アグリゲート署名を用いたBGPsec AS_PATH検証手法の提案と実装評価

OUYANG Junjie[1]　矢内 直人[1]　岡田 雅之[2]　岡村 真吾[3]

概要：インターネットの経路制御プロトコルとして利用される BGP の拡張において，電子署名を用いて経路情報の生成元の正しさを検証する Origin Validation と経路情報が通過してきたパスを検証する Path Validation という二つの Validation の仕組みで動作する BGPsec の標準化が進んでいる．これらの仕組みは，BGP でやりとりする経路情報に対し，電子署名が付加されることにより，経路情報の増大を招き，結果として BGP ルータのメモリ不足が懸案されている．本稿では，BGPsec の Path Validation におけるメモリ消費量の削減方法に注目し，新たなアグリゲート署名方式を用いて改良した検証手法を提案する．また，その手法に基づいて BGP ルーティングシステムの一つである BIRD を改良したプロトタイプ実装を行い，実際の BGP ルータ同士を想定した実験環境での動作確認と実装評価を行う．実験結果として，200 件の長さ 4 の経路を格納する場合，従来の BGPsec と比べ，メモリ消費量は 80%まで削減できることを示す．

キーワード：BGPsec，インターネット経路制御，Path Validation，アグリゲート署名，メモリ消費量

Proposal and Evaluation of a PATH Validation Method based on Aggregate Signatures for BGPsec

Junjie Ouyang[1]　Naoto Yanai[1]　Masayuki Okada[2]　Shingo Okamura[3]

Abstract: BGPsec is a security extension of the Border Gateway Protocol - the routing protocol of the Internet, which is currently being standardized. BGPsec provides the Origin Validation to authenticate the origin AS and the Path Validation to verify AS_PATH attributes with digital signatures. However, routers should need significantly more memory while running BGPsec to store the digital signatures added to routing information. Our study is focused on how to reduce the memory size through a new PATH Validation method with signature aggregation techniques. We create a virtual BGP network and evaluate our method by using the BIRD Internet routing daemon based on a prototype implementation. As a result, we present that the memory size can be reduced to 80% while receiving 200 routes of 4 ASes.

Keywords: BGPsec, Internet Routing, Path Validation, Aggregate Signature, Memory Size

1. はじめに

1.1 研究背景

インターネットで使われる経路制御プロトコルの一つである Border Gateway Protocol（BGP）[23] では，唯一の番号が割り当てられた自律システム（Autonomous System：以下 AS）と呼ばれる単位で経路情報の交換が行われている．経路情報を AS 間で交換することによって，任意の到

[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] 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
[3] 奈良工業高等専門学校情報工学科
Department of Information Engineering, National Institute of Technology, Nara College

© 2018 Information Processing Society of Japan

達可能な AS へパケットを送信することが可能になり，インターネットの基礎が成立している．しかし，BGP で交換されている経路情報の正当性を保証する機能が備わっていないため，攻撃者が不正な経路情報を送り出しても，それを受け取った AS は正当性を判断できず，現状の経路情報として受理してしまう．この脆弱性を利用した経路ハイジャックの事例として，2008 年のパキスタンテレコムによる YouTube アクセス不能事件 [5]，2018 年の仮想通貨イーサリアムの窃盗事件 [24] などが挙げられる．これらの理由から，経路情報の正当性保証は重要な課題となっている．

経路情報は一般に，ネットワーク層到達性情報（Network Layer Reachability Information：以下 NLRI）と AS_PATH 属性の二つからなる．各 AS が管理・運用しているネットワークの IP アドレスとサブネットマスクを示すのは NLRI である．一方，AS_PATH 属性は経路情報が通過してきた各 AS の AS 番号をリスト化したものである．NLRI と AS_PATH の最後尾にある経路生成元の AS 番号の組み合わせのことを ORIGIN AS 情報といい，パケットの宛先確定に使われる．経路情報の正当性を保証するためには，NLRI と AS_PATH 属性の両方を考慮する必要がある．前者の ORIGIN AS 情報については，電子署名を付与することで正当性を保証する Route Origin Authorization（ROA） [14] [18] の実用化が検討化されている．

一方，AS_PATH 属性の正当性保証についても，電子署名を使用したより安全な経路制御プロトコルとして，Border Gateway Protocol Security Extension（BGPsec） [19] の標準化が検討されている．BGPsec は経路情報に電子署名を付加してから広告し，受け取った経路情報の電子署名を検証することで，経路情報の正当性を確認できるプロトコルである．これにより，ORIGIN AS 情報の正しさを検証する Origin Validation 機能と AS_PATH 属性 の正しさを検証する PATH Validation 機能を同時に提供できるようになる．しかしながら，電子署名の導入により，BGP ルータのメモリ消費量が膨大に増加する恐れがあり，例えば文献 [25] によるとメモリ消費量が 10GB にもなると言われている．また，BGPsec の標準化は 2011 年にその検討が開始したものの，標準化に向けた実装実験などは十分に行われておらず，実際にどの程度の効果があるかも不明である．

1.2 本稿の貢献

本稿では，BGPsec の Path Validation におけるメモリ消費量増大問題に対し，複数の署名を一つに集約可能なアグリゲート署名 [8] を用いた新たな AS_PATH 検証手法を提案する．また，その手法に基づいてプロトタイプ実装を行う．さらに，仮想 BGP ルータからなる仮想ネットワーク環境でのメモリ消費量の測定を通じて，提案手法を評価する．結果として，提案手法では，200 件の長さ 4 の経路を格納する場合，宛先情報に付随するルート属性のメモリ消費量は，従来の BGPsec と比べ，80%まで削減できることを示す．

以下に，本研究における二つの技術的な貢献を述べる．まず，本研究では BGPsec に適したアグリゲート署名を新たに設計している．アグリゲート署名は暗号理論の研究としては BGPsec への応用が期待が期待されている一方，実は署名自体が持つ代数的構造が従来の BGPsec の要件に適していない．より厳密には，単純に既存のアグリゲート署名を導入しただけでは，署名の集約能力か BGPsec の安全性かいずれか一方を損なう可能性がある．本稿ではこの問題を解決するような新たなアグリゲート署名として**併存型アグリゲート署名**という方式を設計することで，BGPsec の安全性を損なうことなく，メモリ消費量の削減を図っている．（詳細は 4 節に記載する．）

二つ目の貢献は，BGP ルータを仮想化するツール BIRD Internet Routing Daemon（以下 BIRD） [1] を改良することで，提案手法を実装したことである．前節で述べた BGPsec の実装実験に関する問題は，評価ツールなどの開発が乏しいことに起因する．これに対し，本稿では BIRD 上に併存型アグリゲート署名を含めた導入改良により，**提案検証手法を実環境で評価できるツールを作成している**．これにより上述したような評価結果を得ることができた．これは将来的には本実装を拡張することで，後続する研究開発用の評価も潜在的に可能にしたことを意味する．（本実装及び実験の詳細は 6 節に記載する．）

2. 関連研究

本節では関連研究として，BGPsec とアグリゲート署名のそれぞれの既存成果，およびそれらの融合成果について簡潔に紹介する．

2.1 BGPsec

BGPsec の研究は，Kent らによる PKI 技術を用いた Secure-BGP（S-BGP） [16] に始まる．これは経路生成元が作成する Address Attestations（AAs）証明書と経路を受け取った各 AS が作成する Route Attestations（RAs）証明書を併用することで，経路情報の不正を検出する手法である．しかし，計算コストなど性能上の問題が原因で，普及に至らなかった．その後，BGP のセキュリティとオーバーヘッドの両立を図る手法として secure origin BGP（soBGP） [28] や Pretty Secure BGP（psBGP） [26] が提案されたが，安全性が低下するなど新たな課題も生まれた [15]．近年においても BGPsec の導入戦略に関する考察や問題点の整理文献 [11] もされているが，未だに抜本的な解決策は文献 [22] によると得られていない．

一方，BGP の安全性要件の具体化や関連する PKI に関する考察は 2004 年に Hu ら [13] が，近年では証明可能安全性

© 2018 Information Processing Society of Japan

の観点からの安全性解析を Boldyreva と Lychev [7] が，既存の BGP の安全性要件の調査を 2014 年に Li ら [20] がそれぞれ行っている．これらの成果により，BGPsec の安全性自体は理論的に解明されつつある．

なお，BGPsec の導入実装の現状としては，BGP 専用の PKI として resource PKI（RPKI） [17] の標準化が完了している．また，BGPsec の評価ツールとしては，AS_PATH 検証用に [1], [4] が，ORIGIN AS 検証用に [2], [3] がそれぞれ開発されている．本稿では BIRD ベースの BIRD BGPsec [1] を改良することで，実装を行っている．

2.2 アグリゲート署名

アグリゲート署名 [8] は独立に生成された平文と署名の組を，単一の署名に集約する技術として提案された．これまでに提案されているアグリゲート署名は逐次型と呼ばれる方式 [21] と非逐次型 [8] と呼ばれる方式に大別される．これらの方式の違いは署名の持つ代数構造にある．逐次型では各署名者は前の署名者から受け取った署名に対して署名をする署名チェーンを作る一方，非逐次型では各署名者が任意のタイミングで署名生成を行うため署名チェーンを持たない．文献 [9] によると BGPsec は前の AS が生成した署名と平文に対して署名することから，BGPsec には逐次型のほうが好ましい．一方で，逐次型では異なる署名チェーン同士は集約できないことが文献 [29] で示されている．これは参入 AS が増加することでネットワークが複雑化した際に，メモリ消費量が大きく増加してしまうことを意味する．本稿で提案する併存型アグリゲート署名は，（大ざっぱにいうと）逐次型で生成された署名チェーンを非逐次型のように集約できる方式と言える．本稿では省略するが，この安全性は文献 [10] の方式と同様の方法で証明が可能である．なお，アグリゲート署名の問題としては，検証に失敗する署名を集約することでともに集約された署名全てが検証に失敗する‘‘巻き添え問題 [31]’’ が知られているが，これは文献 [12], [31] の回避策を使うことで容易に克服できる．

2.3 BGPsec とアグリゲート署名の融合研究

最後に，本研究に最も近い研究は，Path Aggregate Authentication [30] と APAT [31] である．これらの研究では BGPsec（および先駆けとなった S-BGP）の負荷を削減するべく，アグリゲート署名を導入した．しかしながら，これらの研究では最も考慮すべきメモリ消費量について，実験的には示されていない．また，本稿で議論しているようなアグリゲート署名の代数構造に関する改良や，BIRD の改良のような実環境での実装も行われていない．

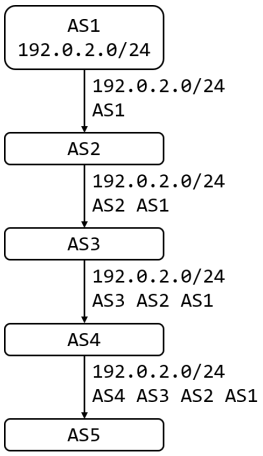

図 1 経路広告

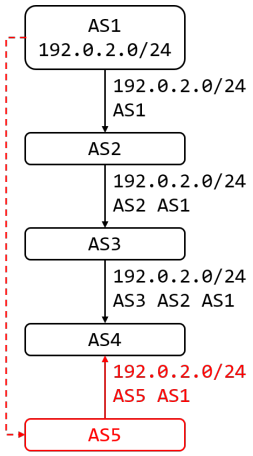

図 2 経路ハイジャック

3. Border Gateway Protocol Security Extension (BGPsec)

3.1 BGPsec の登場背景：経路ハイジャック

BGPsec の登場背景として，BGP での経路ハイジャックの問題について以下に簡潔に説明する．BGP は，インターネット全体での宛先問題を AS 単位で解決するためのプロトコルである．各 AS には，一意な AS 番号が割り当てられている．BGP では，この AS 番号で各 AS を区別し，経路情報を TCP 方式で交換している．AS 間の経路情報交換は，接続確立後に NLRI と AS_PATH を含んだアップデートメッセージの送受信により実現している．

図 1 は正常な BGP 経路広告例を示している．このとき，AS4 に登録される AS1（192.0.2.0/24）への経路は‘‘AS3 AS2 AS1’’ になる．しかし，アップデートメッセージに含まれる AS_PATH は AS によって意図的に書き換えることが可能なため，経路ハイジャックを容易に実現できる．図 2 では，AS5 が AS4 に AS1 へのより短い経路‘‘AS5 AS1’’ を広告している．実際に存在しない経路情報にもかかわらず，AS4 はそれを疑わずに，ベストルート選択アルゴリズムに従って，自分のルーティングテーブルに登録してしまう．

BGPsec [19] は，経路情報に電子署名を付加することで，AS_PATH の正当性保証を通じて経路ハイジャックの正当性も確認できる．本稿では BGPsec の PATH Validation 機能に注目する．具体的な保証手法について次節にて詳しく説明する．

3.2 PATH Validation

BGPsec では，AS_PATH 属性の代わりに，BGPsec_PATH 属性が新たに定義されている [19]．BGPsec_PATH 属性は，図 3 に示される Secure_PATH と Signature_Block によって構成される．Secure_PATH は経路情報が通過してきた各 AS の AS 番号をリスト化したものであり，従来の AS_PATH

© 2018 Information Processing Society of Japan

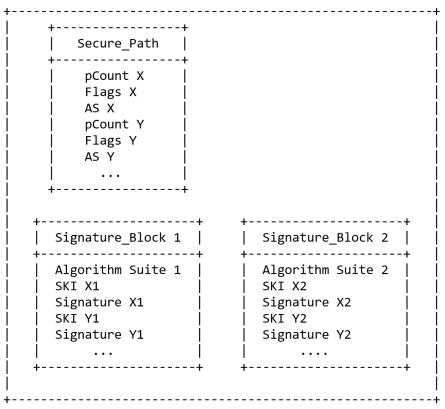

図 3 BGPsec_PATH Attribute Format

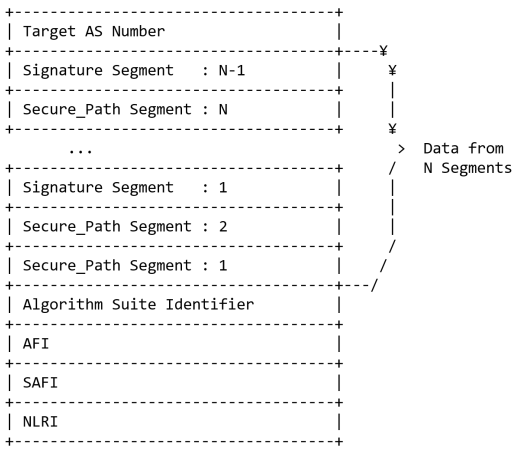

図 4 Sequence of Octets to Be Hashed

と同等である．一方，Signature_Block は Secure_PATH の中の各 AS が付加した電子署名を格納するところである．Algorithm Suite Identifier の値によって指定される署名アルゴリズムに応じて，署名長が可変する．

公開鍵や秘密鍵の生成や管理については省略するが，図 4 に示される署名対象となるデータ列の各パラメータについて簡単に説明する．

Target AS Number：経路情報の送り先の AS 番号
Signature Segment：追加された電子署名の値（通過してきた AS の数と同個数であり，経路生成元の場合は無し）
Secure_Path Segements：通過してきた各 AS の AS 番号の値（少なくとも経路生成元の AS 番号が必要）
Algorithm Suite Identifier：署名生成において利用する署名アルゴリズムを特定するための識別子
AFI：アドレスの識別子（IPv4 or IPv6）
SAFI：AFI で識別されたプロトコルの詳細識別
NLRI：経路生成元が管理しているネットワークのアドレスとサブネットマスクの値

各 AS がアップデートメッセージを送受信する際の動作について簡単に説明する．

（ORIGIN AS の場合）経路生成元のため，署名の検証は不要である．送り先の AS 番号を確定後，所持の秘密鍵を利用し，図 4 に示されている（N = 1 としたときの）データ列に対して署名を行う．その後，必要な情報を取り出し，アップデートメッセージを作成し送信する．

（中継 AS の場合）経路情報が AS を通過するたびに，Signature Segment と Secure_Path Segement が 1 個ずつ増えていく．それゆえに，経路情報を受信後，アップデートメッセージの中からから適切な情報を抽出し，署名の対象となるデータ列を最も小さいものから再構成する必要がある．Signature_Segment に含まれる SKI の値を利用し公開鍵を取得した上で，各データ列の正当性を検証する．経路情報一件あたりの検証回数は AS_PATH 長に等しい．検証に通ったら，最適な経路（あるいは新規経路）だと判明された場合にのみ，ルーティングテーブルに反映する．当該経路を別の AS に広告する際は，図 4 に示されりデータ列（N = AS_PATH 長 + 1）に対して署名を行い，アップデートメッセージを作成し送信する．

表 1 BGPsec のメモリサイズ推定値

単位：GB

年	BGP		BGPsec	
	経路数	メモリサイズ	経路数	メモリサイズ
2016	7124634	0.15	116252	0.5
2017	7829923	0.17	497095	0.22
2018	8119682	0.17	1456390	0.64
2019	7660160	0.16	3352322	1.48
2020	6332177	0.13	6332177	2.79
2021	4433446	0.09	10130562	4.47
2022	2547235	0.05	14201374	6.62
2023	1149812	0.02	18111088	7.99
2024	355617	0.01	21794419	9.61
2025	0	0	25472541	11.23

3.3 BGPsec の問題点

BGPsec では，アップデートメッセージに署名情報が付加されるため，アップデートメッセージのサイズが膨大し，データの大半を電子署名が占用することなる．

アメリカ国立標準技術研究所（NIST）が BGPsec の展望調査として，BGPsec ルータにおけるメモリ消費量を推定した [25]．具体的に，BGP アップデートメッセージの平均サイズは 78 Bytes であることに対して，BGPsec アップデートメッセージは署名アルゴリズムによって 388 Bytes から 1188 Bytes になると考えられている．また，BGPsec は 2016 年頃に運用を始め，それ以降，署名を付加した経路が迅速に増えていき，2025 年頃に導入完了と予想された．経路数の増加と伴う ECDSA-256 の署名方式を利用した BGPsec ルータのメモリ消費量変化を年ごとにまとめたもとを**表 1** に示している．表中のメモリサイズは，宛先情報の登録されているルーティングテーブルやそれらに付

© 2018 Information Processing Society of Japan

Algorithm 1 Setup

Ensure: グローバルパラメータ $para$

1: 双線形パラメータ $(p, \mathbb{G}, \mathbb{G}_T, \mathbf{e})$ を生成
2: $P \leftarrow \mathbb{G}$
3: ハッシュ関数 $H : \{0,1\}^* \rightarrow \mathbb{G}$ を選択
4: $para = (p, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, P, H)$

Algorithm 2 UserKeyGen

Require: グローバルパラメータ $para$

Ensure: 秘密鍵 sk，公開鍵 pk

1: 乱数 $x \leftarrow \mathbb{Z}_p$ を生成
2: $X = xP$
3: $sk = x$，$pk = X$

随するルート属性などの合計値を表している．この予測では，世界規模の経路情報，いわゆるフルルートを運用する BGPsec ルータのメモリ消費量は 10GB 以上にもなる．

4. 併存型アグリゲート署名

本節では Path Validation に向けた新たな署名方式として併存型アグリゲート署名（Bimodal Aggregate Signatures）を提案する．これは n 人の署名者がそれぞれ独立した n 個のデータに署名する際，逐次型アグリゲートにおける署名チェーン [21] と非逐次型アグリゲート [8] における独立に生成された署名の合成という二つの集約機能を包括する方式である．これは従来のアグリゲート署名と比べて，署名チェーンを通じた従来の BGPSEC の安全性と，互いに独立したチェーン同士の署名集約も可能な性質を併せ持つ点で利点が大きい．以下に併存型アグリゲート署名の構成を述べる．

本方式では以下に定義する双線形写像を用いる．$\mathbb{G}$，$\mathbb{G}_T$ を素数位数 p を持つ群とする．このとき，双線形写像 $\boldsymbol{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ は以下の条件を持つ写像とする．まず任意の $U, V \in \mathbb{G}$ および $a, b \in \mathbb{Z}_p^*$ において，$\boldsymbol{e}(aU, bV) = \boldsymbol{e}(U, V)^{ab}$ となる．次に，任意の生成元 $P \in \mathbb{G}$ において，$\boldsymbol{e}(P, P) \neq 1_{\mathbb{G}_T}$ となる．ここで，$1_{\mathbb{G}_T}$ は $\mathbb{G}_T$ 上の単位元とする．最後に，任意の $U, V \in \mathbb{G}$ において，効率的に $\boldsymbol{e}(U, V)$ を計算可能である．また，本稿では群 $\mathbb{G}$，$\mathbb{G}_T$ 上の離散対数問題（DLP）は困難とする．このとき，上述の条件を満たす $\mathbb{G}$ を双線形群と呼び，そのパラメータ $(p, \mathbb{G}, \mathbb{G}_T, \boldsymbol{e})$ を双線形パラメータ（pairing parameter）と呼ぶ．

また，以下では各署名者は互いに一意な番号 i を持つものとする．任意の署名における署名者の集合を S とし，任意の i に関して i 番目の署名者までの署名チェーンに参加する署名者の集合を S_i とする．また，文字列間の連結を $\|$ とし，とくに S_i に属する署名者が持つ文字列間の連結を任意の j を用いて $\|_{j \in S_i}$ で表す．

Algorithm 1-5 に方式を記載する．**Algorithm 3** の 2 行目および **Algorithm 5** の 3 行目の処理が，それぞれ署名チェーンに相当する．より厳密には σ は署名を指しており，**Algorithm 3** の $\mathbf{e}(\sigma, P)$ と **Algorithm 5** の $\mathbf{e}(c_j, X_i)$ はそれぞれ署名の検証式の核となる計算に対応している．これは署名とその検証式をそれぞれハッシュ関数に入力することで，署名チェーンを構成していることを意味する．すなわち，σ を圧縮したとしても，対応する $\mathbf{e}(c_j, X_i)$ を検証式で計算することで署名チェーン自体が構成できる．これにより，逐次型による署名チェーンと非逐次型による互いに独立した署名の圧縮が両立できる．

Algorithm 1-5 に記載した方式の安全性は，形式的に定義と証明を行うことが可能である．紙面の都合上，詳細は省略するが，ランダムオラクルモデルにおいて CDH 仮定の下で偽造不可能であることが証明できる．

Algorithm 3 SeqAggSign

Require: グローバル $para$，秘密鍵 sk_i，公開鍵 pk_i，平文 $m_i \in \{0,1\}^*$，公開鍵-平文リスト $L = \{(pk_j, m_j)\}_{j \in S}$，署名 σ

Ensure: 署名 σ，公開鍵-平文リスト $L' = \{(pk_j, m_j)\}_{j \in S} \cup \{(pk_i, m_i)\}$

1: $L = \emptyset$ の場合，$\sigma = 0$ と設定
2: $c = H\left(\mathbf{e}(\sigma, P) \parallel pk_i \parallel m_i \parallel_{j \in S_i} (pk_j \parallel m_j)\right)$
3: $\sigma = \sigma + x \cdot H(c)$

Algorithm 4 AggSign

Require: グローバル $para$，公開鍵-平文リスト 1 $L_1 = \{(pk_j, m_j)\}_{j \in S}$，公開鍵-平文リスト 2 $L_2 = \{(pk_j, m_j)\}_{j \in S'}$，署名 $1\sigma_1$，署名 $2\sigma_2$

Ensure: 署名 σ，公開鍵-平文リスト $L' = L_1 \cup L_2$

1: $\sigma = \sigma_1 + \sigma_2$

Algorithm 5 Verify

Require: グローバル $para$，公開鍵-平文リスト $L = \{(pk_j, m_j)\}_{j \in S}$，署名 σ

Ensure: True or False

1: 任意の $i \in S$ において pk_i を X_i として構文解析
2: 全ての $(pk_i, m_i) \in S$ が互いに異なるか確認
3: $\forall i, c_i = H\left(\left(\prod_{j \in S} \mathbf{e}(c_j, X_i)\right) \parallel pk_i \parallel m_i \parallel_{j \in S_i} (pk_j \parallel m_j)\right)$
4: $\mathbf{e}(\sigma, P) = \prod_{i \in S} \mathbf{e}(H(c_i), X_i)$ が成り立つか確認
5: **return** 両方の確認ができたなら True，そうでないなら，False

5. 提案手法

前述の BGPsec の PATH Validation の仕様に基づいて，BGPsec におけるメモリ消費量の削減を図り，新たプロトコルを構成する．

新たなプロトコルを使った経路広告のイメージを図 **5** に示す．各中継 AS は受信したアップデートメッセージに含まれる必要な情報を抽出し，**Algorithm 5** で検証を行う．その後，当該経路と受信した電子署名に対し **Algorithm3** を通じて，新たな署名の生成と集約を行う．その署名を次の AS へ送信する．また，異なる ORIGIN を持つ別に受信した経路情報とその署名に関しては，上述した手法で署名を生成した後，**Algorithm4** を通じて署名同士を集約する．

© 2018 Information Processing Society of Japan

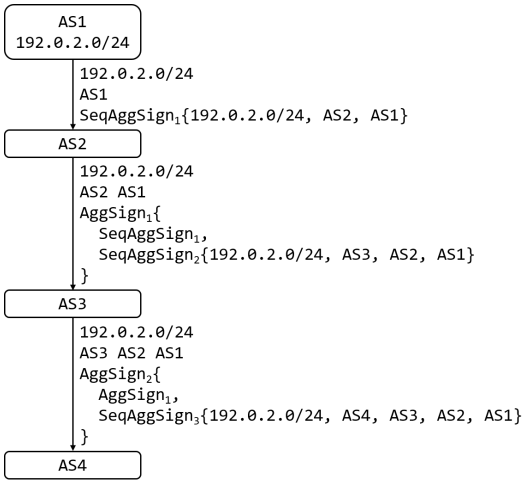

図 5 提案手法における経路広告

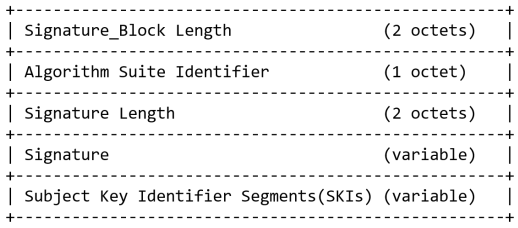

図 6 新しい Signature_Block Format

以下は具体的な構成について述べる．

BGPsec で定義された Signature Segment Format は，20 Bytes の Subject Key Identifier（SKI），2 Bytes の Signature Length および可変長の Signature から構成されている．複数の署名の存在を回避するために，SKI の部分を独立した SKI Segment（20 Bytes）として定義し，新しい Signature_Block Format を図 **6** に示す．

署名方式は **Algorithm 1-5** を適用する．ここで要注意なのは，平文 m はあらかじめ用意するものではなく，受信したアップデートメッセージの情報を利用して再構成するものであり，図 4 に示されているデータ列から Signature Segment Format を除いたものを利用する．また，署名を生成する際に必要な一つ前の署名者が使ったハッシュ値に関しては，アップデートメッセージに含まず，経路情報の検証時に得られた演算結果を再利用する．これにより，アップデートメッセージのサイズと格納すべきデータ量を抑えることが可能になる．

6. 実装と評価

ルーティングシステム BIRD の BGPsec 拡張版 [1] と，ペアリングライブラリ TEPLA [6] を用いて，前述の提案手法の実装を行い，動作確認を行う．また，BIRD で仮想線状ネットワークを構築し，新たな検証手法のメモリ消費量を測定し，従来の検証手法と比較する．

表 2 ライブラリおよび利用パラメータ

TEPLA ver.	2.0
ペアリング	ECBN254a
有限体	bn254_fpa
P（事前生成）	1462ea218754f628c4...

6.1 実装

BIRD は，C 言語で書かれた，一台の計算機を一台の BGP ルータとして使用することができるシミュレーションツールである．コンフィグレーションファイルを編集することで，プロトコルの増減，経路情報の書き込み，フィルタの設定など，様々な変更ができる．現在，BIRD の BGPsec 拡張版として，The BGPsec enabled Bird Routing Daemon [1] が Secure Routing（http://www.securerouting.net/）にて公開されている．本稿では，BIRD の BGPsec 拡張版をベースに実装を行う．

BGP に関連するソースコードは proto/bgp/の下に存在する．前述の提案手法を実現するためのコード変更点は主に二つある．一つは，アップデートメッセージのエンコードとデコードである．attrs.c にある encode_bgpsec_attr() 関数と decode_bgpsec_attr() 関数を中心に，アップデートメッセージのエンコードとデコードおよび各属性の格納を実現する．もう一つは，具体的な署名と検証手法である．validate.c にある bgpsec_sign_data_with_key() 関数と bgpsec_verify_signature_with_key() 関数の中身を，TEPLA による併存アグリゲート署名の実装に書き換える．ペアリング計算に用いるパラメータを**表 2** に示す．

6.2 実験環境

BIRD は様々なネットワークトポロジに対応しているが，経路数及び平均 AS_PATH 長をより容易に把握できるようなトポロジとして，最も簡単な線状ネットワークが考えられる．この場合，ネットワーク中に交換される経路数は各 AS が広告する静的経路の総数になり，平均 AS_PATH 長（＝平均署名数）は静的経路を広告する各 AS との距離を平均したものになる．

仮想ルータを**表 3** の実験環境のもとで合計 6 台作成し，プライベート AS 番号 65001〜65006 を割り当てる．また，それぞれ AS 番号の下一桁に応じて固定 IP の 192.168.0.201〜206 を割り振り，各 AS を接続させ，線状ネットワークを構成する．AS65001 にのみ静的経路を広告させる場合，他の各 AS が同様数量の経路情報を受け取ることとなり，平均 AS_PATH 長が AS 番号の差になるため，データの収集が比較的に容易に行える．BIRD が安定して広告できる静的経路の上限はおおよそ 250 件であるため，AS65001 が広告する静的経路の件数を 50，100，150，200，250 に指定した．具体例として，AS65101 のみに 200 件の静的経路を広告させ，AS65004 がそれを受け取る場合，AS65004

© 2018 Information Processing Society of Japan

表 3　実験環境

OS	Ubuntu 16.04 LTS
CPU	Intel Core i7-6500U
メモリ	1 GB
VMware ver.	Workstation Pro 14.1.2
BIRD ver.	1.6.0
BIRD BGPsec ver.	0.9

が持つ経路情報の総数が 200 件になり，各 AS_PATH の長さが 3 になる．この設定により，200 件の長さ 3 の経路を持っているルータのメモリ消費量の計測を実現できる．

上述した環境において，AS65002〜AS65006 のメモリ消費量をそれぞれ計測した．なお，本稿では，Path Validaion の演算部分にのみ注目しているため，鍵の取得方法は問わない．鍵のペアは事前に生成されたものとし，公開鍵もあらかじめ共有できているものとする．また，広告される経路情報は事前にランダムに生成したものを使う．

6.3　実験結果

BIRD のメモリ使用量情報は，ルーティングテーブル，ルート属性，ROA テーブル及びプロトコルを包括しているが，ROA テーブルとプロトコルのサイズは今回の実験では変化しないため，本稿では，経路の一覧であるルーティングテーブル及それらに付随するルート属性が占有するメモリ量のみを計測する．計測結果の単位はすべて KB としている．

提案手法において，計測結果の一例として，各 AS が 200 件の経路情報を受信したときのメモリ消費量を**表 4** に示す．また，同じ 200 件の経路情報を受信したときの BGP ルータ及び BGPsec ルータのメモリ消費量を**表 5** と**表 6** に示し，三つの表のデータをまとめてグラフ化したものを**図 7** に示している．

この結果から，本稿で提案している検証方式のメモリ消費量は，BGPsec より増幅が小さいことがわかる．平均 AS_PATH 長が 1 の場合にのみ，新手法のほうがメモリを多く消費していることに関しては，署名単体のサイズが原因だと考えられる．また，Wang らの調査 [27] によると，インターネットの経路情報の平均 AS_PATH 長はおおそよ 3.9 となっている．本実験において，平均 AS_PATH 長が 4 の場合，提案手法のルート属性のサイズは従来の 80%しか要しないことがわかる．今回は長さ 5 以上の経路を使わなかったが，AS_PATH が長ければ長いほど，メモリ消費量の削減率が大きくなる．異なるルート属性が増えたとしても提案手法では **Algorithm 4** で圧縮ができるので，メモリ消費量が増えない．

7.　まとめ

BGPsec では，電子署名を導入することにより，メモリ消費量の増加が問題視されるようになっている．本稿では，複数の電子署名を一つに集約できるアグリゲート署名を BGPsec へ適用する方法を考案した．また，BIRD を使用して，提案手法を実装し，実際のメモリ消費量を定量的に計測した．また，BGP ルータと BGPsec ルータのメモリ消費量と比較し，200 件の長さ 4 の経路を受信した場合のメモリ消費量を 80%までに削減できるなど，提案手法の有効性を実際の環境にて示した．

今後の課題は，複雑なトポロジでのシミュレーションと経路格納速度の向上である．本稿では，最も簡単な線状ネットワーク環境でシミュレーションを行い，メモリ消費量を計測した．しかし，実際のネットワークトポロジのようなより複雑な環境についても確認する必要がある．また，アグリゲート署名の計算量は従来の署名方式より大きいため，計算に時間がかかってしまい，ルータの性能が低下するという新たな問題が発生した．実際に運用されている BGP ルータの経路交換が少なく，経路の格納時間よりもメモリ消費量のほうが影響が大きいとも考えられるが，ミスコンフィグレーションなどにより，経路情報を大量に受信した場合の対処などを考慮し，格納時間の短縮方法も今後の課題となる．

謝辞　本研究は JSPS 科研費番号 18K18049 およびセコ

表 4　提案手法のメモリ消費量（経路数 200）

単位：KB

AS 番号	65002	65003	65004	65005	65006
ルーティングテーブル	46	46	46	46	46
ルート属性	116	122	128	134	140

表 5　BGP ルータのメモリ消費量（経路数 200）

単位：KB

AS 番号	65002	65003	65004	65005	65006
ルーティングテーブル	46	46	46	46	46
ルート属性	10	10	10	10	10

表 6　BGPsec ルータのメモリ消費量（経路数 200）

単位：KB

AS 番号	65002	65003	65004	65005	65006
ルーティングテーブル	46	46	46	46	46
ルート属性	104	124	144	164	188

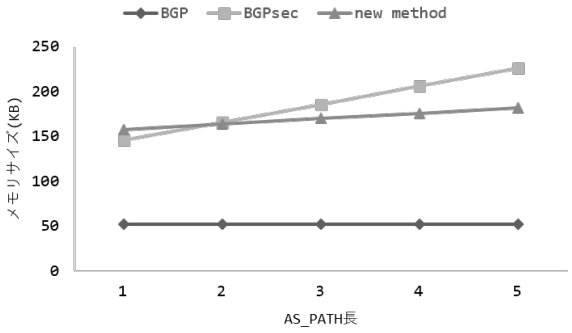

図 7　メモリ消費量

© 2018 Information Processing Society of Japan

ム財団挑戦的研究助成により支援されている．

参考文献

[1] Bird bgpsec. http://www.securerouting.net/tools/bird/.

[2] Frrouting. https://github.com/FRRouting/frr.

[3] Go-bgp. https://osrg.github.io/gobgp/.

[4] Nist bgp secure routing extension (bgp / srx) prototype. https://www.nist.gov/services-resources/software/bgp-secure-routing-extension-bgp-srx-prototype.

[5] YouTube Hijacking: A RIPE NCC RIS case study. 2008.

[6] TEPLA (University of Tsukuba Elliptic Curve and Pairing Library), 2015.

[7] Alexandra Boldyreva and Robert Lychev. Provable security of S-BGP and other path vector protocols: model, analysis and extensions. In *Proc. of CCS 2012*, pp. 541–552, 2012.

[8] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. pp. 416–432, 2003.

[9] Kyle Brogle, Sharon Goldberg, and Leonid Reyzin. Sequential aggregate signatures with lazy verification from trapdoor permutations - (extended abstract). In *Proc. of ASIACRYPT 2012*, pp. 644–662, 2012.

[10] Marc Fischlin, Anja Lehmann, and Dominique Schroder. History-free sequential aggregate signatures. In *Proc. of SCN 2012*, pp. 113–130, 2012.

[11] Phillipa Gill, Michael Schapira, and Sharon Goldberg. Let the market drive deployment: a strategy for transitioning to BGP security. In *Proc. of ACM SIGCOMM 2011*, pp. 14–25, 2011.

[12] Gunnar Hartung, Bjorn Kaidel, Alexander Koch, Jessica Koch, and Andy Rupp. Fault-tolerant aggregate signatures. In *Proc. of PKC 2016*, pp. 331–356, 2016.

[13] Yih-Chun Hu, Adrian Perrig, and Marvin A. Sirbu. SPV: secure path vector routing for securing BGP. In *Proc. of ACM SIGCOMM 2004*, pp. 179–192, 2004.

[14] Geoff Huston and George G. Michaelson. Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs). RFC 6483, February 2012.

[15] Geoff Huston, Mattia Rossi, and Grenville Armitage. Securing bgp - a literature survey. *IEEE Communications Surveys & Tutorials*, Vol. 13, No. 2, pp. 199–222, 2011.

[16] Stephen Kent, Charles Lynn, and Karen Seo. Secure border gateway protocol (S-BGP). *IEEE Journal on Selected areas in Communications*, Vol. 18, No. 4, pp. 582–592, 2000.

[17] Martin Lepinski and Stephen Kent. An Infrastructure to Support Secure Internet Routing. RFC 6480, 2012.

[18] Matt Lepinski, Derrick Kong, and Stephen Kent. A Profile for Route Origin Authorizations (ROAs). RFC 6482, February 2012.

[19] Matthew Lepinski and Kotikalapudi Sriram. *BGPsec Protocol Specification*. No. 8205 in Request for Comments. RFC Editor, 2017. Published: RFC 8205.

[20] Qi Li, Yih-Chun Hu, and Xinwen Zhang. Even rockets cannot make pigs fly sustainably: Can bgp be secured with bgpsec? In *Proc. of SENT 2014*, 2014.

[21] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In *Proc. of EUROCRYPT 2004*, pp. 74–90, 2004.

[22] Asya Mitseva, Andriy Panchenko, and Thomas Engel. The state of affairs in bgp security: A survey of attacks and defenses. *Computer Communications*, Vol. 124, No. 2018, pp. 45–60, 2018.

[23] Yakov Rekhter, Susan Hares, and Tony Li. *A Border Gateway Protocol 4 (BGP-4)*. No. 4271 in Request for Comments. RFC Editor, 2006. Published: RFC 4271.

[24] Aftab Siddiqui. What Happened? The Amazon Route 53 BGP Hijack to Take Over Ethereum Cryptocurrency Wallets.

[25] K Sriram. RIB Size Estimation for BGPSEC. 2011.

[26] Tao Wan, Evangelos Kranakis, and Paul C. van Oorschot. Pretty secure bgp (psbgp). In *Proc. of NDSS 2005*. IEEE, 2005.

[27] Cun Wang, Zhengmin Li, Xiaohong Huang, and Pei Zhang. Inferring the average as path length of the internet. In *Proc. of IC-NIDC*, pp. 391–395. IEEE, 2016.

[28] Russ White. Securing bgp through secure origin bgp. *The Internet Protocol Journal*, Vol. 6, No. 3, pp. 47–53, 2003.

[29] Naoto Yanai, Masahiro Mambo, Kazuma Tanaka, Takashi Nishide, and Eiji Okamoto. Another look at aggregate signatures: Their capability and security on network graphs. In *Proc. of INTRUST 2015*, pp. 32–48, 2015.

[30] Meiyuan Zhao, Sean W. Smith, and David M. Nicol. Aggregated path authentication for efficient BGP security. In *ACM, CCS 2005*, pp. 128–138, 2005.

[31] 田中和磨, 矢内直人, 岡田雅之, 西出隆志, 岡本栄司. Apat: An application of aggregate signatures to bgpsec. 情報処理学会論文誌, Vol. 58, No. 2, pp. 544–556, 2017.

© 2018 Information Processing Society of Japan