチャを想定した場合，表 3，表 4 の数値を約 3 倍から 4 倍に見積もればよいことを意味しており，その結果として APAT は実用的に動作可能といえる．また，ORIGIN AS（R1 に相当）においては，初期化処理を事前に実施しておくことで，ECDSA と比較して，最大で署名計算時間分の遅延である 0.5 ミリ秒程度まで速度を改善できる．このような実用的な計算時間のもとで，次節で説明するデータサイズの大幅な改善を得ることができる．

トレーシングの計算時間についても同様に，有用性が確認できる．表 4 の値はワーストケースであるが，BGPSEC ルータの台数が 1 台増加するごとに，計算時間は 6 ミリ秒増加する傾向が得られた．これはルータのアーキテクチャに換算すると 1 台あたりおおむね 20 ミリ秒程度の増加になる．また，5 台程度のルータにおいても 180 ミリ秒程度で動作する．また，前述のとおり，トレーシングは BGPSEC ルータが待機中に実行できる．これにより，トレーシングも現実的な計算時間で動作可能と見なせる．

5.3 データサイズの評価

5.3.1 アップデートメッセージの評価

BGP アップデートメッセージのサイズが APAT により，どの程度削減するかの見積り計算の結果を示す．以下の条

件のもとで計算を行った．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 個のサイズは ECDSA とアグリゲート署名どちらも 64 バイト（512 ビット）
(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，以下のような内訳になっている．

- BGP Message Header (20 byte)
- UPDATE Message
  - 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 by a path number)
      - 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 by a path number)
      - Subject Key Identifier (20 byte)
      - Signature Length (2 byte)
      - Signature (64 byte in our program)

上記のデータフレームの内訳と条件に沿い，アップデートメッセージのデータサイズを計算した結果が**表 5** である．前述したデータフレームのうち BGP ルータの数に対し線形に増加するものは Secure_Path Segments および Signature_Block である．正確にはアグリゲート署名を用いた場合，Signature_Block 内の Signature は BGPSEC ルータに対し定数サイズとなり，ECDSA や GDH 署名を用いた場合は線形サイズとなる．それ以外の項目は署名方式に限らず線形となる．このため，表 5 の数値が得られる．なお，表 5 において Sig size はアップデートメッセージのうち R1 から R5 までの署名に関するデータサイズ，Total size は R1 から R5 までのアップデートメッセージ自体のデータサイズ，および general size は任意の BGPSEC ルータ数 n に対するデータサイズの評価式である．なお，比較用の参考数値として従来の BGP におけるアップデートメッセージのデータサイズも表 5 に記載する．BGP のアップデートメッセージの基本構造は BGPSEC と同じであるが，BGPSEC の BGPsec_Path 属性に代わり，PATH 属性が導入される．この最少となる構成は経路の始点となる Origin（4 byte），AS PATH（variable by a path number），次のホップ先を表す Next Hop（4 byte）からなる．このうち，Origin と Next Hop に関する情報は固定長 8 byte であること，各 AS の情報は 4 byte で表現される．このため，ヘッダなど固定長の部分 25 byte を加えると，表 5 のようになる．特に，従来の BGP においても，AS PATH を取り扱うというその性質から，アップデートメッセージのデータサイズが線形になることに注意されたい．

表 5 アップデートメッセージのデータサイズ
Table 5 Data size of update message.

(UNIT: byte)	Sig size	Total size	general size
BGP	none	54	$4n + 33$
BGPSEC	320	493	$92n + 33$
APAT	64	237	$28n + 97$

Total size をみると，アグリゲート署名を利用した APAT のアップデートメッセージのサイズが従来の BGPSEC の半分程度になることが確認できる．また，general size においても n の係数のうち Signature の増加分を定数に削減できており，10 個程度の AS を想定した場合，データサイズは従来の 4 割程度まで削減可能である．これらの点から，APAT は有意義といえる．APAT の利点は，従来の ECDSA 署名を利用した一般的な BGPSEC では経由する AS が増加すればするほど署名の数が増加する点に比べ，アップデートメッセージのサイズが一定となることである．従来の BGPSEC の問題は，アップデートメッセージのデータサイズのうち線形増加部分（n の係数に相当）が BGP と比べて非常に大きいことに起因していた．APAT では署名のデータサイズが一定になることで，BGP 従来のアップデートメッセージのものに大きく近づけることが可能となった．これはメモリサイズの削減に関して，大きく寄与すると考えている．その詳細は次項に記載する．

5.3.2 ルータのメモリサイズの評価

まず，ルータのメモリサイズに大きく影響するものは，アップデートメッセージのデータサイズにおいて n の項である．これは実際のルーティングテーブルの情報が BGPsec_Path 属性に起因するためである．従来の BGP と比較した場合，APAT は n の係数が 7 倍程度にとどまっている．これはたとえば，BGP として現在の標準的なルータ

のメモリサイズである 256M バイトを最大限使った場合，APAT では約 2G バイトのメモリで運用できることを意味する．同様の環境を従来の BGPSEC で行う場合，約 6G バイトのメモリを必要とすることから，その差は無視できないものと考えている．2G バイトのメモリも 2016 年現在ではきわめて高水準なルータモデル[*3]であるが，現実に存在するルータを用いての運用が期待できることも事実である．将来的に BGPSEC でメモリが 30G バイト必要となる状況が発生した場合であっても，APAT では 9.1G バイト程度で運用が可能であり，有効性が期待できる．

最後に，本稿における未解決問題として，APAT のさらなる効率化方法について述べる．APAT のデータサイズにおける線形増加部分に関して，大部分を占めているものは公開鍵鍵識別子である Subject Key Identifier（20 byte）である．これは署名がどの AS が所有する鍵で生成されたかを表すものであることから，取り除くことが難しい．すなわち，従来の公開鍵型電子署名の枠組みを用いる場合，これ以上の効率化は容易には望めない．これを改善する手法として，著者らは ID ベース署名 [26] の利用を提案する．ID ベース署名とは公開鍵として任意の文字列を扱える電子署名であり，たとえば AS 番号を公開鍵として使うことも可能である．この場合，Signature_Block の Subject Key Identifier が不要になるため，理論上は n の係数を 8 まで削減することが可能となる．また，ID ベース署名のアグリゲート署名も知られている [7]．一方で，ID ベース署名は従来の公開鍵型電子署名と比べて仕様や管理基盤が備わっていない．このため，ID ベース署名導入の実現には，ID ベース署名自体の基盤を整える必要がある．

6. 関連研究

経路のハイジャックは 1 日あたり 5 件弱（Vervier ら [29]），日本国内では 1 カ月に 10 件前後（中野 [34]）も発生している．これに加えて，2015 年には Jacquemart ら [13] はブラックスペースと呼ばれる IP アドレスが割り当てられていない IPv4 アドレススペースを広告している BGP ルータおよびこれを行っている人に注目して，その特定方法および広告目的を調査している．これらの多くは IP ブラックリストの回避によるスパム送信や冒頭で述べたような正当な経路のハイジャックによるものであった．このような背景から経路のハイジャックへ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 Secure Border Gateway Protocol（S-BGP）[15] 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 [9] が，2004 年に Hu ら [12] 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，本稿ではこれらの問題に焦点を当てて調査した．本稿と類似した研究として 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 [33] を提案している．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前述した技術的な問題点が評価されていないため，本稿の成果とは大きく異なる．特に，メモリサイズに関する考察は非常に重要であるにもかかわらず言及されていないため，これもまた明らかにする必要があった．したがって，本稿ではこれらの問題についても調査した．

BGP で DoS 攻撃をする手法についての考察を 2011 年に Schuchard ら [25] が，安全な BGP の導入戦略に関する考察を 2011 年に Gill ら [8] が行っている．さらには安全な 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 [22] が議論している．本稿の主眼はプロトコルの設計であるため同様の検討は行っていないが，標準化など提案プロトコルの導入および普及を進める際はこれらの調査も不可欠である．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 [3] が行い，署名が偽造不可能かつ各 AS の認証が適切に行われていれば，BGPSEC が安全であることを明らかにした．この結果は提案方式における経路情報の偽造不可能性も保証している．また，既存の BGP の安全性要件の調査を 2014 年に Li ら [20] が改めて行っている．Li らによると，DoS 攻撃など直接的な経路のハイジャック以外の攻撃の防御は既存の仕様では難しく，新たな対策が必要となる．このような調査は当分野の研究開発を進めるうえで，将来的な課題となるだろう．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，Message Authentication Code（MAC）を利用して AS PAHT 属性の保証を行う手法を 2014 年に Zhang ら [32] が提案している．Zhang らの研究は形式検証を通じて経路の安全性を保証する手法であり，同様の手法を用いた安全性解析は文献 [3] の結果とも異なる安全性の根拠を与えうる．特に形式手法は安全性の検証を自動化し第三者的な確認を可能にすることから，同様の手法に関する議論は今後の研究課題の 1 つと考えている．

BGP セキュリティを取り巻く環境としては，BGPSEC などを運用するうえで欠かすことができない公開鍵基盤である Resource Public Key Infrastructure（RPKI）[18] の環境が活用されはじめ，経路情報中の ORIGIN AS 情報を保証するときに使用される認証データである Route Origin Authorization（ROA）[19] を発行する仕組みも活用されつつある．これらの技術を導入した評価実験も，提案プロトコルの普及を進めるうえで欠かせない課題である．今後の調査ではこれらの技術を導入した実験にも着手する．

[*3] Cisco 1000 シリーズ Connected Grid などが該当する．
http://www.cisco.com/web/JP/product/hs/routers/cgr1000/prodlit/datasheet_c78-696278.html

7. 結論

BGP の経路情報の正当性を保証するために，BGPSEC の AS PATH 属性の保証がかかえているメモリサイズ増加の問題に注目し，電子署名を集約可能なアグリゲート署名の導入について検討した．これによりアグリゲート署名を導入することで巻き込み署名不受理問題が生じることを明らかにし，また，その対策として不受理な署名を発見可能なトレーシング手法を提案した．これら一連の機能を導入した提案プロトコルが APAT である．また，ECDSA 署名およびアグリゲート署名を実装し，計算時間とアップデートメッセージサイズの見積りを行い，APAT の効率性を従来の手法と比較評価した．この結果，APAT は ECDSA 署名を使用した従来の BGPSEC と比較して，BGPSEC ルータの数が 10 台以上のときにメモリサイズを従来の 4 割程度まで削減できることを示した．今後のさらなる課題は，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 APAT を実装し，シミュレータ上で評価を行うことである．

謝辞 本研究の一部は JSPS 科研費 26880012, 26330151, JSPS A3 Foresight Program の助成を受けたものです．

参考文献

[1] Ahn, J.H., Green, M. and Hohenberger, S.: Synchronized Aggregate Signatures: New Definitions, Constructions and Applications, *CCS 2011*, pp.473–484, ACM (2010).
[2] Akamai: Prolexic Routed, available from ⟨https://www.akamai.com/us/en/solutions/products/cloud-security/prolexic-routed.jsp⟩.
[3] Boldyreva, A. and Lychev, R.: Provable security of S-BGP and other path vector protocols: Model, analysis and extensions, *CCS 2012*, pp.541–552, ACM (2012).
[4] Boneh, D., Gentry, C., Lynn, B. and Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps, *EUROCRYPT 2003*, LNCS, Vol.2656, pp.416–432, Springer-Verlag (2003).
[5] Boneh, D., Lynn, B. and Shacham, H.: Short Signatures from the Weil Pairing, *ASIACRYPT 2001*, LNCS, Vol.2248, pp.514–532, Springer-Verlag (2001).
[6] Coron, J. and Naccache, D.: Boneh et al.'s k-Element Aggregate Extraction Assumption Is Equivalent to the Diffie-Hellman Assumption, *ASIACRYPT 2003*, LNCS, Vol.2894, pp.392–397, Springer-Verlag (2003).
[7] Gentry, C. and Ramzan, Z.: Identity-Based Aggregate Signatures, *PKC 2006*, LNCS, Vol.3958, pp.257–273, Springer-Verlag (2006).
[8] Gill, P., Schapira, M. and Goldberg, S.: Let the market drive deployment: A strategy for transitioning to BGP security, *ACM SIGCOMM 2011*, pp.14–25, ACM (2011).
[9] Goodell, G., Aiello, W., Griffin, T., Ioannidis, J., McDaniel, P.D. and Rubin, A.D.: Working around BGP: An Incremental Approach to Improving Security and Accuracy in Interdomain Routing, *NDSS 2003*, Internet Society (2003).
[10] Hohenberger, S., Koppula, V. and Waters, B.: Universal Signature Aggregators, *Proc. EUROCRYPT 2015*, LNCS, Vol.9057, pp.3–34, Springer-Verlag (2015).
[11] Hohenberger, S., Sahai, A. and Waters, B.: Full Domain Hash from (Leveled) Multilinear Maps and Identity-Based Aggregate Signatures, *CRYPTO 2013*, LNCS, Vol.8042, pp.494–512, Springer-Verlag (2013).
[12] Hu, Y., Perrig, A. and Sirbu, M.A.: SPV: Secure path vector routing for securing BGP, *ACM SIGCOMM 2004*, pp.179–192, ACM (2004).
[13] Jacquemart, Q., Vervier, P., Urvoy-Keller, G. and Biersack, E.W.: Demystifying the IP Blackspace, *RAID 2015*, LNCS, Vol.9404, pp.111–132, Springer-Verlag (2015).
[14] Johnson, D., Menezes, A. and Vanstone, S.: The elliptic curve digital signature algorithm (ECDSA), *International Journal of Information Security*, Vol.1, No.1, pp.36–63 (2001).
[15] Kent, S.T., Lynn, C. and Seo, K.: Secure Border Gateway Protocol (S-BGP), *IEEE Journal on Selected Areas in Communications*, Vol.18, No.4, pp.582–592 (2000).
[16] Laboratory of Cryptography and Information Security, University of Tsukuba: TEPLA (2013), available from ⟨http://www.cipher.risk.tsukuba.ac.jp/tepla/⟩.
[17] Lepinski, M.: BGPSEC Protocol Specification, draft-ietf-sidr-bgpsec-protocol (work in progress) (2015).
[18] Lepinski, M. and Kent, S.: An Infrastructure to Support Secure Internet Routing, RFC 6480 (2012).
[19] Lepinski, M., Kent, S. and Kong, D.: A Profile for Route Origin Authorizations (ROAs), RFC 6482 (2012).
[20] Li, Q., Hu, Y.-C. and Zhang, X.: Even rockets cannot make pigs fly sustainably: Can BGP be secured with BGPsec?, *SENT 2014* (2014).
[21] Litke, P., Joe Stewart, J. and Dell SecureWorks Counter Threat Unit: BGP Hijacking for Cryptocurrency Profit (2014), available from ⟨http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit/⟩.
[22] Lychev, R., Goldberg, S. and Schapira, M.: BGP security in partial deployment: Is the juice worth the squeeze?, *ACM SIGCOMM 2013*, pp.171–182 (2013).
[23] Lysyanskaya, A., Micali, S., Reyzin, L. and Shacham, H.: Sequential Aggregate Signatures from Trapdoor Permutations, *EUROCRYPT 2004*, LNCS, Vol.3027, pp.74–90, Springer-Verlag (2004).
[24] RIPE: Youtube Hijacking: A Ripe NCC RIS Case Study (2008), available from ⟨http://www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study⟩.
[25] Schuchard, M., Mohaisen, A., Kune, D.F., Hopper, N., Kim, Y. and Vasserman, E.Y.: Losing Control of the Internet: Using the Data Plane to Attack the Control Plane, *NDSS 2011*, Internet Society (2011).
[26] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes, *CRYPTO 1984*, LNCS, Vol.196, pp.47–53, Springer-Verlag (1987).
[27] Sriram, K., Borchert, O., Kim, O., Cooper, D. and Montgomery, D.: RIB Size Estimation for BGPSEC (2011), available from ⟨http://www.nist.gov/itl/antd/upload/BGPSEC_RIB_Estimation.pdf⟩.
[28] Verma, R.: Efficient Implementations of Pairing-Based Cryptography on Embedded Devices, Thesis, Rochester Institute of Technology (2015), available from ⟨http://scholarworks.rit.edu/theses⟩.
[29] Vervier, P., Thonnard, O. and Dacier, M.: Mind Your Blocks: On the Stealthiness of Malicious BGP Hijacks,

NDSS 2015, Internet Society (2015).
[30] Rekhter, E.Y., Li, E.T. and Hares, E.S.: A Border Gateway Protocol 4 (BGP-4), RFC 4271 (2006).
[31] Yanai, N., Mambo, M., Tanaka, K., Nishide, T. and Okamoto, E.: Another Look at Aggregate Signatures: Their Capability and Security on Network Graphs, *INTRUST 2015*, LNCS, Vol.9565, pp.32–48, Springer-Verlag (2015).
[32] Zhang, F., Jia, L., Basescu, C., Kim, T.H., Hu, Y. and Perrig, A.: Mechanized Network Origin and Path Authenticity Proofs, *CCS 2014*, pp.346–357 (2014).
[33] Zhao, M., Smith, S.W. and Nicol, D.M.: Aggregated path authentication for efficient BGP security, *CCS 2005*, pp.128–138, ACM (2005).
[34] 中野達也：ルーティングセキュリティインシデント事例の紹介，Internet Week 2013 (2013).

推薦文

AS 間経路制御プロトコルの 1 つである BGPSEC において，ルータのメモリサイズが増加するという問題を解決するために，アグリゲート署名を導入する手法を検討し，メモリサイズの削減，およびアグリゲート署名の導入により生じる問題を解決するトレーシング手法を提案している．実用性が高いと考えられ，具体的な実装と評価により有用性を示しており，ソースコードも公開されていることから信頼性も高く評価できる．

（コンピュータセキュリティシンポジウム 2015
プログラム委員長　山内利宏）

田中 和磨

2014 年筑波大学情報学群情報科学類卒業．2016 年筑波大学大学院博士前期課程修了．同年株式会社野村総合研究所入社，情報セキュリティ業務に従事．2015 年 CSS2015 優秀論文賞．

矢内 直人　（正会員）

2009 年一関工業高等専門学校専攻科修了．2011 年筑波大学大学院博士前期課程修了．2014 年同大学院博士後期課程修了．2014 年より大阪大学大学院情報科学研究科助教．暗号，情報セキュリティの研究に従事．2014 年 SCIS2013 論文賞，2015 年 CSS2015 優秀論文賞，2016 年 CANDAR2016 Outstanding Paper Award 各賞受賞．

岡田 雅之

2000 年東邦大学大学院理学研究科情報科学専攻修士課程修了．同年 ISP，通信事業者においてネットワーク設計，運用に従事．2004 年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入社，2012 年筑波大学大学院システム情報工学研究科リスク工学専攻博士後期課程修了．2014 年東邦大学訪問研究員兼務，2016 年 JANOG 運営委員，インターネット経路制御，IP アドレス管理・運用に関するシステムの研究と開発に従事．

西出 隆志　（正会員）

1997 年東京大学理学部情報科学科卒業．同年日立ソフトエンジニアリング株式会社入社．セキュリティ，ネットワーク製品の設計開発に従事．2003 年南カリフォルニア大学大学院修士課程修了．2008 年電気通信大学博士（工学）．2009 年九州大学大学院システム情報科学研究院助教を経て，2013 年より筑波大学システム情報系准教授．暗号，情報セキュリティの研究に従事．

岡本 栄司　（正会員）

1973 年東京工業大学工学部電子工学科卒業．1978 年同大学大学院博士課程修了．工学博士．同年日本電気入社．その後，北陸先端科学技術大学院大学，東邦大学を経て，2002 年筑波大学教授，2016 年より筑波大学名誉教授，現在に至る．情報セキュリティを中心とする情報数理工学の教育・研究に従事．1990 年電子情報通信学会論文賞，1993 年本会ベストオーサ賞受賞，2008 年本会論文賞，2007 年・2009 年 CHES Best Paper Award．2003 年電子情報通信学会フェロー．著書『暗号理論入門』（共立出版），『電子マネー』（岩波書店）等．IEEE，電子情報通信学会会員．2004 年本会フェロー．