Vol.2018-DPS-175 No.39
Vol.2018-MBL-87 No.39
Vol.2018-ITS-73 No.39
2018/5/25

vAS: インターネット環境におけるスケーラブルで柔軟なトラフィック・エンジニアリング機構

近藤 賢郎[2,a)]　寺岡 文男[1,b)]

概要：近年 OpenFlow, segment routing，network service header に代表される要素技術をもとに最短経路に基づかない end-to-end の通信路をプログラマブルに構成するトラフィック・エンジニアリング (TE) 手法の開発が盛んである．しかし，これらの TE 手法では network service provider (core entity) と application service provider (end entity) 間でのセマンティクスの共有がなされず，core entity は end entity のセマンティクスに基づかない通信路しか形成できない．また end entity による通信路構築が適切かを core entity で検証する機構も存在しない．本稿では，インターネットに代表される Open Network 環境を対象とする，core entity と end entity との間でのセマンティクス共有を前提とした TE 手法を提案する．共通のセマンティクス理解する entity 群は closed な overlay network (vAS: virtualized autonomous system) に属する．vAS では end entity と core entity の双方のセマンティクスを考慮した end-to-end の通信路が構成され，その通信路に従ってパケットは転送される．vAS 上でのパケット転送は既存の Open Network 環境のパケット転送に failover 可能なので，end-to-end に渡った柔軟な通信路形成とその到達可能性が両立する．

キーワード：トラフィック・エンジニアリング，Software Defined Networking，Policy-baced Rrouting

vAS: A Mechanism for Scalable and Semantics-aware Traffic Engineering in the Internet

Kondo Takao[2,a)]　Teraoka Fumio[1,b)]

1. はじめに

インターネットに代表される *Open Network* 環境ではトラフィック流量の制御のためのトラフィック・エンジニアリング (TE) 技術の開発が盛んである．これらの TE 技術は Autonomous System (AS) 内ないしは AS 間に跨がって適応されるものに大別される．AS 内では明示的にラベルパスを設置する手法 (e.g., MPLS-TE[1], RSVP-TE[2]) と Layer3 マルチパス (e.g., ECMP[3], OSPF-OMP[4], PEFT[5]) を用いた手法が利用される．一方 AS 間では community や local preference といった BGP メッセージの属性に従った手法が利用される．

これら伝統的な TE 技術では事前に明示的なラベルパスの設定が必要であったり宛先アドレスを中心としたトラフィック制御しか実現できない．このような伝統的な TE 技術が抱える問題を克服すべく，end-to-end の通信路 (*path*) をより柔軟・プログラマブル (path programmable) に構築してトラフィック流量制御を目指す TE 技術が近年提案されている．Path programmable な TE 技術は以下の 3 つに大別される: (i) *Flow-based approach* (OpenFlow[6], BGP Flowspec[7] etc.), (ii) *Source routing* (e.g., Segment routing[8]), (iii) *Network service ID tags* (Network Service Header[9] etc.)．これらの要素技術は場合により組み合わされて複雑な機能を持った *Network Function* を構成し

[1] 慶應義塾大学理工学部
Faculty of Sciense and Technology, Keio University
[2] 慶應義塾大学大学院理工学研究科
Graduate School of Sciense and Technology, Keio University
[a)] latte@inl.ics.keio.ac.jp
[b)] tera@keio.jp

TE に利用される．

これらの TE 技術を用いた path は Open Network を構成する 2 種類の主体 (entity) の何れかによって構築される．(a) *Core entity* (network service provider): Open Network に接続するネットワークを構築・定常的に運用する．(b) *End entity* (application service provider): アプリケーション・サービスを構築・定常的に運用し，network service provider からは Open Network への接続性をサービスとして購読する．Core entity は (i) や (iii) の TE 技術を用いて path を構築する．End entity は (ii) の TE 技術を用いて path を構築する．しかし何れの手法においても path を構築する entity と他の entity との間で構築した path に関するセマンティクスを共有する手段が欠落する．その結果 core entity が構築した path に対して end entity でのアプリケーション・メッセージのセマンティクスを反映できず効率的な in-network caching を形成できなかったり，end entity が構築した path が悪意あるものかを core entity で validation できないといった問題点が生ずる．

本稿では end entity / core entity 両者でのセマンティクス共有を前提とした Open Network における TE 手法を提案する．本手法では Open Network に属する entity のうち共通のセマンティクスを理解する集合として *virtualized autonomous system* (*vAS*) をオーバレイ・ネットワークとして構築する．ここでの entity とはネイティブな Open Network を構成する autonomous system (AS) である．vAS の control plane は属する entity 間で共有されるセマンティクスを交換しそれらに従った path を構築し，data plane はその path に従ってデータを転送する．vAS 上で構築した path は vAS 内で共有されたセマンティクスに従って構築されるのに対して，その underlay にある Open Network 環境 (native network) では最短経路に従った path が利用可能である．従って vAS 上でのデータ転送は native network に failover 可能となり，現在の Open Network で実現されている end-to-end 到達性を害さずに柔軟な path の構成が実現する．

2. vAS に基づく TE のユース・ケース

本節では end entity / core entity 間でのセマンティクス共有を前提とした TE 手法のユース・ケースを述べる．

2.1 Semantics-aware multipath selection

Core entity は自ネットワーク内でのトラフィック負荷分散を目的としてマルチパス・ルーティングを実施することがある．Path programmable な TE 手法では Layer3 マルチパス等の伝統的な TE 手法の限界を超えてフローや Network Service Header に従った path の構築が可能となった．しかし end entity / core entity 間のセマンティクス共有機構が欠落しているため core entity が path を構築する際に end entity で動作するサービスの特性やセマンティクスを反映することができない．

例えば同じ宛先に対して広帯域かつ高遅延 (廉価) な回線と狭帯域かつ低遅延 (高価) な回線がある場合，片方向通信に基づく大容量コンテンツ配信トラフィックは前者の回線を通して双方向通信に基づくテレビ会議トラフィックは後者の回線を通すといったサービスの特性に応じたマルチパスの選択ができない．また，特に core entity が in-network caching を生成する場合には，end entity で動作するサービスのセマンティクスを考慮しながらトラフィックをマルチパスに分割する必要がある．

2.2 Semantics-aware in-network caching

Core entity はトランジット回線を通過するトラフィック削減を目指して自ネットワーク内に in-network caching を生成することがある．このとき end entity で動作するサービスのセマンティクスに従ってコンテンツの特性の判別がなければ再利用されない冗長な in-network caching が生成される．

例えば YouTube のような動画ストリーミングサービスのコンテンツは再利用可能性があるのに対して VoIP のような一対一の通信は再利用可能性がない．この場合 in-network cache を生成すべきなのは前者のみである．また動画ストリーミングサービスの中でも権利性のあるコンテンツを配信するサービスの場合は，アカウンティングの実施可能な entity でのみ in-network caching を生成してその再利用が可能である．

2.3 Precise resource provisioning in core entities

Core entity はこれまでのネットワーク資源の消費状況から将来必要となる資源量を予測して TE を実施したり回線やネットワーク機器に対して設備投資する．しかし特に大規模なコンテンツ配信を行う end entity からのネットワーク資源の消費に関する情報なしには今後必要なネットワーク資源量に対応する TE や設備投資が困難となる．

例えば iOS update や Windows Update といった全世界規模での定期的なコンテンツ配信の場合，その配信元となるデータセンタが変更されることがある．このとき，配信元となるデータセンタの変更が core entity に通知されないままネットワーク資源への TE や設備投資を施しても，将来のネットワーク資源の消費に対して効果的な施策とはならない．

2.4 Semantics-aware network functions for commodity traffic

Open Network を流れるトラフィック (*commodity traffic*) を対象として特定の network funciton を適応する取組がある．その中でも，IETF の working group (I2NSF,

Vol.2018-DPS-175 No.39
Vol.2018-MBL-87 No.39
Vol.2018-ITS-73 No.39
2018/5/25

MILE, SACM, DOTS など) でも活発に議論されているように，commodity traffic を対象にセキュリティに関する network function (*network security function*; *NSF*) を適応するための取組が盛んである．この NSF を構成する上でも end entity / core entity 間でのセマンティクス共有が有用となる．

例えばゲートウェイ型の Intrusion Detection System (IDS) において end entity からのトラフィックを観測して異常検知 (ふるまい検知) する場合，end entity で動作するサービスのメッセージ・セマンティクスを踏まえて観測する方が精度よく異常検知が可能となる．

3. 提案する TE 手法に対する要求事項

以上のようなユース・ケースを満たすような TE 手法を実現するためには，以下に示す 5 個の要求事項を満たす必要がある．

(i) *Path programmability*: end-to-end に渡って最短経路以外の path を柔軟に構築できる必要がある．そのためには path の構築後に，構築した path に関する情報を Open Network を構成する entity 間で交換する必要がある．また最短経路に基づかない path を構築する場合，意図しないルーティング・ループを path 中から排除する必要がある．

(ii) *Semantics awareness*: path を構築する際に end entity / core entity 間でお互いのセマンティクスを共有する必要がある．End entity のセマンティクスを理解した上で core entity がマルチパスや in-network cache を構築することで，end entity 上で動作するサービスのセマンティクスに矛盾しない効果的な TE が可能となる．また end entity におけるネットワーク資源の消費に関する予測を core entity が共有することによって実際の資源消費に見合った適切な TE が実施できる．

(iii) *Scalability*: Open Network 規模のネットワークにおけるスケーラビリティを害さない必要がある．Open Network 環境で交換される経路情報はアドレス資源の階層構造に従って順次集約される．Entity 間で共有されるセマンティクスはポリシの一種であるため経路情報のような集約が難しいことから，スケーラビリティを害さないためにそれらが共有される範囲は closed な範囲 (*domain*) に留める必要がある．

(iv) *Reliability*: end entity / core entity の両者でセマンティクスの共有が適切になされていることの validation 機構が必要である．まず end entity が構築した path に対して悪意あるルーティング・ループが含まれていないかを確認する機構が必要である．また core entity が path 中で in-network cache を生成する際には end entity が示したコンテンツの特性 (e.g., 権利性など) に応じた取り扱いがされているかを確認する機構が必要である．

(v) *End-to-end connectivity*: 柔軟な path の構築を許容する一方で，それが原因となって Open Network で実現されている end-to-end 到達性を害さない必要性がある．現在のインターネットでは最短経路に基づく経路情報を BGP によって広告・交換し合って全世界規模の end-to-end の到達性を確保している．End-to-end の到達性はこのような単純な機構によって担われるべきで有り end entity / core entity が構築した path とは独立したものであるべきである．つまり end entity / core entity のセマンティクスを反映した path に従った際に end-to-end 到達性が確保できない場合は最短経路に基づく path に failover 出来ることが必要である．

表 1　要求事項に従った関連研究の比較

	Path programmability	Semantics awareness	Scalability	Reliability	End-to-end connectivity
Flow-based approach	O (OpenFlow) / X (Flowspec)	X	X (OpenFlow) / O (Flowspec)	X	X (OpenFlow) / O (Flowspec)
Source routing	O	X	-	X	O
Network service ID tags	O	X	-	O	△
vAS-based (proposal)	O	O	O	O	O

4. 関連研究

本節では 1 節で示した 3 種類の path programmable な TE 要素技術に加えて，それらを組み合わせて実現される *Network Function Chaining* (*NFC*) 技術に関して述べる．これらの技術は 1 節で示した伝統的な TE 手法より柔軟なトラフィック制御ことを目的として提案されたものである．本節で述べる要素技術群と提案手法を 3 節の要求事項に従って整理すると表 1 を得る．

4.1 Flow-based approach

この分類の要素技術の代表例として OpenFlow[6] や BGP Flowspec[7] が挙げられる．これらの要素技術では主に 5 tuples に基づきフローが識別されて core entity が path を構築する．データはフロー単位でフォワーディングルールが適応され，構築された path に沿って転送される．

OpenFlow では中央集権的なコントローラ・ノードから各スイッチに対して path を構成するルールをデプロイする．このような中央集権的な機構は Open Network 環境で展開するにあたってスケーラビリティ上の問題を生む．BGP Flowspec では BGP メッセージ中の Flowspec 属性の中に path を構成するためのルールを格納して，BGP に参加するノード間で自律分散的にそれらのルールが共有される．このため OpenFlow にみられるようなスケーラビリティ上の問題はないが，BGP という Open Network に渡って伝搬することを前提とした経路制御プロトコルの中にフロー単位でのポリシが伝搬されることから，そのポリシの適応の際には強力な validation 機構 [7] が存在する．

そのため Flowspec による TE には柔軟な path の構築に課題が残る．

4.2 Source routing

古くから IPv4 loose source routing[10] や IPv6 type0 header[11] に基づく source routing があるが，ここでは近年 IETF にて議論が盛んな segment routing[8] を挙げる．Segment routing ではネットワークをその部分集合である *Segment* に分割して，Segment を単位とした path を end entity が構築する．構築された path は Segment を単位とするラベルパスとして表現され，OSPF や IS-IS といった Intra Gateway Protocol (IGP) の経路制御プロトコルによってネットワーク内に伝搬される．データ転送は TCP/IP に基づく現在のフォワーディングとの整合性との矛盾がないよう設計されており，代表例としては IPv6 や MPLS を選択可能である．

Segment routing では path を構成するラベルパスの伝搬は IGP に基づくことが想定されている．このため Open Network 環境で segment routing を利用することは想定されていない．また end entity が構築した path を悪意あるものであるかを core entity が確認する validation 機構がない．このため意図的に小さな segment をもとに path を構築することで悪意ある path の構築が可能となる．これは旧来から source routing が抱える問題点である．

4.3 Network service ID tags

この分類の要素技術の代表例として Network service header (NSH)[9] が挙げられる．NSH は ingress ルータにおいてデータ・パケットの network header と transport header の間に挿入されて，egress ルータにおいてデータ・パケットから除去される．NSH には *service path header* などに core entity が構築した path の情報が含まれている．つまり ingress / egress ルータ間でどのノードを経由してどの network service を通過するかが示されている．

NSH は ingress / egress ルータ間の path 構築のみを範囲内としていることから Open Network 環境における end-to-end の path の構築を範囲外としている．また NSH では network header と transport header の間に独自のヘッダを追加するためのミドルボックスを配置する必要がある．このようなミドルボックスの配置は Open Network 中では end-to-end の到達性を阻害する要因となる．

4.4 NFC based on SDN/NFV technologies

これまで述べてきた 3 種類の要素技術を組み合わせて柔軟な TE 手法を提案する研究が SDN/NFV の分野では盛んである．例えば *SIMPLE*[12]，*MIDAS*[13]，*Cloud4NFV*[14] などが挙げられる．これらの提案では Service Function Chaining アーキテクチャ [15] に基づいた control plane と data plane を持つ．データは例えばフロー情報等によって *SFC classifier* により分類されて *Service Function Forwarder* (*SFF*) によって転送される．この分類に属するの提案は 4.1, 4.2, 4.3 の各小節の要素技術を組み合わせたものであるので，その組み合わせ方によって各要素技術毎の問題点を継承する．

5. vAS: Virtualized Autonomous System

5.1 想定する環境

図 1 に本稿が想定する環境を示す．個々の vAS は closed な overlay network として構成される．一方その underlay には既存の Open Network 環境に基づく *native network* が位置する．Native network は Open Network を構成する全ての AS が位置するのに対して，vAS を構成する overlay network にはその vAS に属する AS のみが所属する．vAS を構成する AS には end entity / core entity の双方が含まれる．vAS を構成する overlay network は native network のトポロジに矛盾しないトポロジを取るものとする．従って vAS 内の最短経路に従った path は native network の経路情報に基づき構成可能である．

AS を持たない application service provider (*stub service provider*) や vAS に所属しない native network の AS に所属する加入者 (*client*) は vAS に設置された *Point-of Presence* (*PoP*) に接続することで vAS に参加可能である．Stub service provider と client は接続する vAS への PoP をもつ AS に対して対価を払って vAS への接続性を購読する．

5.2 vAS を前提としたネットワーキング・モデル

vAS を前提とした end-to-end のネットワーキングは overlay network 上での *programmed-path-based networking* と native network 上での *shortest-path-based networking* の両者から構成される．両者を組み合わせることで柔軟な path programmability と Open Network で実現されているのと同等の end-to-end 到達性が両立可能である．

前者は vAS 内で共有された end entity / core entity のセマンティクスに基づいたネットワーキングのことである．Control plane では所属する entity 同士が共有すべきセマンティクスを広告してデータ転送の path を構成し，data plane では構成された path に従ってデータが転送される．

一方後者は既存の native network 越しの最短経路に基づいたネットワーキングのことである．Control plane では既存の Open Network を構成するためのシグナリングに加えて，vAS を構成する overlay network を構築するためのシグナリングと vAS に属さない stub service provider と client が vAS に接続する PoP を発見するためのシグナリングを担う．Data plane は既存の Open Network 環

Vol.2018-DPS-175 No.39
Vol.2018-MBL-87 No.39
Vol.2018-ITS-73 No.39
2018/5/25

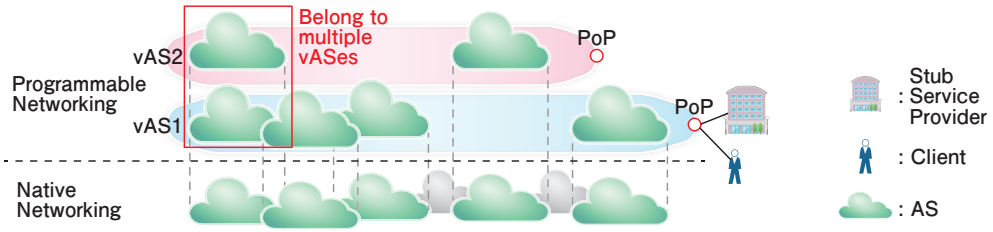

図 1　vAS を取り巻く環境

境における最短経路に従ったデータ転送を担う．このため vAS 上で構成された path に基づくデータ転送に失敗した場合も native network の data plane に基づくデータ転送に failover することができる．

5.3　vAS 上で共有されるセマンティクスのシグナリング

vAS に所属する entity 間は vAS で共有されるセマンティクスを広告するための動的プロトコルが動作する．このプロトコルでは広告されるセマンティクスを *application service class* とフローを構成する 5 tuples を併せた 6 つの要素と紐付けて広告できる．ここでの application service class とはフロー情報に紐付くことなく end entity のサービスを分類出来るクラスのことである．フロー情報に紐付けない理由は，例えば HTTP2 に見られるように，一つのフローの中に性質の異なる end entity のサービスが共存する場合があるからである．

広告されるセマンティクスは application service class と 5 tuples を構成する 6 つの要素の範囲内で集約することが出来る．例えば vAS に属する end entity に着目するとき，自身が展開する application service を特徴付けるためのセマンティクスをその application service class とサービスが動作するホスト群のネットワーク・アドレス及びポート番号と紐付けて vAS に広告することが出来る．

5.4　vAS に属する entity 同士のトラスト・モデル

vAS 内のシグナリングにより広告されたセマンティクスはお互い信頼できるものとする．vAS 内の entity は Public Key Infrastructure (PKI) に従ってお互いを認証することができ，広告元されたセマンティクスに付加された署名によってその完全性を検証可能とする．このような完全性の検証は Open Network 上で経路情報の真正性検証に用いられる BGPSEC/RPKI[16][17] によって実現可能である．

このような前提に立つとき vAS 内のシグナリングでセマンティクスを受信した entity はそのセマンティクスを信頼して accept し path を構築する．一方で実際のデータ転送時にその path に沿ったフォワーディング処理を実施するか (action) はその entity が独自に判断することが出来るものとする．例えばある entity が DDoS 攻撃を遮断するための *Source Remotely Triggered Black Hole*[18] の機能を持たせた path を構築した場合，その action を発動させるべきかを最終的に決定するのは action が適応される entity であるべきである．予め構築された path と矛盾した挙動をデータ転送時に実施する場合には path を構築した entity に対してその旨通知される．

6.　vAS を用いた通信モデル

vAS を用いた通信モデルを図 2 に示す．この図では vAS に所属しない AS の加入者である client と vAS に end entity として所属する application service provider との間での end-to-end の通信を想定している．また client と application service provider 間はクライアント・サーバモデルに従った通信を行うものとする．

まず client は native network を通じて利用したいサービスを展開する application service provider を発見し，両者で共通して利用できる vAS を negotiation する (図 2-1)．次に client は negotiation の結果特定された vAS の PoP の中から一つを選択して接続を試みる (図 2-2)．この図では client との間の native network 上での論理的距離が最も近い PoP に対して接続を試みている．PoP への接続に際しては，必要に応じて client の認証・認可の手続きが実施されて client の PoP への接続権限が確認される．Client と vAS の PoP との間のセッション確立後に vAS 上で application service provider が属する end entity への path を確立してその path に従ってデータを転送する (図 2-3)．仮に vAS 上で確立した path に沿ったデータ転送に失敗した場合は native network 上でのデータ転送に failover する．

7.　今後の開発方針

(i) vAS 内の signalling protocol の詳細設計と実装: vAS 内の entity 間で共有するセマンティクスの詳細な設計を進めて実装する必要がある．vAS には複数の AS が所属しており AS 間に跨がったシグナリングが必要となることから，基本的には BGP に基づくプロトコルにて設計し実装する予定である．

(ii) TLS に変わるセキュリティ・モデル: end entity が TLS に従って transport payload を暗号化してデータ転送

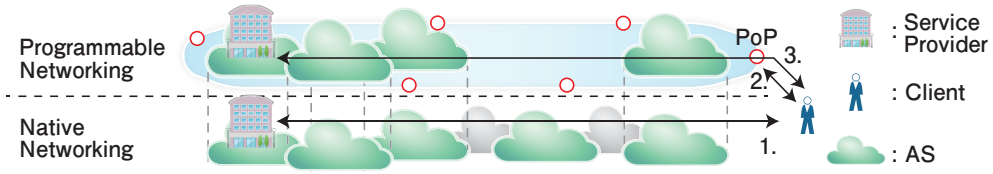

図 2　vAS を前提とした通信モデル

する場合には application service class といったフロー情報を超えての end entity のサービス特性の把握が困難となる．このため vAS 上では application payload の暗号化に基づくセキュリティ・モデルの構築が必要と考える．このようなセキュリティ・モデルでは end entity 上で動作するサービスの application header は vAS に属する entity から識別可能となり application service class の判別が可能となる．また end entity が実際に秘匿したい情報を含んだ application payload については vAS 上でも依然として暗号化される．一方で TLS のセキュリティ・モデルに従った暗号化も vAS 上で必要に応じて利用可能である．この場合 vAS 上でのデータ転送時の path の識別はフロー情報によってのみ実施される．

(iii) DevOps に基づく vAS の運用モデル: vAS に属する entity は network service provider に限らず様々な application service provider も含まれることから，vAS は software-defined な運用モデルに基づくべきである．その際，immutable infrastructure[19] や micro-service architecture[20] といった *DevOps*[19] の考え方に基づいた vAS の運用方法を構築する必要がある．

8. まとめ

本稿では，Open Network 環境を対象とした end entity / core entity の両者でお互いのセマンティクス共有を前提とする TE 手法を提案する．共通のセマンティクスを共有する entity 群は vAS と呼ばれる closed な overlay network に所属する．vAS では end entity と core entity の双方のセマンティクスを考慮した end-to-end の通信路が構成され，その通信路に従ってパケットは転送される．vAS 上でのパケット転送は既存の Open Network 環境のパケット転送に failover 可能である．本稿では vAS を前提としたネットワーキング・モデル，vAS 内のトラスト・モデル，vAS を用いた通信モデルの考察を通して vAS による TE 手法の設計を検討した．

参考文献

[1] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus. Requirements for Traffic Engineering Over MPLS. RFC 2702, *IETF*, 1999.
[2] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209, *IETF*, 2001.
[3] C. Hopps. Analysis of an Equal-Cost Multi-Path Algorithm. RFC 2992, *IETF*, 2000.
[4] C. Villamizar. OSPF Optimized Multipath (OSPF-OMP). Internet-Draft draft-ietf-ospf-omp-02, *IETF*, 1999.
[5] D. Xu, M. Chiang, and J. Rexford. Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering. *IEEE/ACM Trans. Netw.*, Vol. 19, No. 6, pp. 1717–1730, 2011.
[6] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, Vol. 38, No. 2, pp. 69–74, 2008.
[7] P. Marques, N. Sheth, R. Raszuk, B. Greene, J. Mauch, and D. McPherson. Dissemination of Flow Specification Rules. RFC 5575, *IETF*, 2009.
[8] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir. Segment Routing Architecture. Internet-Draft draft-ietf-spring-segment-routing-15, *IETF*, 2018.
[9] P. Quinn, U. Elzur, and C. Pignataro. Network Service Header (NSH). RFC 8300, *IETF*, 2018.
[10] INTERNET PROTOCOL. RFC 791, *IETF*, 1981.
[11] R.M. Hinden and S.E. Deering. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, *IETF*, 1998.
[12] Z.A. Qazi, C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu. Simple-fying middlebox policy enforcement using sdn. In *Proc. of ACM SIGCOMM '13*, pp. 27–38, 2013.
[13] A. Abujoda and P. Papadimitriou. Midas: Middlebox discovery and selection for on-path flow processing. In *Proc. of 7th Int'l Conf. on COMSNETS*, pp. 1–8, 2015.
[14] F. Callegati, W. Cerroni, C. Contoli, and G. Santandrea. Dynamic chaining of virtual network functions in cloud-based edge networks. In *Proc of IEEE NetSoft '15*, pp. 1–5, 2015.
[15] J.M. Halpern and C. Pignataro. Service Function Chaining (SFC) Architecture. RFC 7665, *IETF*, 2015.
[16] M. Lepinski and K. Sriram. BGPsec Protocol Specification. RFC 8205, *IETF*, 2017.
[17] R. Bush and R. Austein. The Resource Public Key Infrastructure (RPKI) to Router Protocol, Version 1. RFC 8210, *IETF*, 2017.
[18] W. Kumari and D. McPherson. Remote Triggered Black Hole Filtering with Unicast Reverse Path Forwarding (uRPF). RFC 5635, *IETF*, 2009.
[19] Kief Morris. *Infrastructure As Code: Managing Servers in the Cloud*. O'Reilly Media, Inc., 1st edition, 2016.
[20] A. Balalaie, A. Heydarnoori, and P. Jamshidi. Microservices architecture enables devops: Migration to a cloud-native architecture. *IEEE Software*, Vol. 33, No. 3, pp. 42–52, 2016.