# Strategic path diversity management across internet layers

Ho Dac Duy Nguyen


LABORATOIRE D'INFORMATIQUE DE PARIS 6

# STRATEGIC PATH DIVERSITY MANAGEMENT ACROSS INTERNET LAYERS

Advisor:
**Prof. Stefano SECCI**

Doctoral Dissertation of:
**Ho Dac Duy NGUYEN**

2018


Thèse

Présentée pour obtenir le grand de docteur de la Sorbonne Université

Spécialité: Informatique

Ho Dac Duy NGUYEN

# STRATEGIC PATH DIVERSITY MANAGEMENT ACROSS INTERNET LAYERS

Soutenue le 22 octobre 2018 devant le jury composé de :

| | | |
|---|---|---|
| Rapporteurs: | Prof. Fabio MARTIGNON | Univ. Bergamo. |
| | Prof. Sidi-Mohamed SENOUCI | Univ. Bourgogne. |
| Examinateurs: | Dr. Geraldine TEXIER | Institut Mines-Télécom. |
| | Dr. Luigi IANNONE | Institut Mines-Télécom. |
| | Prof. Maria POTOP BUTUCARU | Sorbonne Université. |
| Directeur de thèse: | Prof. Stefano SECCI | Cnam. |


LABORATOIRE D'INFORMATIQUE DE PARIS 6

# STRATEGIC PATH DIVERSITY MANAGEMENT ACROSS INTERNET LAYERS

**Author:** **Ho Dac Duy NGUYEN.**

**Defended on October 22, 2018, in front of the committee composed of:**

| | | |
|---|---|---|
| **Referees:** | **Prof. Fabio MARTIGNON** | **Univ. Bergamo.** |
| | **Prof. Sidi-Mohamed SENOUCI** | **Univ. Bourgogne.** |
| **Examiners:** | **Dr. Geraldine TEXIER** | **Institut Mines-Télécom.** |
| | **Dr. Luigi IANNONE** | **Institut Mines-Télécom.** |
| | **Prof. Maria POTOP BUTUCARU** | **Sorbonne Université.** |
| **Advisor:** | **Prof. Stefano SECCI** | **Cnam.** |

*To my family!*

# Abstract

We present in this thesis novel routing protocols able to take into consideration strategic aspects when deciding which path among many to take, and that at the Internet communication network scale. The standpoint adopted in this study is that novel routing architectures are exposing a higher path diversity to networks and applications so that networks and applications can be made capable to more intelligently select their strategy when selecting toward which path to forward their traffic, taking into consideration operational costs as well as performance goals. We present enhanced behaviors to the decision-making core of three routing protocols, the Border Gateway Protocol (BGP), the Locator/Identifier Separation Protocol (LISP) and, at a minor extent, the Multipath TCP (MPTCP) protocol. For each protocol framework we present how routing strategies can be computed, selected and actually operated by real systems, also applying concepts from non-cooperative game theory, evaluating the impact of the routing solutions in terms of operational costs and network performance. The thesis adopts an experimental methodology willing to experiment and evaluate proposals via realistic simulations or actual implementation and observation of real systems. Most of the results are made reproducible by open sourcing the corresponding code.

# Résumé en Langue Française

Nous présentons dans cette thèse de nouveaux protocoles de routage capables de prendre en compte des aspects stratégiques lorsqu'il s'agit de choisir le chemin à emprunter et ce à l'échelle du réseau de communication Internet. Le point de vue adopté dans cette étude est que les nouvelles architectures de routage donnent aux réseaux et aux applications une plus grande diversité de chemins, ce qui leur permet de choisir plus rationnellement leur stratégie lorsqu'ils décident le chemin à suivre pour transférer leur trafic, en tenant compte des coûts opérationnels ainsi que des objectifs de performance. Nous présentons des comportements améliorés au noyau décisionnel de trois protocoles de routage, le protocole BGP (Border Gateway Protocol), le protocole LISP (Locator / Identifier Separation Protocol) et, dans une moindre mesure, le protocole MPTCP (Multipath TCP). Pour chaque cadre protocolaire, nous présentons comment les stratégies de routage peuvent être déterminées, sélectionnées et réellement exploitées par des systèmes réels, en appliquant également les concepts de la théorie des jeux non coopératifs, en évaluant l'impact des solutions de routage en termes de coûts opérationnels et de performances réseau. La thèse adopte une méthodologie expérimentale permettant de tester et d'évaluer les propositions via des simulations réalistes et la mise en œuvre et l'observation réelles de systèmes réels. La plupart des résultats sont reproductibles grace à la publication du code source.

# Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| AS | Autonomous System. |
| BGP | Border Gateway Protocol. |
| BLEST | BLocking ESTimation scheduler. |
| CDF | Cumulative Distribution Functions. |
| DAPS | Delay-Aware Packet Scheduling. |
| DNS | Domain Name Service. |
| EC | Egress Control. |
| ECF | Earliest Completion First. |
| EID | Endpoint IDentifier. |
| ETR | Egress Tunnel Router. |
| FIFO | First In First Out. |
| IETF | Internet Engineering Task Force. |
| IGP | Interior gateway protocol. |
| ITR | Ingress Tunnel Router. |
| LISP | Locator/Identifier Separation Protocol. |
| LISP-EC | LISP Egress Control. |
| LISP-TE | LISP Traffic Engineering. |
| LowRTT | Lowest-RTT-First. |
| MITM | Man In The Middle. |
| MPTCP | Multipath TCP. |

| | |
|---|---|
| MR | Map Resolver. |
| MS | Map Server. |
| MSFD | Mapping Service Function Discovery. |
| OLB | Optimal Load Balancing. |
| OS | Operating System. |
| OSPF | Open Shortest Path First. |
| OTIAS | Out-of-order Transmission for In-order Arrival Scheduling. |
| PEMP | Peering Equilibrium MultiPath routing. |
| ProgMP | Programmable MultiPath TCP Scheduling. |
| RLOC | Routing LOCators. |
| RTT | round-trip-time. |
| SDWAN | Software-Defined Wide Area Network. |
| TCP | Transmission Control Protocol. |
| TE | Traffic Engineering. |
| VM | Virtual Machines. |
| WLB | Weighted Load Balancing. |
| xTR | ITR/ETR. |

# Chapter 1

# Introduction

Communication networks have been shaping the rapid evolution of the digital society. Since the commercialization of the Internet access, in 1992, a gradual yet irreversible liberalization of telecommunication markets took place, bringing pressure on Internet stakeholders, with high competition affecting investments: interconnection facilities became scarce, expensive and contended resources, under a freed and naturally evolving Internet ecosystem. Henceforth, selfish Internet traffic routing and interconnection policies came into play, inevitably leading to a Tragedy of the Commons in Internet-working, supported by standards: unsecured and unreliable protocols, mostly due to their naivety in shared physical or logical resource management.

To avoid bad Internet commons sharing practices and hence Internet infrastructure bloat, novel communications network protocols such as those proposed in [1, 2, 3] at the different interactive decision-making layers of the Internet architecture are proposed. These solutions follow a common game theory based decision-making pattern and introduce a novel approach for coordinated routing equilibrium computing by proposing a potential game modeling for routing problems, and related equilibrium selection strategies.

The objective of this thesis is going beyond game-theoretical modeling of strategic routing and forwarding situations at the state of the art presented in [1] [2] [3], with a practical and implementation-driven perspective, and modeling perspective as well. Our motivation during the thesis was to take the hands on theoretical proposals at the state of the art evaluated only through simple simulations, implement them in real systems, enhance them based on real systems experience, and possibly complete them going beyond basic ideas and modeling choices.

Firstly, we explore the proposed idea of coordinated Internet routing acting at the

network layer (based on the Internet Protocol, IP), and more precisely tuning the decision processes of the Border Gateway Protocol (BGP) [4] and of the Locator/Identifier Separation Protocol (LISP) [5]. We could leverage on efficient available open source implementations for both BGP [6] and LISP [7] routers. These implementations not only allow us to perform a comprehensive performance evaluation over real testbeds, but also provide valuable inputs to improve the proposed system design and to better adapt with the practical network environments. In this direction, we worked at a formalization of the cross-layer interaction in a reference - yet long-term vision framed - Internet framework where both BGP and LISP equilibrium routing practices are operated.

In the second part of the thesis, we explore how similar coordination strategies could take place at the transport layer and in particular connection-oriented multipath capable transport protocols such as Multipath Transmission Control Protocol (MPTCP) [8]. Our investigations first followed a measurement approach around a particular use-case in mind, i.e., the one about using multipath path at the MPTCP layer to increase path diversity for the sake of Internet confidentiality. Then, we investigated how explicit decisions on traffic scheduling could be taken going beyond the default approach of decisions based on features discovered through in-band signaling. We present two promising directions, one acting at the socket buffer management level, and one acting at scheduler behavior directly, positioning such approaches with respect to recent works at the state of the art.

The rest of the manuscript is organized as follows:

- In Chapter 2 we review necessary background on the target routing protocol frameworks, and we summarize existing work on game-theoretic modeling of Internet routing problems.
- Chapter 3 presents our experimental experiences and the proposed enhancement to the core-network peering equilibrium routing framework integrated to BGP.
- Chapter 4 presents our experimental experiences and the proposed enhancement to the edge-network load-balancing framework integrated to LISP.
- In Chapter 5 we design a cross-layer routing coordination framework meant to involve edge and core networks.
- Chapter 6 explores how by pushing the multipath routing decision to the transport layer one can enhance Internet connection confidentiality, by means of a measurement campaign.

- Chapter 7 explores the MPTCP explicit scheduler design problem, drawing two investigated directions and presenting the state of the art on the matter.
- Chapter 8 concludes the manuscript while drawing open future directions.

We complete the manuscript with a summary on the open source software contributions related to the described works and with the corresponding list of publications.

# Chapter 2

# Background

In this chapter, first we briefly introduce different multipath routing solutions at the state of the art, with a particular focus on those leveraging on game theory concepts and targeting application to Internet-scale communications.

## 2.1 BGP, LISP, MPTCP in a nutshell

From a graph analysis perspective, the Internet results in a highly connected scale-free graph [9], with multiple paths connecting two given end points, more than 60000 autonomous networks and millions of routers.

Traffic transmission over the Internet network is determined by both the routing protocols adopted by routers and the connection management protocol adopted by end-point. About the former, the current practice is a single-path routing protocol, the Border Gateway Protocol (BGP) [4]. The restriction to use a single path is given by the standard, despite attempts exist to extend it to integrate a multipath mode [10], with some industrial implementations. Disposing of multiple paths allows for enriching the domain of strategies available to network nodes when deciding over which path to send the traffic. About the latter aspect, i.e., host-based connection management protocol, the legacy is mostly marked by the Transmission Control Protocol (TCP), which is undergoing a huge redesign, referred to as Multipath TCP (MPTCP) [8], to manage concurrent interfaces and related paths. Additional protocols and networks architectures, such as LISP (Locator-Identifier Separation Protocols) [5] and Software-Defined Wide Area Network (SDWAN) solutions, also allow managing multiple paths at the Internet scale. The addition of such a path diversity to Internet routing and connection management protocols leads to a novel decision-making framework for Internet routing.

When it comes to routing and opening connections through a network of this type,

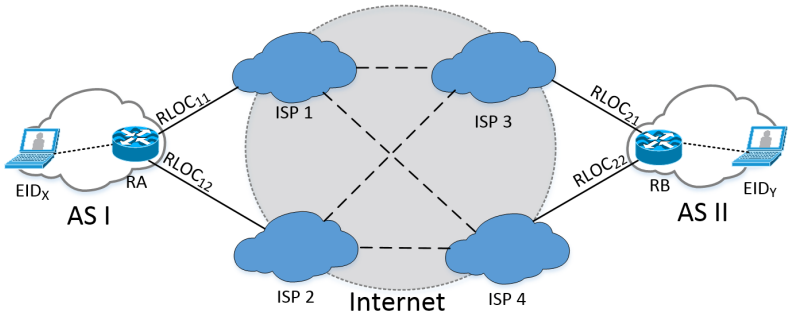



Figure 2.1: LISP communication scenario example

the key point is how to manage congestion across shared bottleneck links in the resulting competitive framework where network nodes are willing to get the fastest path. For the reader not used to the above mentioned Internet protocols, we provide a synthetic description in the following.

### 2.1.1 Border Gateway Protocol (BGP)

As already mentioned, BGP selects one single path toward every Internet network destination. It works as an enriched distance-vector protocol, transmitting to a given neighbor peer the path information towards a given destination; the transmitted path is selected as the best one among the path vectors sent by the other neighbors. The length of the path works as one of the metrics used by the BGP decision process. Indeed, multiple other metrics as used. In particular, a local preference metrics, locally configured, is given higher priority than the path length to influence outgoing traffic path selection. Moreover, a multi-exit discriminator metrics is given lower priority than the path length to suggest to a neighbor own preferences over incoming traffic path. The local-preference and the multi-exit discriminator metrics allow going beyond standard shortest path routing for Internet routing. Their configuration is a means to implement advanced routing policies at the IP layer.

### 2.1.2 Locator/Identifier Separation Protocol (LISP)

Differently from the legacy flat routing structure given by BGP, LISP involves two independent addressing spaces: one for the Routing Locator (RLOCs) and other for the Endpoint Identifiers (EIDs), the latter being mapped to RLOCs by a mapping system. The RLOC addresses are attached to LISP router interfaces, i.e. border routers that connect a LISP site to the Internet. While the RLOC addresses are globally routable, the EID addresses can stay routable only within the local LISP site.

Figure 2.1 depicts a basic LISP communication scenario, where traffic is sent from host $EID_X$ in AS I, to host $EID_Y$ in AS II. RA and RB are the border routers of AS I and AS II, respectively. $RLOC_{11}$, $RLOC_{12}$ are the network interfaces connecting RA with two upstream providers, ISP 1 and ISP 2, respectively. In the LISP jargon, RA is a tunneling router (xTR), while $RLOC_{11}$ and $RLOC_{12}$ are well-known as the routing locators. Similarly, $RLOC_{21}$ and $RLOC_{22}$ are the two routing locators for the tunneling router RB and more precisely AS II. Traffic from $EID_X$ to $EID_Y$ first reaches RA, which looks up its mapping cache to find the corresponding destination RLOCs, the routing locators in AS II that are responsible for routing traffic toward $EID_Y$. Assuming that the mapping for $EID_Y$ is already installed in the RA mapping cache, and $RLOC_{21}$ is the preferred locator, an IP-UDP tunnel is then established, encapsulating all the packets originated from $EID_X$ with an outer IP header with $RLOC_{21}$ as destination address. The source address is selected from the routing table as the best outgoing interface toward $RLOC_{21}$ from RA, i.e., $RLOC_{11}$ or $RLOC_{12}$: this decision is taken by the underlay IP routing protocol, e.g. BGP or an internal gateway protocol or the default IP configuration. Hence the traffic from $EID_X$ is forwarded to $RLOC_{21}$, RB decapsulates the received packets and forwards them internally according to destination address specified in the inner IP header.

### 2.1.3 Multipath Transmission Protocol (MPTCP)

MPTCP extends TCP and allows fragmenting a data flow from a single connection into multiple paths (subflows TCP) [8, 11], as illustrated in Figure 2.2. At the application layer, a connection appears as a normal TCP connection. At the network layer, each subflow looks like a regular TCP flow whose segments carry in their header a new type of TCP option [8]. The protocol improves the performance offered by a single flow and makes the connection more reliable using concurrent and redundant paths.

The initial TCP connection handshake carries an option, the MP_CAPABLE option, to enable MPTCP capability discovery and subflow creation. The handshake can carry additional information, such as a cryptographic key employed to authenticate the end-hosts and set up new subflows [8]. The establishment of additional subflow may employ also a token and random numbers (nonces), to prevent replay attacks on the authentication method. Further, an additional address identifier may be employed to identify the source IP address of a packet. Hence, even if the IP header has been changed by a middlebox (e.g. NATs, firewalls), end-hosts can identify an address without any doubt or ambiguity.

MPTCP can overcome some weaknesses inherent to TCP, achieving (i) a *greater throughput*, (ii) *higher reliability*, and (iii) *higher confidentiality*. Indeed, a multipath

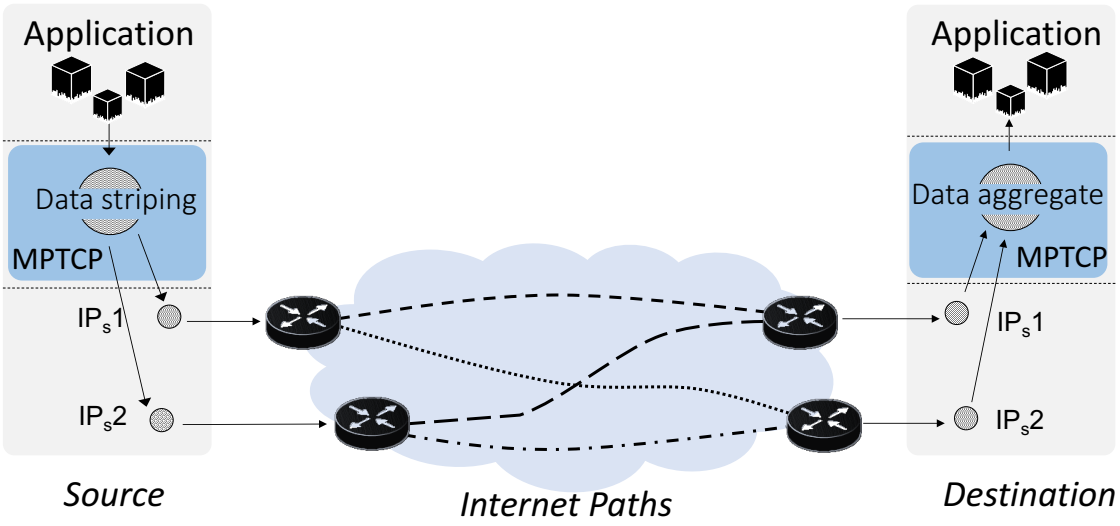



Figure 2.2: Multipath TCP Connection: Overview

connection can improve the throughput aggregating bandwidth over different paths by concurrent data transmission across all available paths. Moreover, a multipath connection can quickly overcome one path failure by sending data to another available path, increasing the data delivery reliability [12]. Finally, fragmenting data flow across different subflows makes complete connection interception difficult because attackers would need to capture the transmitted content through all the subflows to build the content.

Therefore, MPTCP can provide a greater level of confidentiality than a regular TCP transmission if the subflows of a connection are routed along disjoint paths: the higher the level of disjointedness, the higher the confidentiality guarantee, and furthermore the higher the level of robustness against such attacks. The goal of this work is to precisely quantify the level of robustness in use-cases where MPTCP is adopted not (only) to improve communication performance or reliability, but (also) to improve confidentiality. When addressing this aspect, router-level path disjointedness can be considered as being too weak in particular against AS-level traffic capturing and route hijacking. This is the reason why we focus instead on a larger scale of path disjointness, i.e., AS-level path disjointedness, which do make sense in practical scenarios as elaborated here after. Running an analysis on an even larger scale than AS-level scale (e.g., regional or country level) would likely be either infeasible or not sufficiently realistic.

## 2.2 From competitive routing to coordinated routing

When it comes to routing and opening connections through a network of this type, the key point is how to manage congestion across shared bottleneck links in the resulting

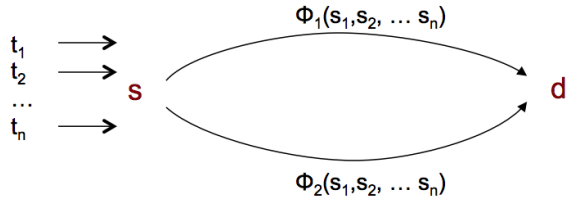



Figure 2.3: Competitive routing (passive nodes)

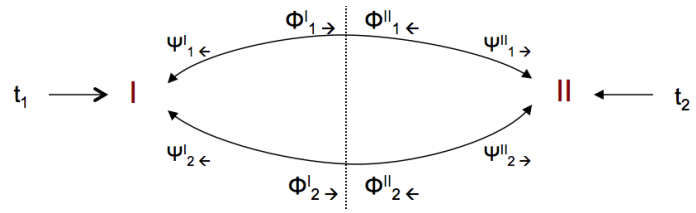



Figure 2.4: Coordinated routing (active nodes)

competitive framework where network nodes are willing to get the fastest path.

Deriving from the seminal work [13], the classical competitive routing situation is depicted in Figure 2.3: a number of sources have to send traffic by a same common gateway node that is connected with parallel direct links (two links in Figure 2.3) to a common destination. Each source $i$ has to decide how much of its traffic $r_i$ to send over which link $l$, i.e., ${f^i}_l$. Moreover, let each source be aware of the link cost function, i.e., ${l_k}({f^i}_l)$, that is convex, monotonically increasing with the overall load sent on the link: the more the load on a link, the higher the routing cost suffered by the sources transmitting on the link.

In [13] it is proven that a pure-strategy routing equilibrium always exists, i.e., it is possible to decide in a stable manner how much traffic to send on which link so that each network node has no unilateral incentive to deviate from the equilibrium solution. In the specific case where there are intermediate nodes along the way to destination, the existence of equilibrium is also guaranteed but only for very specific cost functions.

Several works followed on the topic. A common contribution is to define self-enforcement protocols to decrease the so-called price-of-anarchy (PoA) of the equilibrium solution, i.e., the gap between the equilibrium profile and the social optimum profile, as for example done in [14, 15]. A useful application of PoA-guided routing system design is presented in [16]: network design can be done in such a way that each network configuration is associated with the expected equilibrium routing solution, so that the best possible equilibrium routing solution guides network design choices compliant also with provider's goals. Moreover, repeated game variations of the competitive routing game are quite present in the literature. The common assumption is that a repeated interaction can more easily guarantee convergence and the efficiency of self-enforcement algorithms aiming at decreasing the PoA. Common variations consider various utility functions, which can be made sensible

to interference adjustments as in [14] for wireless network situations, to destination server states as a function of the load as in [17], or to connection-level flow-control throughput and latency states as in [18].

Canonical competitive routing works are therefore particularly appropriate for applications where there is a common passive destination among multiple sources that share a common communication channel or subpath. When instead the destination is not passive but it is one among the players (see Figure 2.4), as in the targeted reference peering AS scenario, the competitive routing situation is fundamentally different. When nodes in competition are both active and exchange traffic with each other, models such as those in [13]-[17] are not directly applicable and implementable in real systems. Another IP network requirement that is not easily met by such competitive routing approaches is that the IP link cost setting and routing decision are, in practice, two different decisions, only lightly correlated to each other, if not completely independent for some specific usages.

In Figure 2.4, both nodes (I and II) are source and destination of traffic: they are autonomous decision-makers and they send traffic to each other using parallel links. The routing costs are, this time, directional costs, as traffic goes from I to II and from II to I; hence for each link and each node there are two routing costs. As such, nodes have to coordinate on the load-balancing over parallel links and the competitive routing situation can be seen as a coordinated routing problem. In the literature, approaches can be classified as negotiation-based approaches as in [19, 20, 21], and game-theoretic approaches as in [22, 1]. The former approaches target the conception of an inter-domain routing protocol supporting route proposal and acceptance/rejection signaling; in [19] a route negotiation best-reply approach is adopted, built upon bidirectional costs. In [22], instead of explicit negotiation it is proposed to exchange routing costs using in-band signaling channels; as resolution method, they propose to sum up the cost of the two players, to sort the corresponding path alternatives and then to select the shortest path. Their argument in favor of this approach, rather than a non-cooperative game equilibrium computation approach, is that the latter is NP-hard. However, in a later study [1], it is proven that preserving the unilateralism of the routing cost components as in Figure 2.4 - whose value may be on different scales for different Internet networks (Autonomous Systems, ASes) - the resulting non-cooperative game is a special game such that an equilibrium always exists and it can be computed in a polynomial time.

## 2.3 Peering equilibrium multipath routing

Peering Equilibrium Multipath (PEMP) routing [1] was proposed as a solution to enhance routing stability and bilateral cost across inter-AS peering links. It was specified

so with marginal modifications to the current inter-domain routing protocol (BGP). More precisely, the modifications are as follows:

- *BGP signaling*: in standard BGP, the Multi-Exit Discriminator (MED) attribute can be used to suggest to an AS neighbor, connected via multiple inter-AS links, an entry point to its own AS; the MED value is typically set to the interior gateway protocol routing cost toward the destination, so that it suggests a ranking over multiple inter-AS links for a given destination IP prefix. In PEMP, it is specified to use the MED as a coordination signaling media; it is coded to transport not only the incoming routing cost, but also the outgoing routing cost.

- *BGP decision process*: when multiple routes to a same destination via a same AS exist and are considered equivalent with respect to local preference and AS hop count, the least MED rule is used to route toward the downstream AS preferred exit point. With PEMP, the least MED rule is changed so that it decides the best route or the multiple routes that correspond to the PEMP equilibria. The game components are built using the ingress/egress routing costs (four for each link, as in Figure 2.4) exchanged via the MEDs.

PEMP models the inter-AS bilateral routing decision process as a 2-player non cooperative game; the two ASes act as rational players - referred to as players $I$ and $II$ - and the game strategy sets - $X$ and $Y$ - are the available peering links toward a given destination IP network. A combination of choices forms a strategy profile $(x, y) \in X \times Y$; every profile associates with a pair of unilateral payoff values that reflect the benefit of AS players associated with the corresponding routing decision. The payoff of each participant - $f(x, y)$ and $g(x, y)$, respectively - is a cost defined by the sum of directional unilateral cost components. For a given AS, the egress cost component - $\phi_I(x)$ and $\phi_{II}(y)$ respectively - depends on the strategy selected by the AS itself, while the ingress cost - $\psi_I(y)$ and $\psi_{II}(x)$ - is determined by the choice of its neighbor. Hence $f(x, y) = \phi_I(x) + \psi_I(y)$ and $g(x, y) = \phi_{II}(y) + \psi_{II}(x)$.

Therefore, the resulting game $G(X, Y; f, g)$ is such that a profile indicates a link to use for each of the two players, for each of the two traffic flows from one network to the other. The two flows are considered to be equivalent, where equivalence may not strictly mean the same bit-rate, but also uneven bit-rates (as it happens in content provider to transit provider peering agreements) and even a more generic equivalence definition. This implies that at least two distinct destination IP prefixes are associated to a routing game (one for each AS), and that at most each AS associates a set of IP prefixes to the routing game. The way to segment different routing games decisions can rely on the usage of the 'BGP community' marking, which can be captured by the BGP decision process.

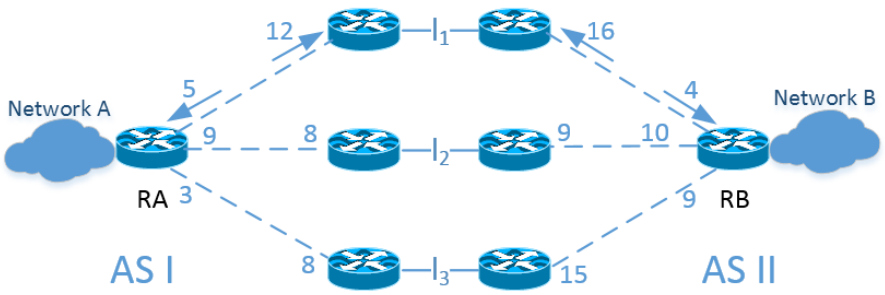



Figure 2.5: Routing setting

Table 2.1: Example game form

| I\II | $l_1$ | $l_2$ | $l_3$ |
|---|---|---|---|
| $l_1$ | (17,20)$^{11}$ | (21,13)$^{4}$ | (15,19)$^{10}$ |
| $l_2$ | (13,26)$^{7}$ | **(17,19)**$^{0}$ | (11,25)$^{6}$ |
| $l_3$ | (13,25)$^{7}$ | **(17,18)**$^{0}$ | (11,24)$^{6}$ |

Under complete information sharing, both ASes can compute the same equilibrium solution. $G(X,Y;f,g)$ is a potential game, i.e., each profile $(x,y)$ can be associated with a potential value $P(x,y)$ such that the difference in potential values between two profiles differing from an unilateral strategy move is the same independently of the other player strategy, i.e. $P(x,y)-P(x',y)=P(x,y')-P(x',y'), \quad \forall x,x' \in X, \forall y,y' \in Y$. In potential games, the minimum potential profile corresponds to a Nash equilibrium and always exists. Moreover, as proven in [1], for $G$ all Nash equilibria always correspond to a potential minimum, which is not true for the general case. This property makes PEMP routing attractive toward realistic implementations.

It should be noted that by letting the routing decision to follow the PEMP equilibrium solution, the peering ASes reach a strategically stable routing state such that no single AS has an incentive to change its routing decision.

An example is given in Figure 2.5. AS I and AS II interconnect with each other via three peering links: $l_1$, $l_2$ and $l_3$. As a result, router RA in AS I has three options for routing traffic from source network A to destination network B. Similarly, the same set of strategy is also available at router RB in AS II. For each intra-domain path connecting customer's network with border router, there are two internal routing costs: (a) an ingress cost represents the payoff when incoming traffic from peering AS flows on that path and (b) an egress cost indicates the payoff when forwarding packets to peer via that path. The

corresponding game form is given in Table 2.1: it summarizes all the possible outcomes of the routing game built from the above topology, it also includes the payoff and potential value of each profile.

For instance, profile ($l_3$,$l_2$) has a payoff value of $(17, 18)$ in which 17 is the sum of 8 and 9, that are, respectively, the routing costs at AS I when routing outgoing traffic via $l_3$ and receiving incoming traffic from $l_2$.

The profiles ($l_2$,$l_2$) and ($l_3$,$l_2$) are in the Nash set. When there are multiple equilibria, if there exists a Pareto-superior one, it can be preferred as an implicit coordination rule of thumb. Otherwise, in general, load-balancing can be performed on the equilibrium profiles (as further elaborated in the next chapter).

It is worth noting that, in the provided example, the routing outcome is the same as early exit (hot potato) routing, which shows that the provided framework is correctly modeling the current interconnection policies; more generally, this situation manifests when multiple profiles with the same minimum potential exists.

Relying on the IGP routing cost to make routing decisions, PEMP faces the same challenge of routing instability when transient failure occurs in the intra-domain network that legacy BGP routing faces. PEMP circumvents this problem by taking into account the IGP path cost variation when deciding which profiles can eventually be considered in the routing equilibrium solution. A profile $(x, y)$ is selected when it has potential within the minimum potential plus a threshold $\tau$ whose value is derived from the IGP path cost variation due to intra-AS link failures. Indeed, whenever a link failure happens, the costs for routing traffic across selected paths can increase. Consequently, the potential values $P(x, y)$ are recalculated, and new routing decision is made to adapt with such path cost deviation. By determining a proper threshold $\tau$, the network operator can anticipate routing variations caused by transient failures and hence select robust equilibrium routing solutions.

More precisely, the potential threshold is calculated by each peer relying on an exchange (also via the MED attribute) of global directional path cost error computed as a function of link failures that could manifest at each side, taking the maximum among the minimum best path cost variations.

In [1] it was further proposed to add a performance component to the routing game so as to allow influencing the routing decision also taking into consideration performance aspects. The overall game can therefore be decomposed as $G = G_{cost} + G_{perf}$, where $G_{cost}$ is the game already described above and $G_{perf}$ is a game of pure externality with peering link congestion functions. As the latter is pure externality, the composed game remains a potential game.

## 2.4 LISP traffic engineering

Internet protocols offering inter-domain multihoming traffic engineering (TE) capabilities can be classified into two major categories: host-centric and network-centric solutions. In the former one, the capability to decide source gateway for outgoing packets relies on local TE or the scheduling policies defined at individual hosts, as it can be done with Site Multihoming by IPv6 Intermediation (SHIM6) [23], Host Identity Protocol (HIP) [24], Multipath TCP (MPTCP) [24].

With host-centric approaches, the selection of outgoing interface can be a purely local decision, or the result of a negotiation between end-points, possibly passing via a server as proposed in [25]. When it is not a purely local decision, intensive signaling between the end-hosts and/or the server is usually required in the host-based protocol. Another drawback of host-based protocols is that, in order to influence the egress network exit point selection when multiple ones are presented, forms of source-specific routing, e.g., [26], are needed to follow ingress filtering policies implemented at upstream providers [27].

In network-centric solutions, traffic engineering mechanisms are defined and operated at the border router level and are made transparently with respect to the end systems; the aforementioned host-centric constraints therefore no longer exist. With such protocols, an end-point identifier (EID) is assigned to one or multiple routing locators (RLOCs), by means of a control-plane. Among the network-centric locator-identifier separator protocols proposed in the literature, LISP [5] (already explained synthetically in the beginning of this chapter) is the one that has been standardized since a decade, and undergoing industrial adoption for network multihoming. In the following, we synthetically present the LISP traffic engineering capabilities, and how to leverage on them to perform multipath equilibrium routing.

Traffic engineering support in LISP relies on two metrics that are assigned to RLOCs and distributed by the mapping system: the priority and weight metrics. When multiple RLOCs exist for a LISP EID prefix, the best priority one is preferred (i.e., least priority cost metric value), and in case of equal priority, traffic is distributed among them in proportion to their weight metric. The usage of the RLOC metrics is often referred to as LISP-TE in the literature. By regulating the EID-to-RLOC mappings that a LISP site (and its xTRs) registers with the LISP mapping system, then distributing to other LISP sites transmitting traffic to it, the LISP site can control how traffic enters in its network from the LISP-capable Internet.

Considering the scenario in Figure 2.1, AS I may have local preference on its default inbound ISP, e.g. ISP1 because less expensive or with better performance. To express that policy, RA can register its mapping entry so that both $RLOC_{11}$ and $RLOC_{12}$ are

announced as the routing locators for $EID_X$, but with a priority cost metric value for $RLOC_{11}$ set to a lower value than the one assigned for $RLOC_{12}$, the backup locator. This mapping is distributed by the mapping system (in a pull mode), and then employed by all other LISP sites that send traffic to $EID_X$, therefore via $RLOC_{11}$ and ISP1.

Despite LISP provides inbound TE capabilities, it does not offer outbound control features, i.e., which source RLOC to use when sending traffic to a destination RLOC. From TE perspective, this can be seen as a limitation, and could also lead to override destination network RLOC preferences if opposed to local ones as argued in [2]. Supposing a way to support egress control in LISP is made available, the resulting situation is strategically comparable to the PEMP one, which is the reason why the authors in [2] developed a similar game-theoretic framework to determine which paths to select and at which load-balancing ratio.

Let us present the work in [2] as it is later adopted in the manuscript. Thus, under the hypothesis that two LISP networks communicate with equivalent traffic volumes over the two directions, the LISP routing game consists in selecting the RLOC-to-RLOC path corresponding to a routing equilibrium solution that strategically takes into consideration the preferences of both parties on both inbound and outbound routes. To be more precise, applying that traffic engineering policy the routing interaction between LISP networks is modeled as a non-cooperative game. In which the two LISP networks are rational players with the strategy set consisting of RLOC-to-RLOC paths, i.e., a pair source RLOC and destination RLOC choices. Taking into account both inbound and outbound routing preferences and also the performance associated with an RLOC-to-RLOC path choice, the routing game $G$ between the two LISP networks takes the form of $G = G_{cost} + G_{perf}$ similar to the form already discussed for PEMP routing. In the LISP context, $G_{cost}$ denotes the cost game built upon the routing preferences for sending traffic over source RLOC as well as receiving traffic over destination RLOC. For the purpose of improving resiliency between LISP networks by promoting the use of path with high level of diversity, the performance of a RLOC-to-RLOC path is modeled as the number of path connecting the source and destination RLOCs; the higher the number of path the lower the performance cost, and the performance game $G_{perf}$ is built upon these values.

It is worth noting that in the LISP routing game, a different way of threshold computation is proposed than the one proposed for PEMP routing. Relying on the fact that the maximum as well as the minimum potential values of the routing game changes with the game configuration, the authors proposed to compute the threshold based on arbitrary statistical choices (third quartile of the potential distribution).

## 2.5 MPTCP strategic load balancing

The idea of equilibrium multipath routing is potentially extensible to any routing or load-balancing framework where agents can express preferences on both egress and ingress transmission links, associated or not with additional performance metrics. Besides the already presented peering carrier networks or the remote edge network communications at the transit and edge layers, it can also be extended to support strategic interactions at the application transport layer if multiple interfaces are made available to the application, such as with MPTCP. This is what is explored in [3], where a multipath equilibrium routing framework was proposed to enhance the load balancing decision between interacting multihomed end devices (terminals and servers).

More precisely, the interactions between the source and destination endpoints of a MPTCP connection when deciding the percentage of traffic sent on each of their subflows could be modeled as a non-cooperative routing game. In that game, each endpoint has as strategies the subflows for routing traffic to the other end. For such an endpoint, a subflow is defined by a pair of its outgoing interface and the incoming interface of other end point. For instance, if the source has two network interfaces connecting it with the destination and the destination maintains three network interfaces for connecting to the source, then each end point has six subflows for sending traffic to the other end. The utility for each end point is defined in [3] as the function of the amount of traffic uploaded on the outgoing interface and the amount of traffic downloaded in the incoming interface. Because the utility of an endpoint not only affects its decision but also the other endpoint decision, but no binding agreements can always be set up, a non-cooperative game modeling is appropriate in this situation as well.

Via a similar routing coordination framework as the PEMP and LISP-TE ones, source and destination endpoints of an MPTCP communication exchange their preferences for sending as well as receiving traffic over a subflow in term of routing costs. Some MPTCP signaling attributes can be made suitable for such usages. The routing game between these endpoints is built upon these cost values with the resulting equilibrium profiles determine the subflow load balancing strategy for each endpoint. Taking into account both interconnection and performance costs for a given strategy, the routing game can be considered as a sum of the cost game $G_{cost}$ and the performance game $G_{perf}$. In [3], a trade-off coefficient between $G_{cost}$ and $G_{perf}$ cost components was also introduced.

While the interconnection cost game $G_{cost}$ is built upon a monetary interconnection cost or any customizable arbitrary preference, the performance game $G_{perf}$ is built upon a metric that directly impacts the performance of MPTCP communication. In fact, the performance cost associated with a subflow in $G_{perf}$ is modeled as the one-way delay for

sending traffic over that subflow.

Similarly to the coordinated routing game solutions with PEMP or LISP-EC based, a potential threshold is exploited to increase the path diversity for the routing solution. However, a different approach for threshold computation is proposed. Arguing that the trade-off coefficient already be used to combine performance and interconnection by weighting the importance of one-way delay in the load balancing decision, the way potential threshold is computed is proposed as a function of the trade-off coefficient.

## 2.6 Summary

We presented in this chapter the three protocols that are covered by this manuscript, BGP, LISP and MPTCP, and we presented as well how to enhance their behavior by means of non-cooperative game decision making. We presented existing works at the state of the art making use of multipath equilibrium routing.

The studied game theory based routing coordination framework were proposed to enrich the routing decisions between peers at different layers. At the transit network layer, PEMP routing enables two peering carriers to strategically route their traffic following equilibrium paths. At the Internet edge network layer, remote networks can enable multipath equilibrium routing, provided egress control is made available. At the application transport layer, a MPTCP communication can strategically load balancing traffic over multiple subflows according to multipath equilibrium solutions.

Therefore, a similar form of coordinated routing game is constructed at different layers. However according to its own context, each layer leads to different definition of cost components. For instance, the approach employed for computing the potential threshold is also not the same for every game. In Table 2.2 we summarize the cost definitions and threshold computation approaches for each of the three application domains described above and we leverage on for the contributions described in the next chapters.

| routing game | routing costs | performance cost | potential threshold |
|---|---|---|---|
| BGP-PEMP routing [1] | IGP ingress and egress routing costs | Peering link congestion cost | IGP path cost error |
| LISP-EC routing [2] | Routing preferences for source RLOC and destination RLOC | Transit path diversity | 3rd third quartile of statistical distribution of potential values |
| MPTCP scheduling [3] | Monetary device interconnection cost | One-way delay | Set linearly with the routing-performance cost trade-off coefficient |

Table 2.2: Summary of discussed multipath equilibrium routing applications

# Chapter 3

# Carrier network equilibrium routing: from theory to practice

Competitive routing across peering links is a notable problem in Internet routing. A few years ago, a proposal to incrementally modify the Border Gateway Protocol (BGP) decision process was done, to improve routing coordination by leveraging on the existing multi-exit discriminator BGP attribute as signaling medium among peering Internet networks. It is the already overviewed Peering Equilibrium Multipath (PEMP) routing: based on a non-cooperative potential game, it can improve routing stability and efficiency while respecting unilateral routing choice, by supporting strategic multipath forwarding decisions. Our contribution in this work is twofold. First, we document an implementation of PEMP routing in the Quagga open source router, better specifying some aspects. Then, we specify how weighted load-balancing should be done in PEMP routing and examine the benefits against even load-balancing. We provide a performance evaluation of the resulting PEMP routing system, showing that the computing overhead is limited.[1]

## 3.1 Introduction

The Internet routing system is today based on the Border Gateway Protocol (BGP) [4], which is a path-vector distributed routing protocol allowing, in the current Internet, dozens of thousands of Internet Autonomous Systems (ASes) to exchange hundreds of thousands of inter-domain paths. In its current version, BGP is such that unilateral preferences of ASes can be expressed by means of policy routing, for both inbound and outbound traffic, at the prefix and neighbor levels. After filtering routes by policy routing rules, when multiple routes are available for a same destination network prefix, the BGP decision

[1]The content of this article was published in [28].

process can avoid an arbitrary path selection either by taking the path allowing to exit your AS network at the least cost (also known as 'hot-potato' routing), or by taking the path that is preferred by the neighbor ('cold-potato' routing) on a per-neighbor basis. While the former is a purely selfish routing rule, the latter (rather altruistic) makes business sense only when the neighbor is a customer AS.

Where there is no business agreement between two interconnected ASes, and an equivalent traffic volume exchange between respective customers over both directions exists, the ASes interconnect under a so-called 'peering agreement'. In such cases, hot-potato routing can lead to quite inefficient bilateral routing solution because of the possible double application of selfish routing [29]. A few attempts in the literature try to overcome these limitations by forms of multipath routing, for example by explicit route negotiation as in [19, 20, 21], or implicit equilibrium routing as in [22, 1]. The common idea behind these works is to enlarge the set of announced BGP paths to allow improving the bilateral routing, namely in terms of routing stability.

The contribution of this work is twofold. First, we enhance the PEMP routing framework, addressing its load-balancing algorithm. Then, we document and evaluate its real implementation in a widely-used open source BGP router, the Quagga routing suite [6], publishing the code as open source [30]; we followed the specifications in [1], rectifying some aspects. We show that the computing overhead is indeed limited.

## 3.2 BGP-based routing coordination protocol requirements

PEMP is an extension of the standard BGP mode that can be incrementally deployed in the current Internet. A pair of ASes willing to deploy PEMP need to just update the BGP border routers that collect the traffic from the target BGP destination cone, i.e., the set of IP prefixes to which apply equilibrium routing (e.g. marked by a BGP community). The other core BGP routers, as well as the BGP border routers at the frontier with the peering AS, do not need to be aware of PEMP routing: they just need to let MED signaling pass transparently through their filtering rules. The functional blocks to be implemented by a PEMP-enabled BGP router can be briefly summarized as follows:

- Computing directional routing cost between itself and each egress router for a given set of prefixes.
- Encoding the computed routing costs into the MED attribute of corresponding route advertisements.
- Upon advertisement reception, decoding the MED and updating the routing game by considering all the possible combinations of path selections from both domains.

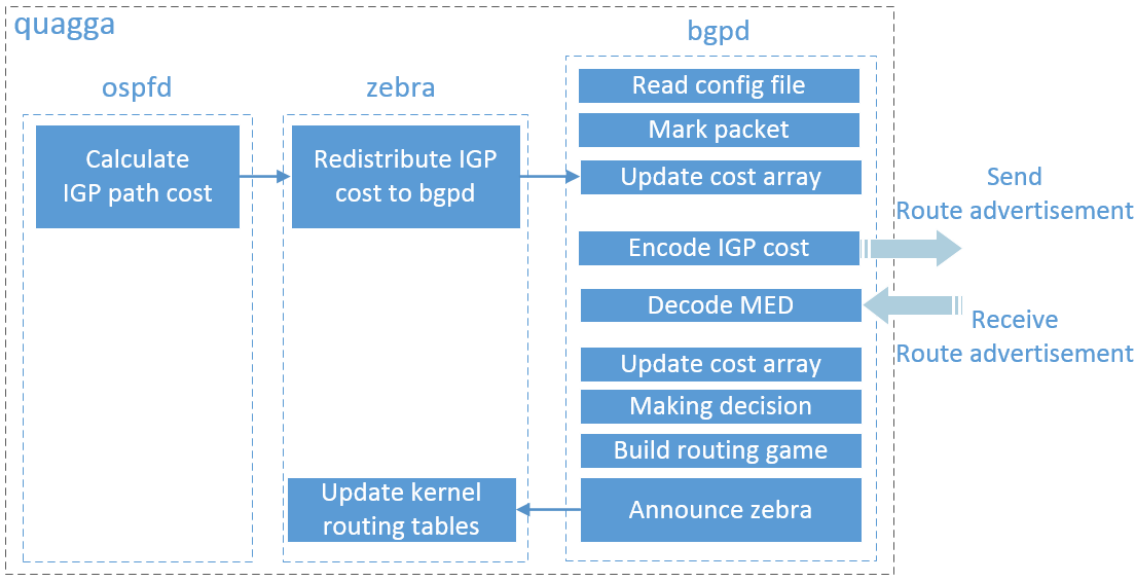



Figure 3.1: System architecture of PEMP-enable Quagga router

- Upon each setting update, determining the equilibrium routes based on the weighted load-balancing logic.
- Classifying and forwarding packets based on source and destination addresses.
- Processing inter domain routing decision and distributing load efficiently among selected paths.

## 3.3 System architecture

We enhance Quagga [6], a well-known open source routing software, more precisely its v. 0.99.23, a stable release that supports weighted multipath routing. We choose Quagga also because differently from other common routing software like BIRD [31], it has a modular design in which each routing protocol works separately and operates as an independent process. For exchanging routing information, these processes interact and communicate with each other via a core process (ZEBRA) that plays the central role in the whole working model of the router: it summarizes routing information learned from different active protocols and frequently updates the kernel's forwarding table with new paths. The game-theoretic logic about equilibrium and load-balancing computation is externalized to an external 'routing game library (RGL)'. Our code is distributed under a GNU General public license [30].

In Fig. 3.1 we present the PEMP Quagga system architecture meeting the expressed requirements. To highlight the changes, we map all the new supporting functions into the original modular design of Quagga and hide the unaltered processes. We limit the

IGP support to OSPF. Therefore the implementation of IGP path cost calculation only involves changes in the OSPFD module; it has been restructured to include the computation of ingress path cost i.e. the routing cost from each border router to the PEMP router. The other two key daemons involved are ZEBRA and BGPD. To update ZEBRA with directional path costs, we attach in the ROUTE_ADD message sent from OSPFD, the ingress cost value as well as the identification of corresponding border router. Hence we modified ZEBRA to correctly parse the new form of ROUTE_ADD message. With such modifications to ZEBRA and OSPFD, we meet the initial requirement for a PEMP router. Involved functions: zread_ipv4_add(), zsend_route_multipath(). In the following, we detail the major changes applied to the BGPD module to support PEMP routing.

- The routing decision is made on a per-flow basis, where a PEMP flow is defined by a pair of BGP communities: the local community of the upstream source networks, and the peer community of the downstream destination networks.

  The router is made able to differentiate PEMP flow traffic from normal traffic using packet marking: the classification rule is derived from a configuration file that states how to mark an incoming packet belonging to a predefined flow (to be executed by the firewall, these marking follow the FWMARK rule format). A flow-based forwarding mechanism is then needed to fulfill the requirement. Involved function: *bgp_route()*.

- Both ingress and egress cost of a routing strategy are embedded in the ROUTE_ADD message sent from ZEBRA to BGPD: the egress filtering function that automatically checks route attributes has to be customized to let the related BGP advertisement being eventually sent. The MED coding is implemented over the 32-bit value.

  Involved function: *bgp_redistribute_add_pemp()*.

- PEMP decoding is implemented to let the routing game data structure be built.

  The game structure is called every time an advertisement for a PEMP flow is detected, and is processed using the RGL methods. Involved functions: *bgp_med_decode()*, *bgp_pemp_game_build()*.

The above ones are control-plane enhancements. Additional forwarding plane changes are described in the following.

- In BGPD, routes determined by both the standard BGP and the PEMP decision processes are added to the same multipath route structure, where they are distinguished by the community ID attribute. BGPD then announces the multipath route

to ZEBRA by a ROUTE_ADD message customized to allow attaching at each update the load-balancing weight and the community ID information. Involved functions: *bgp_best_selection(), bgp_pemp_game_build(), bgp_zebra_announce().*

- Eventually, ZEBRA needs to update the kernel's forwarding table with routes learned from the BGP/PEMP decision process. Adaptations were needed to process the new ROUTE_ADD message format, which can include different next hops for a same destination. A separate routing table than the default table is needed as PEMP routing is source-destination based and not simply destination-based as in standard BGP. Hence we extended ZEBRA to allow to update both types of tables, the default one and the PEMP one reserved for local community specific traffic. The target table is so identified thanks to the community ID information set as above specified. Involved functions: *zread_ipv4_add(), net_link_route_multipath().*

One significant merit of PEMP comes from its design, rather than looking for a separated routing coordination protocol, PEMP marginally enhances the current BGP protocol by adding the necessary extensions to the signaling and decision process to allow for equilibrium routing solutions.

Interoperability with legacy routers is considered as one of the crucial requirements we took into consideration when designing how to classify incoming packets, to do selective encoding IGP path cost, and to construct multiple routing tables. As we show hereafter, a PEMP-enabled router is able to work as smoothly as a legacy BGP router while performing effectively equilibrium routing for configured peering domains.

Overall, the added-in capabilities increase the total number of lines of code in Quagga by only 8%, 5% of which due to the BGPD process, the modifications in both ZEBRA and OSPFD processes being accountable for the remaining 3%). The complexity of implementing a new capability is quite interesting for developers, however it is not the right indicator for network operators that are more interested to the impact of router's performance instead.

## 3.4 System level performance evaluation

We emulate a realistic peering scenario by deploying two ASes interconnected via three peering links, using a partial mesh topology and OSPF as IGP. Each AS domain is constructed with 10 Quagga routers, among which one is configured as PEMP router and three others are selected as border routers with the neighbor AS.

We report in the following stress-test results on the PEMP routing system. We measured the performance of a router in term of processing time, i.e., the total amount of

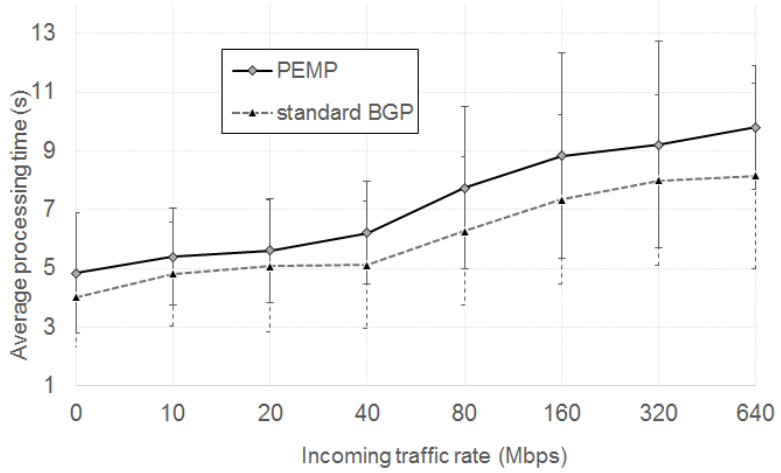



Figure 3.2: Average processing time upon IGP path cost change.

time required for processing PEMP network/link state updates and for installing new routing decision, for an increasing data-plane traffic load. The experimented routers are built in Ubuntu virtual machines with two 2.397GHz CPUs and 8GB of live memory. Two experiments are conducted to study the overheads of PEMP solution in different scenarios.

In the first experiment, we measure the processing time of router in case of OSPF path cost changes. This time typically is due to the time to recompute the IGP shortest paths and costs, to update the BGP states and to issue (possibly new) BGP routing decisions depending on the IGP costs. With PEMP, extra marginal delays are introduced for ingress cost calculation, local IGP path cost update, and game building processes. We aim to have an experimental evaluation of the total PEMP execution time overhead. It is worth noting that the current BGP implementation in Quagga waits for a periodic update process that runs every 60s to capture IGP path cost variations, we subtracted this constant time to focus on the marginal time increase. As depicted in Fig. 3.2, the average processing time of both PEMP and BGP are rising gradually as the data-plane traffic increases. Unsurprisingly, the standard BGP router always shows a better performance than its extension. The processing time gap is, however, quite limited, about 15%, and regardless of incoming bitrate. As observed from the experiments, the IGP path cost update phase was the most time consuming task. With PEMP, the delay for path cost calculation is higher than with standard BGP because it needs to calculate the ingress path cost to each egress point. We believe this phase could be improved by code optimization to make this step faster.

In the second experiment, we measure the BGP router processing time in case of MED-icated route updates. Differently from the previous experiment, changing MED signaling is handled right upon reception. By default, once a MED value is received by PEMP

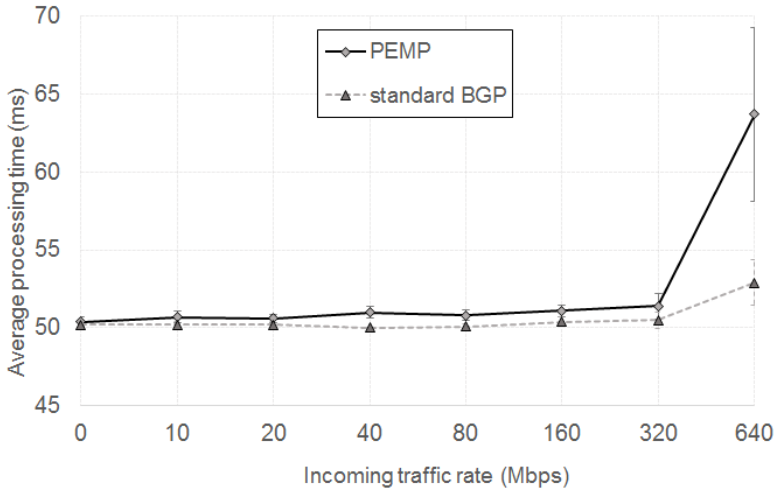



Figure 3.3: Average processing time upon MED attribute change.

enable router, the corresponding routing game is rebuilt and the routing decision is made in response to the game equilibrium routing.

To simulate a real operational router and evaluate its processing time under different traffic load scenarios, we increase the incoming data-plane traffic rate. The stress-test result is presented in Fig. 3.3, again in terms of average processing time. For this experiment, the processing time is at a much smaller scale than for IGP link state changes ($ms$ instead of $s$). The difference between standard BGP and PEMP is this time much smaller (lower than $2ms$), and almost negligible for low and medium loads. However, for high loads the processing time gap with PEMP increases to roughly 20%, which is not enormous, also considering that for very high bit rate the usage of open source routers is a seldom choice. The marginal gap in high-end multi-core routers is expected to be much lower.

## 3.5 Enhanced load balancing

Leveraging on the potential sensibility and the potential threshold to fine-select routing equilibria, PEMP can alleviate the routing instability caused by hot-potato routing by preventing single equilibrium solution. When multiple equilibria exist, it is needed to develop an efficient load distribution strategy. In [1] it is proposed to perform an even load-balancing over the links corresponding to the routing equilbria. In this section, we present how to go beyond this basic rule.

For the previous example in Fig. 3, let us assume that the computed threshold value is $\tau = 4$; this implies that the profile $(l_1, l_2)$ is also selected in the set of equilibrium solutions, hence the related routing solution indicates load-balancing over the three peering

links from AS I to AS II and single-path routing over $l_2$ for traffic from AS II to AS I. Performing an even load-balancing as suggested in [1], e.g., 33% on $l_1$, 33% on $l_2$, and 33% on $l_3$ for traffic flows from AS I to AS II, may appear in this context a rude decision as those profiles with lower potential value should attract more traffic as they are strategically more stable.

It is worth recalling that a profile $(x, y)$ is selected in the routing solution if and only if $P(x, y) \leq P_{min}+\tau$. With the purpose of minimizing the change in equilibria set before and after intra-domain failures, the value of threshold $\tau$ is computed relying on the variation of IGP path cost upon possible failures. In this way, the threshold enables to select in the routing solution the profiles that have good chances to become a pure-strategy equilibrium, i.e., which have a potential value equal or near to the minimum potential. In other words, the lower the potential value of a routing profile is, the higher the routing stability is. Distributing traffic over selected profiles equally (i.e., doing an even load balancing as specified preliminarily in [1]), does not adequately reflect this concern.

Therefore, we propose to implement an explicit PEMP load-balancing weighted as a function of the distance from the potential minimum. Let $S \in X \times Y$ be the set of selected profiles, profiles with a potential value below a threshold $\tau$. $X$ and $Y$ are the set of all routing strategies available at local and peering AS respectively. The load balancing ratio for a link strategy $x$ in $X$ is $b_x$ computed as (dually for $b_y$):

$$b_{x'} = \frac{\sum_{(x,y)\in S}^{x=x'}[1 + \tau - P(x, y)]}{\sum_{(x,y)\in S}[1 + \tau - P(x, y)]} \quad \forall x' \in X \tag{3.1}$$

The approach for determining the threshold initially proposed in [1] consisted in exchanging via the MED also a global directional path cost error computed as a function of link failures that could manifest at each side, taking the maximum among the minimum best path cost variations. In practice, we realized during implementation that this process would be too complex to implement, because it would add computational overhead and would mind the reliability of PEMP signaling.

We propose, instead, a more light-weight computation of the potential threshold $\tau$ for PEMP weighted load-balancing. It consists in computing a statistically relevant differential potential value corresponding to the occurrence of link failures based on known experimental failure distributions at each side. Let $\Delta P$ denote the potential difference of a strategy profile before and after an intra-domain failure. By monitoring the variation of $\Delta P$ over a number of individual link impairment scenario, a distribution of $\Delta P$ can be computed.

As an example, we apply the experimental individual link failure distribution made

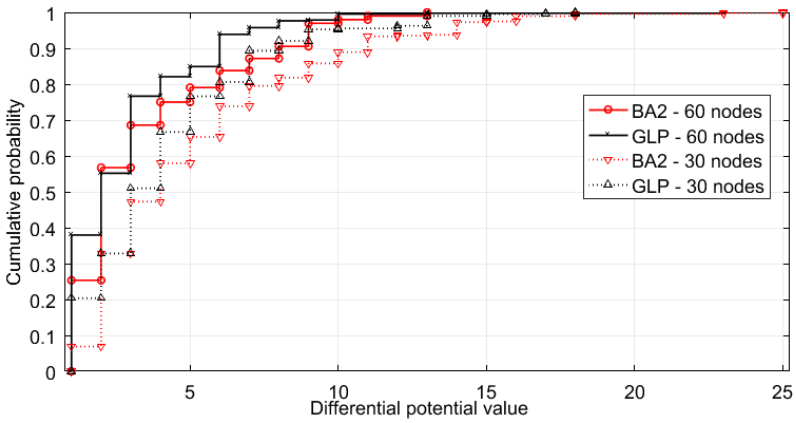



Figure 3.4: CDF of $\Delta P$ in 30 and 60 nodes topologies.

available in [32], which is a power-law for core links (high failure link) $n(l) \propto r(l)^{-0.73}$, in which $n(l)$ denotes the number of failures on a link $l(l = 1, ..., L)$ and $r(l)$ returns the ranking of link $l$ with respect to its connection degree. We employ the BRITE topology generator [33] for topologies of 30 and 60 nodes, using the Barabasi and Albert BA2 model [34] and the Generalised Linear Preference (GLP) model [35]. In the generated network topologies, links are configured with a $[1, 20]$ weight range. For every case, we repeat the failure simulation 50 times, each time with a different topology and IGP weight configuration. Figure 3.4 reports the Cumulative Distribution Function (CDF) of $\Delta P$. It can be seen that with large topologies, the 95% $\Delta P$ is lower than 10, and for small topologies it is lower than 15, as small topologies are more subject to route instability than large topologies, as the chance that shortest path goes along a failed link is higher. It is worth noting that there is no need to have the threshold to be set exactly the same at the two borders, despite it could be a desirable routing behavior in some cases.

With the proposed approach, determining a proper threshold is no longer a concern when considering the complexity of the PEMP routing solution for practical implementation. More important, the enhanced load-balancing technique introduced in this work offers a fair distribution over the extended set of equilibria. Forwarding a larger portion of traffic to more stable path, weighted load-balancing strategy helps to reduce the volume of traffic shifted when routing change due to transient failure. The following experimental result justifies the effectiveness of proposed solution.

As already mentioned, in PEMP the choice on the potential threshold determines the stability of routing decision, whereas the load-balancing scheme decides on the amount of traffic sent on each route. When there is a transient failure in the network, routing decision may be varied and consequently traffic load is shifted from one path to another path according to the new load balancing decision which decided by the load-balancing scheme. Therefore, to evaluate the efficiency of one load-balancing scheme over the other,

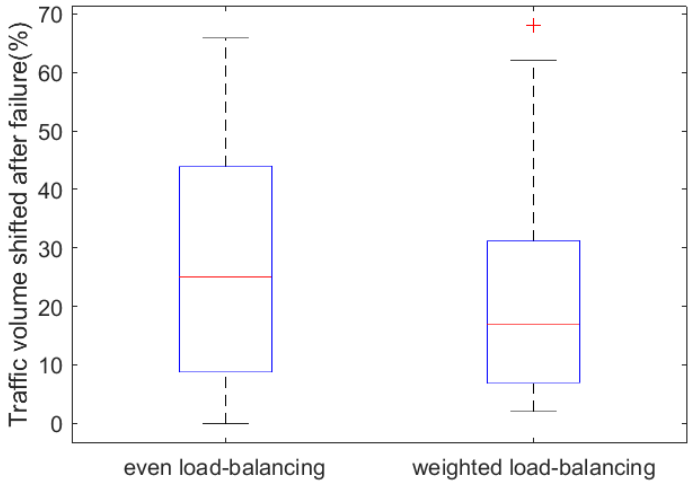



Figure 3.5: Volume of traffic shifted after failure

we examine the difference in the amount of traffic shifted during a network impairment. In this experiment, we closely monitor the change of traffic distributed at each selected path before and after a simulated failure. This measurement is applied for both weighted and even load-balancing schemes under identical conditions (same potential threshold, network topology and link failure). The failure generation follows the power-law distribution described in [32]. Network topologies are created from BRITE [33] with BA2 [34] as the modeling approach; the experiment is performed over 20 different such random topologies. At least five individual failures are generated in each topology.

In Fig. 3.5 we report the experiment results. Weighted load-balancing shows a better performance than even load-balancing: it has a median of 17% shifted traffic, against 26% with even load-balancing. Furthermore, its upper quartile is more than 10% smaller.

Employing the proposed algorithm, the load distribution ratio is derived directly from the potential value, therefore it takes into account also small variations. It is worth mentioning that weighted load-balancing shows a higher sensitivity to small variations; this is the reason why the minimum with even load-balancing is slightly lower.

## 3.6 Conclusions

In this chapter we presented how Peering Equilibrium MultiPath (PEMP) routing can be implemented in real routers. PEMP routing was proposed for making inter-domain routing more stable, in particular across peering settlements among Internet Autonomous Systems.

Its implementation allowed us to validate most of the modeling choices, as well as to revisit some of the design choices at the light of implementation-specific constraints. More precisely, we specified how weighted load-balancing should be performed over PEMP routers, and how equivalent paths can be identified. We also specified how the forwarding logic should operate a dual logic for both standard traffic and PEMP traffic.

By means of extensive tests on realistic emulated network interconnections, we showed that PEMP can be integrated at low computation overhead. We released the PEMP-capable open-source Quagga-based router code [30].

# Chapter 4

# Edge network routing coordination and egress control

The Locator/Identifier Separation Protocol (LISP) was specified a few years ago by the Internet Engineering Task Force (IETF) to enhance the Internet architecture with novel inbound control capabilities. Such capabilities are particularly needed for multihomed networks that dispose of multiple public IP routing locators for their IP networks, and that are willing to exploit them in a better way than what possible with the legacy Border Gateway Protocol (BGP). In this chapter, we specify how to enhance the LISP routing system to perform egress control too. Our goal is to give the highest possible routing optimization degree to LISP networks, so that ingress and egress traffic engineering strategies can be jointly performed, without requiring coordination between LISP and BGP. We design the enhancement to the LISP router system, specify the required protocol extensions, open sourcing the code and proving the overhead and the achievable gains by experimentation.[1]

## 4.1 Introduction

In this work, we focus on giving a more efficient inter-domain traffic engineering scope of operation to multihomed stub Internet networks, which account for the majority of Internet Autonomous Systems (ASes). In fact, roughly 84% of the them are stub ASes, and most of them are multihomed [2].

The growing number of multihomed networks challenges the scalability of the whole Internet routing system. When a multihomed AS announces its network prefixes to several providers, a common practice consists in de-aggregating the parent prefixes to perform fine-grained inbound traffic engineering, so that, with $n$ transit providers, a multihomed AS

[1] The content of this article was published in [36].

typically announces about $n$ different sub-prefixes for each single prefix, hence contributing to Internet routing table bloat [37]. As shown in [38], over a period of 4 years multihomed ASes created approximately 20-30% more prefixes to the BGP routing table than single-homed ones. In order to preserve Internet scalability, a wide range of alternative solutions have been proposed, most of them relying on IPv6 addressing and/or following the concept of separating the locator and the identifier roles of an IP address.

Besides the primary goal of a highly available Internet interconnection, multihomed ASes also target to improve their network performance by employing intelligent route control. With BGP, egress traffic engineering (e.g., to which transit provider to send which traffic) can be performed by means of local preferences in the routing decision process [38], while ingress traffic engineering (e.g., through which transit provider which incoming traffic comes from) is strongly limited by the absence of adequate control-plane functions: despite some tricks are possible, there is not direct control on incoming traffic routing. Also to enhance this aspect of Internet routing, the Locator/Identifier Separation Protocol (LISP) [5] was proposed; indeed, LISP allows to associate to each prefix (announced via BGP) a preferred routing locator, among many possible ones, by means of a mapping system (independent of BGP). In this way, a multihomed network can perform egress traffic engineering using BGP, while using LISP for ingress traffic engineering purpose. However, to dispose of both ingress and egress traffic engineering for a given multihomed network, the two protocols are supposed to inter-work, which is not specified in the standard. More precisely, it is not explicitly specified how LISP and BGP should run in a same node, or how physically separated LISP and BGP routers should be interconnected, etc. This is also complicated by the fact that a multihomed LISP network may not be running BGP, as it happens with edge networks with provider dependent addressing, or targeting a specific LISP deployment use-case that poses no strict external addressing requirements as BGP deployment does. Indeed, as it is evidenced in the new LISP Working Group charter, LISP has many applications (e.g., data-center networking, mobile user mobility management) that do not encompass BGP routing, hence having egress control in LISP without requiring integration with BGP is appealing.

In order to cope with these operational limitations, we specify in this chapter how the LISP routing system can be enhanced in order to integrate egress control functions, besides the standard ingress control ones, in a way that does not impact the LISP architecture, nor any other protocols, and that can stay purely local to a LISP site.

## 4.2 LISP Egress Control

To cope with the lack of egress control in LISP, we propose LISP Egress Control (LISP-EC), an enhancement of the LISP routing behavior that gives an xTR the control on source RLOC selection, independently of the underlying IP routing decision or static configuration.

LISP-EC is based on an alternative EID-RLOC mapping structure that allows associating destination RLOCs with multiple source RLOCs. The novel design permits xTRs to determine both head and end points of a tunnel without consulting the underlying routing protocol. In Fig. 4.1 we depict the proposed LISP-EC mapping entry structure. Fundamentally, it is an extension of the legacy mapping structure, in which each destination RLOC is associated with an extra list of source RLOCs. These attached RLOCs maintain the same properties as the destination RLOC, but hold a different meaning. While the destination RLOC is the routing locator for the remote EID-prefix(es), the source RLOC is the routing locator for the prefixes originated from local network. Within a LISP site, these local or source locators can be seen as the gateways for end hosts. Thanks to the LISP-EC mapping design for remote EIDs, an xTR is now capable of relating source RLOC choice with the selection of destination RLOC, and vice-versa.

It is worth mentioning that LISP-EC mapping design is not a traffic engineering mechanism per se, it is rather an extended behavior of LISP routers giving them the novel traffic engineering capability to distribute traffic among the gateways, which could be used by an external control-plane. LISP-EC mapping design introduces a new dimension for jointly controlling inbound and outbound traffic.

### 4.2.1 From RLOC selection to LISP-EC traffic engineering

Traffic engineering in standard LISP is limited to the capability, for the destination network, to announce its preferences over its RLOCs through the LISP mapping system; the source network is supposed to follow the destination network preferences. However, there may be a strategic clash in case the destination network preferences are for some reasons opposed to source network preferences. In such a case, the source network can bypass destination network preferences, knowing that if it sends traffic to an RLOC that is currently not the preferred one by the destination, such traffic will not be dropped (this is the case of all the public LISP implementations as of today).

With LISP-EC, we allow the source network taking into consideration its upstream preferences in a way that (i) it still permits to take into account destination network preferences, and (ii) increases the path diversity available between two edge networks. Indeed, while with BGP the number of available paths is equal to the number of external

BGP peers, and with standard LISP it is equal to the number of destination network RLOCs, with LISP-EC it is equal to the product between the number of source RLOCs (possibly equal to the number of external BGP peers) and the number of destination RLOCs.

The processes configuring the egress priorities and weights at the source LISP network and the ingress priorities and weights at the destination LISP network can be two independent processes – as considered in [2], supposing the two edge networks are independent autonomously managed networks – or can be the result of a bilateral routing decision of Internet routing optimizers (commercial solutions exist, e.g.[39]) – which makes sense when the border routers of the two edge networks are operated by a same administrative entity.

Therefore, with LISP-EC, a new dimension of outbound traffic engineering mechanism is defined: it is no longer restricted to the determination of gateway or destination locator solely, it is now the control of load distribution over all possible RLOC-to-RLOC paths. By evaluating all combinations of gateway and destination locator, the best RLOC-to-RLOC path can be decided. Thanks to the LISP-EC extended mapping design, such a decision can be expressed and operated by means of RLOC priorities and weights.

Different traffic engineering policies can emerge in a LISP-EC communication context. In the following we list some we could identify - from one requiring no coordination whatsoever between LISP sites, to one requiring full TE control of both sites, passing through light coordination ones.

- *best source locator*: this policy consists in determining the best source RLOC based on local policies, whereas selecting the destination RLOC preferred by the destination. The decision on the best source RLOC can be taken following local egress TE preferences, for instance based on interconnection costs or performance (e.g., delay).

- *best forward path*: this policy consists in selecting the best RLOC-to-RLOC forward path, among all paths from the source xTR to the destination network, based on local policies, hence overriding the standard LISP behavior for which destination RLOC is chosen following destination preferences (RLOC priorities and weights) distributed by the mapping system. The decision on the best forward path is therefore entirely based on local policies, with a local preference on the destination RLOC that can be opposed to the inbound traffic engineering preference of the destination, because of forward path performance (e.g., delay, reliability) or whatever policy reasons.

- *equilibrium path*: under the hypothesis that two LISP networks communicate with equivalent traffic volumes over the two directions, this policy consists in selecting the

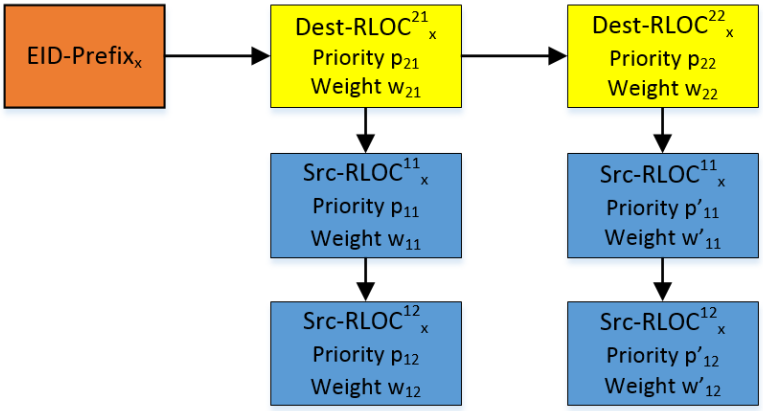



Figure 4.1: Extended mapping entry structure

RLOC-to-RLOC path corresponding to a routing equilibrium solution that strategically takes into consideration the preferences of both parties on both inbound and outbound routes. As a possible approach to compute the egress control metrics we refer to the routing interaction between LISP networks that was modeled previously in [2] as a non-cooperative game; a polynomial-complexity equilibrium computation framework was proposed and evaluated by a simulator assuming LISP egress control capabilities were available at xTRs.

- *global optimum path*: this policy considers, as the previous one, that two LISP networks exchange traffic with each other, but it differs from the equilibrium one in that the source RLOC and the destination RLOC are chosen accordingly to the global optimum path (i.e., what in the non-cooperative game modeling would correspond to the social welfare profile), which could differ from the equilibrium one, and which could override the unilateral preferences of both networks.

Besides the selection of one or multiple destination RLOC(s), the outcome of a LISP-EC TE policy is the configuration of source RLOC priority and weight in a novel LISP mapping entry processing system as proposed hereafter.

### 4.2.2 Implementation Requirements

We address the LISP-EC implementation requirements based on the LIP6-LISP OpenLISP node system architecture [7]. Such a system has four components: the mapping database, the control-plane, the data-plane and the mapping socket. The control-plane runs in the user space and holds the responsibility for constructing and distributing mapping entries. Packet encapsulation as well as decapsulation runs in the kernel space relying on the mapping data. The mapping socket handles all the communications between user

and kernel spaces, and it helps to populate the mapping databases. LISP-EC requires extending all four components.

For the sake of incremental deployability, LISP-EC should inherit the mapping structure from the legacy implementation, relying on the same mapping server and resolver interfaces, and should have no binding impact on control-plane messages.

LISP-EC deployment should require upgrades at the xTRs only. The additional xTR operations needed are:

- to encapsulate outgoing packets with source address set to the source RLOC address determined by a local traffic engineering policy.
- to maintain a novel mapping structure that allows coupling the selection of destination and source locators.
- to manage the independent setting of priority and weight for both source and destination RLOCs associated with a given EID in mapping entries.
- to differentiate traffic control policies for different outbound flows.

### 4.2.3 System architecture

For the xTR system to integrate LISP-EC features, we design the mapping structure in Fig. 4.1; it requires modifications to user and kernel spaces. Accordingly, the mapping socket that handles the interactions between control and data plane also needs to be updated. In Fig. 4.2, we draw the system architecture of LISP-EC capable xTR in which new and modified processes (e.g., egress control, mapping socket, packet encapsulation) are denoted with a different color (green).

With the purpose of validating and manipulating the mapping entries received from destination networks before adding them into the mapping cache, the egress control module is developed by updating the MAP-REPLY processing logic. More precisely, the *read_rec*() function defined in *plugin_openlisp.c* is extended: once received a MAP-REPLY, *read_rec*() populates the mapping entry for the announced EID-prefix with destination RLOCs and associated attributes parsed from the message. Instead of employing the attached priority and weight, the extended logic allows to use an alternative set of RLOC metrics. More importantly, it is possible to associate a destination RLOC with one or several source locators. After parsing one destination RLOC, a list of *source_locator* is constructed by querying the local mapping database. The corresponding priority and weight for each source locator in the list can be statically configured or dynamically computed regarding to the employed traffic engineering policies. It is worth noting that, in order to keep

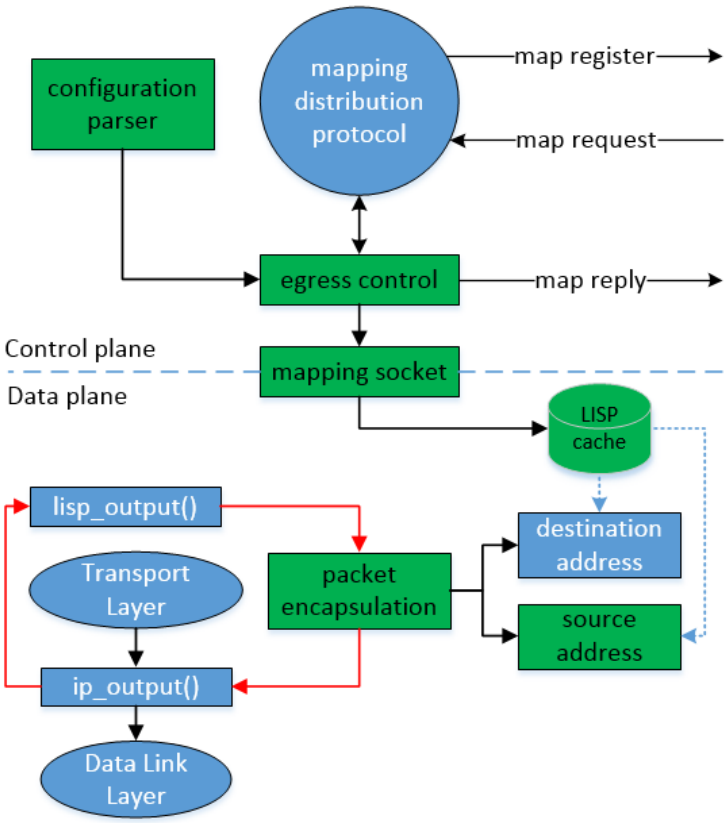



Figure 4.2: LISP-EC system architecture

track of all the received mappings in the mapping cache (own by the control-plane before transferring to the kernel mapping system), we extend the EID-RLOC mapping structure at the user space as well.

The EC traffic engineering logic, i.e., how to couple source and destination locators as well as how to combine priority and weight for source and destination locators for path selection, has to be integrated in the egress control module.

Extending from the standard procedure, the LISP-EC configuration parser allows peering relations to be established between distant edge networks, more precisely between EID-prefixes. Besides specifying its RLOCs, each EID-prefix can now be associated to a peer remote prefix from another LISP site. Once receiving a MAP-REPLY, the xTR checks for a flow control agreement between the local EID-prefixes and the announced prefixes. If local prefix $x$ peers with remote prefix $y$, depending on agreed TE policy the EC module associates a subset or all source locators of $x$ queried from the mapping database with each RLOC of $y$. Thanks to such a 'virtual peering' agreement between LISP sites, different control policies can be applied for the same pair of LISP sites depending on the source and destination prefixes.

Besides expanding the control-plane processing module, we also upgrade the kernel

space with an extended version of the mapping socket and a novel source address selection procedure. The major modifications on those two processes could be summarized as follows.

- *Mapping socket*: to adapt with the extended mapping sent by control plane, it is needed to define an alternative message structure. The legacy mapping message format consists of a message header, followed by an EID socket address and then a list of the routing locators. The total number of locators in the list is specified in the header. Each locator is represented by its socket address, followed by a *rlocs_mtx* structure in which RLOC attributes are included. Our design consists in inserting a list of source RLOCs after each destination RLOC. Both source and destination RLOCs share the same format. In order to differentiate them, a new locator control flag is introduced, and for each destination RLOC, the number of associated source RLOCs is also included as a new attribute in *rlocs_mtx*. Besides that, a different logic for message building and handling is developed at *opl_add_rloc*() in *plugin_openlisp.c* and *map_insertrloc_withsrc*() in *maptables.c* respectively.
- *Packet encapsulation*: the modifications made in packet encapsulation module could be reflected via the changes in its source locator selection process. For packets sending to peering EID-prefixes, instead of looking up the routing table, the extended *map_select_srcrloc*() function queries the mapping cache to find the corresponding local gateway for selected destination locator. To enable load balancing among selected gateways, we employ a technique similar to the one implemented in OpenLISP for destination RLOC handling.

## 4.3 Performance evaluation

Extending the standard mapping structure, LISP-EC offers higher control over the inter-domain routing paths, and consequently opens opportunities for improving network performance. However it also introduces some extra operational costs at the system level. In the following, we present different experiments showing the trade-off between performance and execution time overhead introduced by LISP-EC.

### 4.3.1 Edge to edge delay

We simulate the edge-to-edge interconnection of two arbitrary ASes, each one has a random number of upstream providers between 2 and 6. At each AS, there is one RLOC per upstream provider. The RLOC-to-RLOC tunnels are simulated to have a random one-way delay between 20 and 250 ms. Inbound RLOC priorities are generated randomly

for each simulation instance. In the simulations, we run 500 random network instances and we show the results by boxplots (showing the maximum, third quartile, median, first quartile, minimum and outliers).

We capture the RLOC-to-RLOC path choices at the two LISP sites under different TE policies. The LISP-EC TE policies previously presented are employed with the delay between source and destination RLOCs as unique performance metric.

Besides LISP-EC TE policies, we also include the 'legacy LISP' behavior (i.e., no source RLOC selection and the destination RLOC is chosen as the one with the highest destination-set priority), and a LISP-based TE approach (indicated 'Legacy LISP with TE') that overrides the destination RLOC preferences and selects the source-view best destination RLOC based on the RLOC-to-RLOC delay. For instance, let $D_1$ and $D_2$ be the two destination RLOCs, and let $S_1$ and $S_2$ be the best source locators toward $D_1$ and $D_2$, respectively, from the source viewpoint. If the delay on the $S_1$-to-$D_1$ path is less than the one on the $S_2$-to-$D_2$ path, then the $D_1$ RLOC priority is locally overridden by the source xTR, updating it with the smallest priority value in its mapping entry.

Delay performance simulation results are shown in Fig. 4.3, which report the forward delay as seen by one of the two LISP sites. As one could expect, the legacy LISP routing decision not being based on source-to-destination forward path performance criteria, it always experiences a sensibly higher delay than the TE policies. When outbound TE policies are applied, the forward delay performance is instead under control. Applying various LISP-EC TE policies, described in Section 4.2.1, the highest gain with respect to 'legacy LISP with TE' can be observed when the best forwarding path is selected ('LISP-EC best fw path'). Controlling source RLOC selection only ('LISP-EC best src-RLOC') yields a performance gain comparable to when controlling destination RLOC selection only ('legacy LISP with TE'). Combining both source and destination RLOC selection capabilities leads to a significant improvement, as we can see in the LISP-EC best fw path case. The median edge-to-edge delay is significantly decreased: compared with legacy LISP, it offers a reduction of roughly 77%. LISP-EC policies adopting forms of collaborative TE between source and destination LISP sites, either by two-side minimization of the delay sum ('LISP-EC global optimum') or by selecting the routing equilibrium ('LISP-EC equilibrium'), are obviously a bit lower in performance with respect to the best forwarding path case (with the equilibrium policy statistically slightly outperforming the global optimum case due to the fact that this plot shows the delay performance as seen by only one of the two networks, and not their sum).

Overall, we show that the statistical gain one could get in terms of performance by applying LISP-EC TE can range from roughly 55% to 77% with respect to legacy LISP,

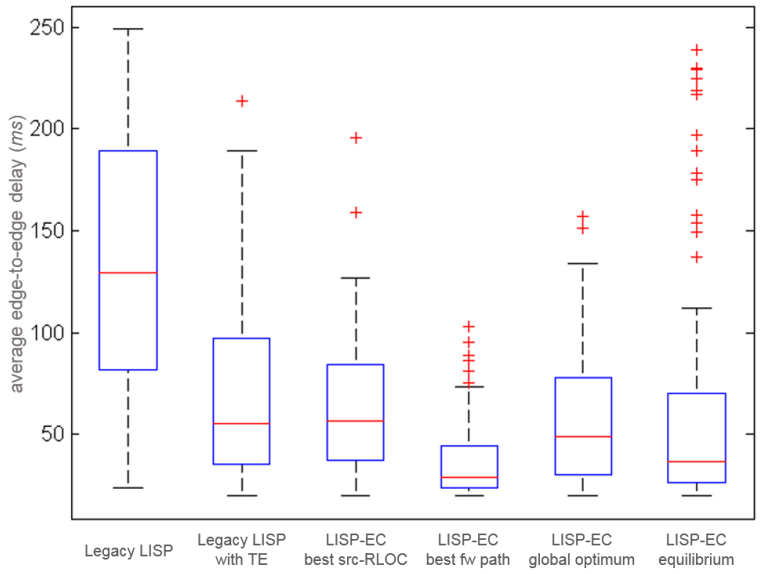



Figure 4.3: Boxplot statistic of edge-to-edge delay

and from 12% to 47% with respect to a performing TE optimization in a legacy LISP setting (i.e., without egress control).

### 4.3.2 System level performance

The benefits from enabling egress control in LISP come at a price, as it obviously introduces extra packet forwarding and control-plane delays. From a practical deployment perspective, we need a better understanding of the extended mapping structure impact on the LISP routing system. In the following, we report the system level performance of LISP-EC router in two different scenarios: (i) when adding a new mapping entry and (ii) when retrieving data from the mapping cache. In both experiments, the performance is measured in term of processing time. The experimented routers are built in FreeBSD virtual machines with one 2.397GHz CPU and 2GB of live memory. We implemented LISP-EC in the LIP6-LISP OpenLISP node, open sourcing the code [30].

In the first experiment, we measure the average delay when a new mapping entry is added into the mapping cache. It takes into account the total amount of time for parsing the MAP-REPLY, executing traffic engineering policies (associating source to destination locator, retrieving priority and weight for each RLOCs), constructing and finally adding the new mapping to the kernel space. In Fig. 4.4 we report the average processing time with legacy LISP and LISP-EC as a function of an increasing number of routing locators. For LISP-EC we include both the case when the egress RLOC metrics are preset, and the case when the egress metrics are computed on the fly. We refer for the latter case to the equilibrium routing computation, which has a linear time complexity [2].

Fig. 4.4 shows that the performance gap increases linearly with the number of locators between two LISP sites - more precisely, the number of RLOC-to-RLOC paths. We can observe that, with 4 paths, LISP-EC leads to a processing time 3 times higher than legacy LISP. Then, the router performance is strongly influenced by the number of additional locator fields appended in the mapping message sent from control-plane to data-plane spaces in the xTR. LISP-EC mapping associates each destination RLOC with a list of source RLOCs, thus multiplying the total number of locator field carried on a message. That explains for the high sensibility of LISP-EC to the number of RLOCs. However the amount of locators is restricted by the number of upstream providers, and for the large majority of edge stub ASes the number of upstream provider is less than 6, and about 2/3 less than 3 [2]. In a quite worst case scenario where each site maintains up to 5 RLOCs, it introduces a difference of 100 ms with respect to the standard LISP. As adding a mapping entry is not a frequent operation in most of LISP use-cases, such a system performance gap could be considered as unimportant.

In the second experiment, our focus moves to the processing time overhead experienced at the kernel space where incoming packets are forwarded. We performed two cases with different mapping cache sizes to capture the amount of time taken for querying source and destination addresses while encapsulating incoming traffic: a first case when the router maintains a small mapping cache with less than 10 entries and a second case with more than 10000 entries. For both cases, we simulate the same traffic condition with more than 1000 incoming packets per second. The experimental results are reported in Fig. 4.5. The median processing time captured at a standard LISP router is around 4 microseconds in case of a small mapping cache, a bit higher than with LISP-EC. In the latter case with a very large mapping cache, we observe the major shift in performance: the median delay experienced with standard LISP is now lower than LISP-EC. The median processing time of LISP-EC capable router is increased from roughly 3000 ns to more than 4000 ns. It indicates the dependence of the novel source address selection with mapping cache size. However, such an overhead can be seen as negligible.

## 4.4 Related LISP control-plane features

Integrating LISP-EC traffic engineering policies in LISP could imply control-plane signaling extensions. Besides the system enhancement we described in the previous section, a LISP operator may see the need to include specific control-plane signaling in support of LISP-EC.

Among the described LISP-EC traffic engineering policies, those purely unilateral one, such as the best source RLOC or best forwarding path policies, rely on local information to

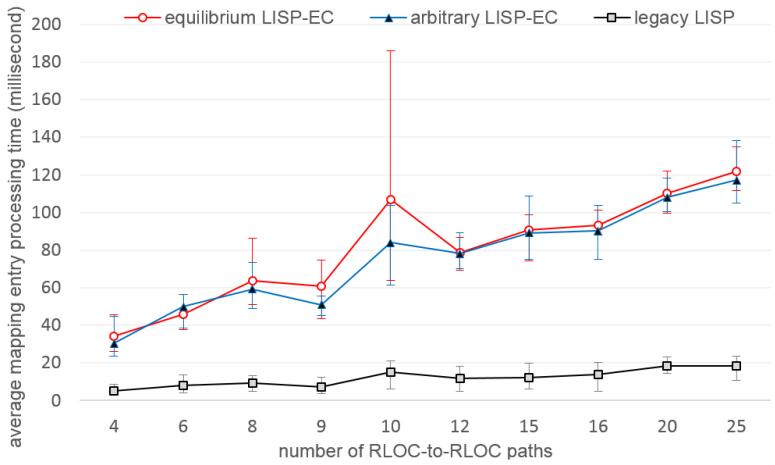



Figure 4.4: Average processing time for adding a mapping entry

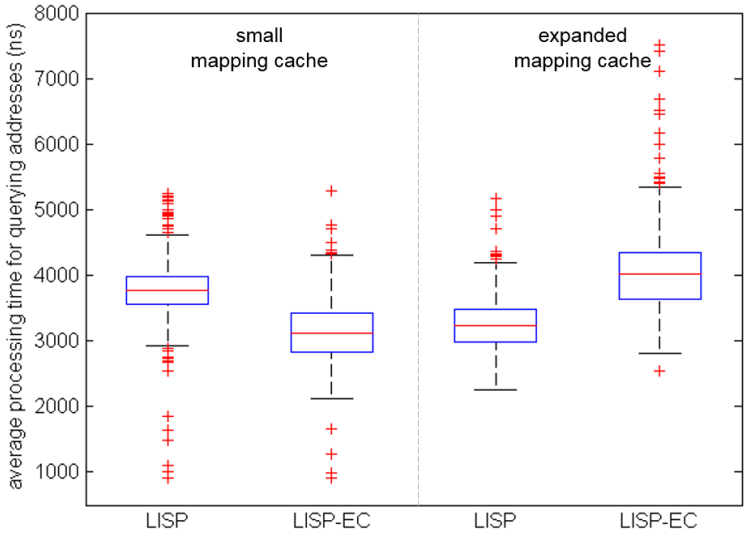



Figure 4.5: Boxplot statistic of look up delay

optimize the outgoing flows of traffic, and LISP-EC specific information exchange between LISP sites is not needed.

Nevertheless, collaborative LISP traffic engineering policies such as the equilibrium and global optimum ones may benefit from a specific control-plane support. As their routing decision does rely on LISP metrics from both sites, it combines the ingress RLOC preferences of the destination network with the egress RLOC preferences of the source network.

Standard LISP distributes RLOC preferences for inbound traffic via three main mapping system messages: MAP-REGISTER, MAP-REQUEST and MAP-REPLY.

We identify two possible modes to disseminate also outbound preferences:

- global outbound preferences dissemination: in this mode, the destination has the same outbound preferences independently of the source. In such a case, MAP-REGISTER messages can be extended to register both inbound and outbound preferences over the local RLOCs, provided the mapping server support such an operation mode. If such an extension is not supported by the mapping system, this could be included only at the ETR-level by extending the MAP-REPLY to also include outbound preferences, provided proxy reply (i.e., the mapping system can reply to MAP-REQUEST messages on behalf of the ETR) is not enabled by the target LISP site.

- source-specific outbound preferences dissemination: in this mode, the destination LISP site wants to reply in a different way as a function of the source LISP site, which is possible when proxy reply is not enabled, hence implementing local TE policies. In such a case, the same extension to the MAP-REPLY message addressed above can be used for this purpose.

The extensions required to MAP-REGISTER and, MAP-REPLY messages are straightforward as the outbound RLOC preferences can be included as additional RLOC objects in the control-plane message structure, and the connotation of the RLOC object (inbound or outbound) can be indicated using a flag in the available 'Reserved' space. In either mode, MAP-REQUEST messages can transport an explicit flag to request outbound RLOC preferences, which can also be taken from the available 'Reserved' space. One could easily add these features to the LIP6-LISP OpenLISP implementation, however excluding the process requiring MAP-REGISTER messages and mapping server interface update as it is a bit more cumbersome. These latter features may indeed become desirable only at a later stage of deployment.

## 4.5 Conclusions

In this work we propose LISP-EC an extended version of LISP for enhancing outbound traffic control. The benefits of LISP-EC over the legacy system is expressed via the capability to balance traffic among upstream providers and the ability to coupling the choice of source and destination locators. Leveraging from the proposed extended design, LISP-EC based traffic engineering solutions show significant improvement in term of delay when comparing with legacy LISP based approaches.

We implemented and released LISP-EC capable OpenLISP-based router at [30]. The implemented system allowed us to study the feasibility of proposed design in the practical network environment, and to verify its interoperability with the existing systems. By comparing the system level performance of LISP-EC enabled router with standard LISP router in realistic emulated networks, we showed that traffic engineering mechanisms emerged from LISP-EC could be deployed at low computation overhead.

# Chapter 5

# Cross-layer equilibrium routing coordination

In this chapter we go beyond the single-layer routing frameworks developed for BGP and LISP systems, for carrier and edge network routing. We define a hierarchical cross-layer game-theoretic framework seeking at controlling and reducing the routing fluctuations arising when both edge and carrier networks concurrently select equilibrium routing solutions in their layer, by means of a cross-layer coordination solution we propose.

## 5.1 Introduction

Within the equilibrium routing game between peering networks, the increase or decrease of peering traffic load can result in a fluctuation of the performance associated with a routing strategy in that game. For example, in the routing game between neighboring Autonomous Systems (ASes) [1], each AS models the performance of a particular routing strategy $s$ as the level of congestion experienced on the peering link $l$ employed by $s$, for sending traffic towards the other peer. The congestion level on $l$ gets changed according to the amount of inter-carrier traffic demand routed on it. Once there is a significant change on the traffic load between these two networks, the performance cost associated with the corresponding strategy could also be changed. As a result of the new cost setting, the routing game cost components have to be updated, resulting in a new multipath equilibrium load balancing decision. The traffic is split among routing paths, more precisely among peering links, once there is a difference between the new and the current load balancing strategy. In other words, elastic traffic load at the peering link interconnection can cause routing instability.

As explained in the previous chapters, the load-balancing distribution over interconnection links is set by the routing game potential threshold. A proper choice of the potential

threshold can cope with routing instabilities caused by the elastic peering traffic loads. Let us recall that rising the threshold above the potential minimum allows extending the equilibria set of the game, thus not only including the strategy profile with minimum potential but also selecting the profiles with potential value smaller or equal than that threshold. Therefore the threshold choice can marginally enhance the path diversity while guaranteeing strategic aspects are presented and while improving routing stability as well. In preceding works, the effectiveness of employing potential threshold to improve routing stability has been proven [1]. In that case, the threshold was set by IGP path cost variations due to transient link failure within an intra domain network, in the frame of a network managed by a distributed link-state routing protocol. In this work, we target the performance cost variation caused by the elastic peering traffic load, taking also in consideration the recent trend toward centralizing the intra-domain route computation with Software Defined Networking. More precisely, we investigate how the choice of the potential threshold in an elastic cross-layer routing context can take place, involving both carrier and edge networks in the routing coordination.

It is worth noting that the variation of traffic load between the two carrier networks $L$ and $L'$ is in fact the result of an increase or decrease of traffic load between their downstream networks. Let $E$ and $E'$ denote two downstream edge networks that connect to $L$ and $L'$, respectively. Changing the upstream load bit rate from $E$ to $E'$ or vice versa results in a (marginal) variation of traffic load between their upstream carrier networks. We adopt in the following a simplified view where two edge networks communication through two carrier networks connected with each other – corresponding to many interconnection situations, or to which one can reduce more articulated interconnection configurations with at least a peering link along the path between two edge networks. Another assumption is that $E$ and $E'$, as well as $L$ and $L'$, coordinate their routing through a multipath equilibrium routing context. Moreover, we consider that the performance metric associated with each routing strategy depends on the load balancing decision of the corresponding upstream carrier, so that an increase or decrease of traffic load could also trigger an update on that edge routing game. Therefore, the variation of traffic load at the edge network level has an effect spreading across layers. Thus, it does not only affect the routing game between networks at the carrier layer but also has an impact on its own routing game at the edge layer. So rather than working on a single routing game at one layer as investigated in previous chapters, we extend the framework targeting the scenario of multiple routing games enabled by (physically or virtually) peering networks across layers.

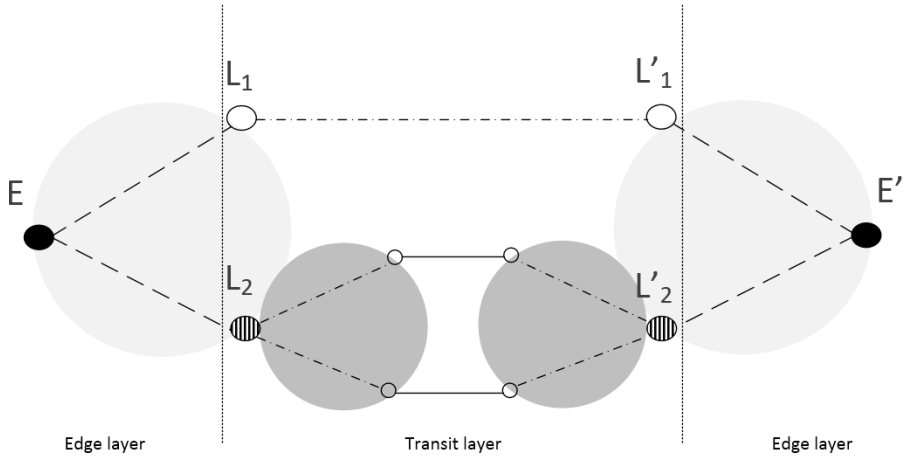



Figure 5.1: Example network scenario

### 5.1.1 Cross-layer routing equilibrium scenario

For a better illustration, we depict our target scenario in Figure 5.1 through an example. For the sake of simplicity, we consider carrier networks with a single downstream edge network. At the transit carrier layer, there are two pairs of peering carrier networks, i.e. $(L_1, L_1')$ and $(L_2, L_2')$. Let us suppose that the first carrier pair $(L_1, L_1')$ does not involve routing decisions using a multipath equilibrium routing logic. Instead, the latter pair, $L_2$ and $L_2'$, does so; as a result, the inter-carrier flows are therein directed by the equilibria of $G_{L_2}$, a notation we use to indicate the routing game between these two carriers.

At the edge layer, the two networks, $E$ and $E'$ are multihomed - they are downstream of $L_1$ and $L_1'$, and $L_2$ and $L_2'$, respectively. Both of them also employ multipath equilibrium routing for their traffic flows. The routing paths for traffic from $E$ to $E'$ and vice versa are therefore impacted by the outcome of the two routing games. First, the edge game $G_E$, i.e., the routing game between $E$ and $E'$. Secondly, the carrier game $G_{L_2}$. The detailed settings as well as the results for each of these peering games are reported in Figures 5.2 and 5.3.

#### Transit routing game

At the transit layer, there are two pairs of peering carriers: $(L_1, L_1')$ and $(L_2, L_2')$. Only one routing game $G_{L_2}$ is established at that layer since we assume the former pair is not routing following a routing game. Therefore $G_{L_2}$ determines how $L_2$ and $L_2'$ route their peering traffic, and for $(L_1, L_1')$ the inter-carrier flows are routed by an inter-domain routing protocol.

We present in Figure 5.2 the network connecting $L_2$ and $L_2'$. For each carrier, there

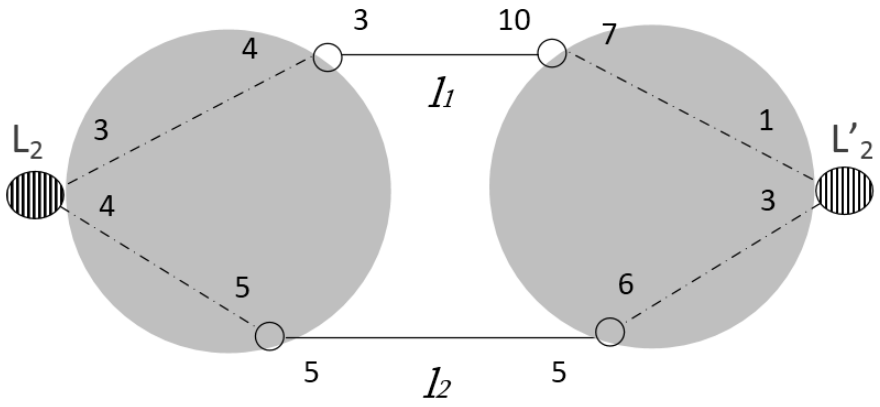



Figure 5.2: Cost settings for the routing game $G_{L_2}$ between carrier $L_2$ and $L'_2$

Table 5.1: Routing game $G_{L_2}$ with $\tau_{L2} = 2$

| $L_2 \backslash L'_2$ | $l_1$ | $l_2$ |
|---|---|---|
| $l_1$ | (17,11)$^4$ | (18,12)$^5$ |
| $l_2$ | **(13,13)**$^0$ | (14,14)$^1$ |

Table 5.2: Resulting load balancing decision

| | $l_1$ | $l_2$ |
|---|---|---|
| $L_2$ | 0% | 100% |
| $L'_2$ | 60% | 40% |

are two options for sending its peering traffic: over $l_1$ or over $l_2$. These two choices are in fact the game strategies. As previously mentioned, $G_{L_2}$ is a cost game, the payoff of each carrier for a strategy profile $(x, y)$ - a combination of strategy $x$ of $L_2$ and strategy $y$ of $L'_2$ - is expressed via a cost function. For carrier $L_2$, we have its cost function $\phi$ defined as follows: $\phi(x, y) = \phi_s(x) + \phi_d(y) + \phi_c(x)$, in which $\phi_s$, $\phi_d$ and $\phi_c$ are the cost function of $L_2$ in its selfish, dummy and congestion game respectively. The selfish game is built upon the egress IGP path cost (from carrier towards peering link), the dummy game is built upon the ingress IGP path cost (inverse direction), and the congestion game is built upon the performance cost on the peering link. Similarly for carrier $L'_2$, we have its cost function $\psi(x, y) = \psi_s(y) + \psi_d(x) + \psi_c(y)$.

Within its congestion game, each carrier models the performance cost of a strategy $l_i$ as the level of congestion over the peering link $l_i$ on the egress direction, i.e., from itself towards the other peer. More specifically, for carrier $L_2$, the performance cost $\phi_c$ assigned to its routing strategy $l_i$ is computed as followed:

$$\phi_c(l_i) = K * \frac{1}{(C_i - p_i)} \tag{5.1}$$

Where $p_i$ is the outgoing flow bit rate on the peering link $l_i$ of carrier $L$. The egress available capacity of the peering link $l_i$ is denoted as $C_i$. If $C_i < p_i$, then $K = \infty$. Otherwise, $K$ is constant to make the performance cost $\phi_c(l_i)$ comparable with other cost components. This is a pretty common congestion cost function [1].

In the example, for carrier network $L_2$, we assume the egress capacity of peering links $l_1$ and $l_2$ to be 100 and 200 units of traffic (e.g., Mbps), respectively. Initially, without $E$-$E'$ edge traffic and with a scaling constant $K = 1000$, the performance cost of strategy $l_1$ and $l_2$ is set to 10 and 5, respectively. For $L'_2$, we assigned an arbitrary value of 3 and 5 as the performance cost associated with strategy $l_1$ and $l_2$, respectively.

With the cost settings given in Figure 5.2, the routing game $G_{L_2}$ between $L_2$ and $L'_2$ is summarized in Table 5.1. Each strategy profile is associated with a pair of cost values, one for $L_2$ and the other for $L'_2$. The strategy profile ($l_1$,$l_2$), for instance, has the cost vector $(18, 12)$ indicating that if carrier $L_2$ selects $l_1$ for routing its peering traffic and $L'_2$ employs $l_2$ for the traffic towards $L_2$, the payoff value of $L_2$ and $L'_2$ for such a decision is 18 and 12 respectively. As discussed in previous chapters, this routing game is also a potential game, and each strategy profile is then associated with a potential value. We report in Table 5.1, besides the payoffs, the potential value of each profile. The choice of potential threshold $\tau$ of the peering game can range from $p_{min} = 0$, i.e., the minimum potential value, to $p_{max} = 5$, i.e. the maximum potential value. With $\tau = 2$ (value for the moment arbitrary chosen), we have the corresponding load balancing decision presented in Table 5.2.

### Edge routing game

At the edge network layer, both peers are dual-homed, so there are two different routing strategies for each edge network. Therefore, in $G_E$, the routing game between $E$ and $E'$, each peer maintains two strategies, and $L_1L'_1$ and $L_2L'_2$ denote the strategies of $E$; the strategy set of $E'$ consists of $L'_1L_1$ and $L'_2L_2$. Similarly to the transit game, the preference of an edge network over a strategy profile $(L_xL'_x, L'_yL_y)$ is expressed via a cost function $\varphi(L_xL'_x, L'_yL_y)$, which is a sum of different cost components, i.e., $\varphi(L_xL'_x, L'_yL_y) = \varphi_s(L_xL'_x) + \varphi_d(L'_yL_y) + \varphi_p(L_xL'_x)$. While the egress cost component reflects the routing (IGP) cost when an edge network decides to forward its peering traffic as given by the routing strategy $L_xL'_x$, the ingress cost component is the routing (IGP) cost when this edge network receives its peering traffic over $L'_yL_y$. Finally, the last cost component is designed to capture the performance of such an edge network when it follows $L_xL'_x$ for sending traffic towards its peer. Differently than the transit game, in the edge routing game, each peer measures the performance of a routing strategy $L_xL'_x$ as the level of path

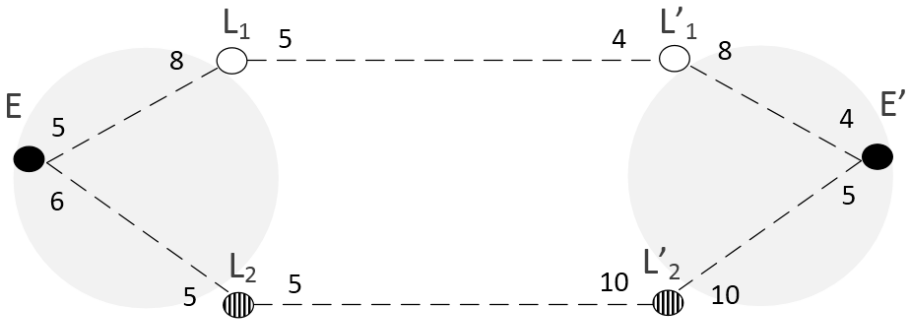



Figure 5.3: Edge networks routing cost setting

Table 5.3: Routing game $G_E$ with $\tau_E = 3$

| $E \setminus E'$ | $L'_1L_1$ | $L'_2L_2$ |
|---|---|---|
| $L_1L'_1$ | **(17,17)**$^0$ | **(18,19)**$^2$ |
| $L_2L'_2$ | **(20,18)**$^3$ | (21,20)$^5$ |

Table 5.4: Resulting load balancing decision

| | $(L_1, L'_1)$ | $(L_2, L'_2)$ |
|---|---|---|
| $E$ | 86% | 14% |
| $E'$ | 71% | 29% |

diversity, as the bottleneck link is not under the management of the player and path diversity plays at this level a more important role for multihomed network as it maps to a level of network availability [2]. The more routes available between the two carriers $L_x$ and $L'_x$, the lower the performance cost of a strategy $L_xL'_x$. More precisely, the performance cost $\varphi_p$ of a strategy $L_xL'_x$ is computed as follow.

$$\varphi_p(L_xL'_x) = A * \frac{1}{N(L_xL'_x)} \tag{5.2}$$

In which $A$ is an arbitrary scaling constant to make the performance cost scalable with other cost components. The function $N(L_xL'_x)$ returns the number of available paths for routing traffic from $L_x$ towards $L'_x$. Consider, for instance, the routing strategy $L_2L'_2$ employed by $E$ for its peering traffic. According to the load balancing strategy presented in Table 5.2, the traffic from $E$ towards $E'$ via the peering carrier pair $(L_2, L'_2)$ is routed on one path, so $N(L_2L'_2) = 1$. Given the scaling constant $A = 10$, the strategy $L_2L'_2$ of $E$ then has the performance cost of 10 (i.e., computed as 10/1). Also referring to the result in Table 5.2, there are two possible ways for routing traffic from $E'$ to $E$ via $L'_2L_2$. So, for $E'$, the cost of strategy $L'_2L_2$ is 5 (i.e., 10/2).

Since there is no routing game taking place between $L_1$ and $L'_1$, we assume an ar-

bitrary cost of 4 as the performance cost associated with strategy $L_1L'_1$ for $E$ and 5 as performance cost of strategy $L'_1L_1$ for $E'$. Together with the ingress and egress IGP path costs associated with each routing strategy as given in Figure 5.3, we have the routing game $G_E$ between $E$ and $E'$ summarized in Table 5.3. With a choice of 3 as its potential threshold (value for the moment arbitrary chosen), the resulting load balancing decision for each peer is reported in Table 5.4. Edge network $E$ splits 86% of its peering traffic demand over the upstream provider $L_1$, and the remaining 14% is shared by $L_2$. With 71% of demand sending over $L'_1$ and 29% over $L'_2$, both transit paths are also employed by $E'$.

### 5.1.2 Dealing with traffic load variations

Continuing with our reference example, let us consider an arbitrary traffic load of 400 units of traffic (e.g., Mbps) for a flow from $E$ to $E'$. According to its load balancing strategy described in Table 5.4, $E$ distributes only 14% of that load over $L_2$. Such a load introduces an inter-carrier flow of 56 volume from $L_2$ to $L'_2$. Following the computed load balancing ratio on Table 5.2, $L_2$ directs its peering traffic over $l_2$. Consequently, in the routing game $G_{L_2}$, the performance cost associated with strategy $l_2$ of carrier $L_2$ rises from 5 to 7 as the traffic rate on it increases to 56. The remaining cost components are unchanged. The new cost settings are updated in Figure 5.4, with the corresponding routing game reported in Table 5.5 and the new load balancing decision given in Table 5.6. Instead of employing $l_2$ for routing its peering traffic as before, carrier $L_2$ now sends 36% of its load via $l_1$.

The routing decision change at upstream carrier $L_2$ has an impact on the routing game $G_E$ of the downstream edge networks. Thus, by employing two instead of only one path for the traffic towards $L'_2$, the performance cost associated with strategy $L_2L'_2$ of edge network $E$ is dropped from 10 to 5 as the path diversity increases from 1 to 2. The updated cost settings for routing strategies in $G_E$ is presented in Figure 5.5, with the corresponding game in Table 5.7, and the resulting load balancing decision in Table 5.8.

A comparison between the load balancing strategy of peering carrier networks before and after adjusting their traffic load is reported in Tables 5.2 and 5.6, respectively; we observe that 36% of the traffic demand on the egress direction (from $L_2$ to $L'_2$) is moved from one peering link to the other. In other words, $L_2$ experiences a 36% traffic shift. For the case of carrier $L'_2$, a traffic shift of 3% is obtained. At the edge layer, $E$ experiences a traffic shift of 61%, and on the reversed direction, a traffic shift of 4% is obtained for $E'$. In Table 5.9 we summarize the percentage of traffic shift experienced by each network at both transit and edge layers when their choice of potential threshold is 3 and 5, respectively.

Therefore, by adjusting its potential threshold, an edge network can mitigate the prob-

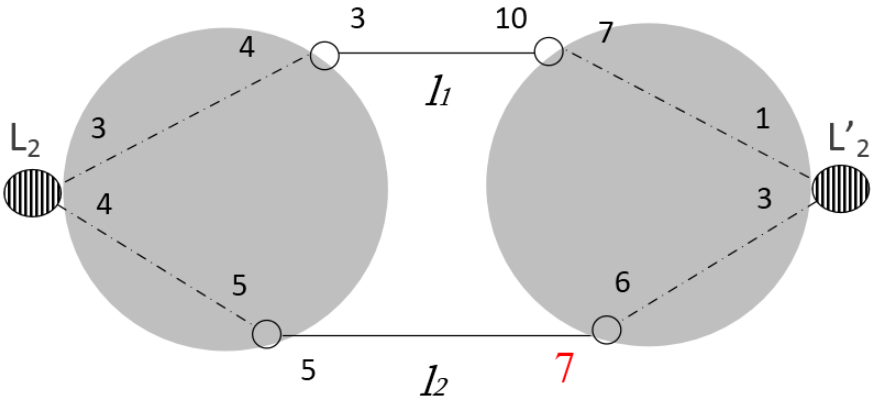



Figure 5.4: Updated cost settings for transit game $G_{L_2}$

Table 5.5: Resulting routing game $G_{L_2}$ with $\tau_{L2} = 2$

| $L_2 \backslash L_2'$ | $l_1$ | $l_2$ |
|---|---|---|
| $l_1$ | **(17,11)**$^0$ | **(18,12)**$^1$ |
| $l_2$ | **(15,13)**$^{-2}$ | **(16,14)**$^{-1}$ |

Table 5.6: Resulting load balancing decision

| | $l_1$ | $l_2$ |
|---|---|---|
| $L_2$ | 36% | 64% |
| $L_2'$ | 57% | 43% |

lem of peering traffic load variation, by dramatically reducing the amount of traffic shift. Similarly, for the peering networks at the transit layer, there is an incentive for $L_2$ to take a different choice of $\tau_{L_2}$ to reduce the amount of traffic shift among its peering links. According to the customer-provider relationship between the edge network $E$ and carrier network $L_2$, the amount of traffic traffic shift experienced by $L$ and $E$ depends on both choices of $\tau_E$ and $\tau_{L_2}$). The requirement logically raising is that the interactions between $E$ and $L$ for determining their potential threshold could be modeled as a non-cooperative game, as well.

| | $(\tau_{L_2} = 2, \tau_E = 3)$ | $(\tau_{L_2} = 2, \tau_E = 5)$ |
|---|---|---|
| $E$ | 61% | 0 |
| $E'$ | 4% | 0 |
| $L_2$ | 36% | 100% |
| $L_2'$ | 3% | 0 |

Table 5.9: The percentage of traffic shift on peering networks for different combinations of threshold choice

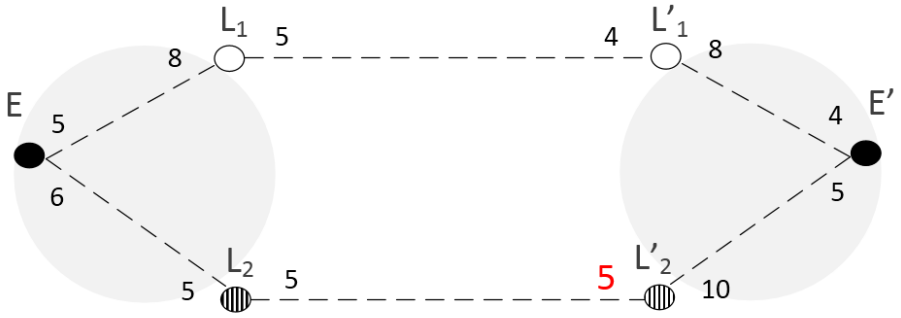



Figure 5.5: Updated routing cost setting for edge network game $G_E$

Table 5.7: Resulting routing game $G_E$ with $\tau_E = 3$

| $E \backslash E'$ | $L'_1L_1$ | $L'_2L_2$ |
|---|---|---|
| $L_1L'_1$ | **(17,17)**$^2$ | (18,19)$^4$ |
| $L_2L'_2$ | **(15,18)**$^0$ | **(16,20)**$^2$ |

Table 5.8: Updated load balancing decision

| | $(L_1, L'_1)$ | $(L_2, L'_2)$ |
|---|---|---|
| $E$ | 25% | 75% |
| $E'$ | 75% | 25% |

### 5.1.3 Toward a potential threshold non-cooperative game modeling

At this stage, it should be clear that in such a cross-layer routing context, a proper choice of the potential threshold could reduce the amount of traffic shift once the peering traffic load is adjusted. For instance, considering the edge network game $G_E$ presented in the example, with 400 traffic units (e.g., Mbps) as the predicted peering traffic load variation, $E$ could fine-tune its choice of $\tau_E$ to minimize the amount of traffic fluctuated among its upstream carriers. As reported in Table 5.9, if the potential threshold of 5 is chosen instead of 3, the ratio of traffic shift among the routing paths of $E$ could be dropped dramatically, from 61% to 0%. This choice of potential threshold could yield the best solution for the peering networks at the edge layer since there is no traffic shifted experienced by both peers; however, it causes a higher quantity of traffic shift at the upstream carrier networks. In fact, the percentage of traffic shift experienced by $L_2$ is increased from 36% to 100% when $E$ changes its decision. More importantly, a different choice of $\tau_{L_2}$ in the upstream carrier game $G_{L_2}$ could result in a different amount of traffic shifted in $E$.

In our network model, when a equilibrium multipath routing is adopted by peering networks, the amount of traffic shift experienced by one peer is not only determined by

| $\tau_E \backslash \tau_{L_2}$ | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | (0,0) | (0,0) | (0,0) | (0,0) | (100,90) | (100,79) |
| 1 | (0,0) | (0,0) | (0,0) | (0,0) | (100,90) | (100,79) |
| 2 | (0,0) | (0,0) | (0,0) | (0,0) | (80,90) | (80,79) |
| 3 | (55,25) | (61,30) | (61,36) | (61,39) | (61,90) | (61,79) |
| 4 | (0,100) | (47,38) | (47,42) | (47,44) | (47,90) | (47,79) |
| 5 | (0,100) | (0,100) | (0,100) | (33,88) | | |
| 6 | (0,100) | | | | | |
| 7 | (0,100) | | | | | |
| 8 | (0,100) | | | | | |
| 9 | (0,100) | | | | | |
| 10 | (0,100) | | | | | |

Table 5.10: A threshold non-cooperative game setting. Note that in this game form some cells are empty as the strategy set of the second player ($\tau_{L_2}$) depends on the strategy taken by the first player ($\tau_E$).

the potential threshold choice in its routing game, but it also be effected by the choice of potential threshold made by other networks in their own routing game when they do change their routing strategy using a routing game. More precisely, the percentage of traffic shift experienced by $E$ or $E'$ is not only defined by their choice of potential threshold in $G_E$ or $G'_E$, but it is also determined by the thresholds choice $\tau_{L_2}$ of $L_2$ in $G_{L_2}$. In other words, for a pair of peering networks in our model, the amount of traffic shift it experienced depends on a combination of threshold choices made by every pair of networks playing the same type of routing game.

In Table 5.10, we enumerate all the possible combinations of threshold choices that could be made by the peering edge network $E$ and the peering carrier network $L2$ in their routing game $G_E$ and $G_{L_2}$, respectively. Along with each combination of choices, we report the corresponding percentage of traffic shift that would be experienced by both $E$ and $L_2$, accordingly. The reported results is relying on the same settings and configuration discussed in the previous example. For instance, with the threshold choices of $\tau_E = 3$ and $\tau_{L2} = 4$, $E$ experiences a shift of 61% on the total peering traffic demand and at $L_2$ a traffic shift of 90% is captured on its peering links.

With the aim to reduce the traffic shift upon traffic load variation, $E$ could fine tune its potential threshold $\tau_E$. However, the amount of traffic experienced by $E$ is also impacted by the choice of $\tau_{L_2}$ in $G_{L_2}$. Similarly, to decrease the amount of traffic shift in $L_2$ when the inter-carrier traffic load changes, $L_2$ needs to consider not only its potential threshold $\tau_{L_2}$, but also the potential threshold $\tau_E$ of $E$. Therefore, the threshold value implicit selection process between $E$ and $L_2$ can be modeled as a non-cooperative game.

In the following section we formalize the resulting threshold game suggested by the

previous multi-stage example, which we propose as the means to allow cross-layer light-way coordination between edge and transit networks playing multipath equilibrium routing.

## 5.2 Problem Formulation

In the following, we describe the threshold game. We first describe the basic notations (also summarized in Table 5.11), and then we provide a formal definition of the game.

### 5.2.1 Notations

Let $(L_x, L'_x)$ be a pair of two peering carrier networks $L_x$ and $L'_x$. Within a carrier pair $(L_x, L'_x)$ that uses multipath equibrium routing, the traffic flow from $L_x$ towards $L'_x$ is split over $N_{L_x}$ peering links accordingly to the presented routing game solution computation.

To avoid confusion we define $L_x(l_i)$ as the peering link $l_i$ of carrier $L_x$ and denote $H_{L_x}$ as the set of these peering links. The load balancing ratio on a link $l_i$ is then denoted by $f_{L_x(l_i)}$; such ratio is determined by the load balancing vector $f_{L_x} = (f_{L_x(l_1)}, f_{L_x(l_2)}, ..., f_{L_x(l_{N_x})})$ resulted from $G_{L_x}$, the routing game between the two carriers $L_x$ and $L'_x$. For a peering link $l_i$ of $L_x$, its load balancing ratio $f_{L_x(l_i)}$ determines the percentage of total traffic that carrier $L_x$ transmits on it. Similarly, let $f_{L'_x} = (f_{L'_x(l_1)}, f_{L'_x(l_2)}, ..., f_{L'_x(l_{N_x})})$ be the vector employed by carrier $L'_x$ for load balancing the traffic towards its peer.

In the resulting transit routing game, both carriers therefore maintain the same number of strategies. The level of congestion experienced by a carrier network when sending traffic over a link is then modeled as the performance cost of the corresponding routing strategy. More precisely, in $G_{L_x}$, $L_x$ assigns to its strategy $l_i$ a performance metric $\phi_c(L_x(l_i))$ computed as follow:

$$\phi_c(L_x(l_i)) = K_{L_x} * \frac{1}{C_{L_x(l_i)} - p_{L_x(l_i)}} \tag{5.3}$$

Where $p_{L_x(l_i)}$ is the outgoing traffic bit rate from $L_x$ to $L_x$ via link $l_i$. The available capacity of link $l_i$ is denoted as $C_{L_x(l_i)}$. If $C_{L_x(l_i)} < p_{L_x(l_i)}$, then $K_{L_x} = \infty$. Otherwise, $K^{L_x}$ is a normalization factor making the performance cost $\phi_c(L_x(l_i))$ at the same scale than other cost components in $G_{L_x}$.

$L_x$ is an upstream provider of a multihomed edge network $E$, while $L'_x$ is an upstream provider of $E'$, another multihomed edge network. There are $M$ carrier pairs at the transit layer able to connect the two edge networks; however, only one pair, i.e., $(L_x, L'_x)$, is supposed to manage its routing via multipath equilibrium computation. More importantly, both $E$ and $E'$ agree on the equilibrium multipath routing solution for the traffic flows

between them. Therefore the traffic from $E$ to $E'$ and vice versa can be distributed to multiple carrier pairs.

We denote $r_E$ as the load (e.g., in bit/s) for the traffic flow from $E$ to $E'$; accordingly to $f_E$ (the load balancing vector resulting from $G_E$), the edge network $E$ splits its demand $r_E$ over $M$ transit paths. More precisely, we have the vector $f_E = (f_E^{L_1}, f_E^{L_2}, ..., f_E^{L_M})$ in which $f_{Lx}^E$ denotes the percentage of $r_E$ that is routed by carrier $L_x$. Similarly, $E'$ employs the load balancing vector $f_{E'} = (f_{E'}^{L'_1}, f_{E'}^{L'_2}, ..., f_{E'}^{L'_M})$ for the load $r_{E'}$ of traffic on the direction towards $E$.

In the routing game $G_E$, a directional transit path connecting $E$ and $E'$ forms a routing strategy. Between these edge networks, there are $M$ pairs of peering carriers connecting them; therefore, each edge network maintains a set of $M$ routing strategies. In order to construct the edge routing game $G_E$, the performance of a routing strategy is modeled as the number of paths for routing traffic from one carrier toward the other. Thus, the performance cost $\varphi_p(L_x L'_x)$ assigned by edge network $E$ for its routing strategy $L_x L'_x$ is computed as follows:

$$\varphi_p(L_x L'_x) = K_E * \frac{1}{N(L_x L'_x)} \tag{5.4}$$

In which $K_E$ is an arbitrary normalization factor defined by $E$ to make the performance cost in the same scale than the other cost components in $G^E$. $N(L_x L'_x)$ gives the number of available paths for routing traffic from $L_x$ to $L'_x$.

Traffic from $E$ to $E'$ via $L_x$ brings a load $r_E^{Lx}$ which is the product of the peering traffic load $r_E$ and the corresponding load balancing ratio $f_E^{Lx}$ on the transit path via $L_x$, i.e., $r_E^{Lx} = r_E * f_E^{Lx}$. In our simplified model, there is only one pair of PEMP-enabled peering carriers at the transit layer, and carriers in that pair are only responsible for routing traffic between their downstream networks at the edge. Thus, for a coordinated carrier pair $(L_x, L'_x)$ at the transit layer, the flow from $E$ to $E'$ is the only inter-carrier flow, therefore $r_E^{Lx} = p_{Lx}$.

For a better illustration, we depict in Figure 5.6 the two edge networks $E$ and $E'$ with their two pairs of upstream carrier networks $(L_1, L'_1)$ and $(L_2, L'_2)$. Among these pairs, $(L_2, L'_2)$ is the only one performing multipath equilibrium routing. According to the load balancing vector $f_E$ resulting from $G_E$, the traffic load $r_E$ from $E$ to $E'$ is split over two upstream carriers $L_1$ and $L_2$. The traffic load from $E$ over the path via $L_1$ and $L_2$ is $r_E^{L_1}$ and $r_E^{L_2}$, respectively. Since $E$ is the only downstream of $L_2$, $r_E^{L_2}$ the outgoing traffic load from $L_2$ toward its peer is $r_E^{L_2} = p_{L_2}$. Following $f_{L2}$, the load balancing vector resulting from $G_{L_2}$, $p^{L_2}$ is split over the two peering links. The load on the first and second link is $p_{L_2(l_1)}$ and $p_{L_2(l_2)}$, respectively.

| Notation | Description |
|---|---|
| $L_x$ | a carrier network |
| $L'_x$ | a carrier network that peers with $L_x$ |
| $G_{L_x}$ | coordinated routing game between $L_x$ and $L'_x$ |
| $\tau_{L_x}$ | the potential threshold of the game $G_{L_x}$ |
| $N_{L_x}$ | number of peering links between $L_x$ and $L'_x$ |
| $H_{L_x}$ | set of peering links between $L_x$ and $L'_x$ |
| $L_x(l_i)$ | peering link $l_i$ of carrier $L_x$ |
| $p_{L_x}$ | traffic load from $L_x$ to $L'_x$ |
| $f_{L_x(l_i)}$ | percentage of peering traffic load that $L_x$ sends over $l_i$ |
| $f_{L_x}$ | load balancing vector of $L_x$, $f_{L_x} = (f_{L_x(l_1)}, f_{L_x(l_i)}, ..., f_{L_x(l_{N_x})})$ resulting from $G^L$ |
| $p_{L_x(l_i)}$ | load on peering link $l_i$ of $L_x$, $p_{L_x(l_i)} = p_{L_x} * f_{L_x(l_i)}$ |
| $C_{L_x(l_i)}$ | available capacity of the peering link $l_i$ of $L_x$ |
| $\phi_c(L_x(l_i))$ | performance cost associated with the peering link $l_i$ of $L_x$ |
| $K_{L_x}$ | scaling factor for the performance cost wrt other $G_{L_x}$ cost components |
| $E$ | a multihomed edge network downstream of carrier $L$ |
| $E'$ | a multihomed edge network that peers with $E$, downstream of $L'$ |
| $G_E$ | routing game between $E$ and $E'$ |
| $\tau_E$ | the potential threshold of the game $G_E$ |
| $M$ | the number of carrier paths connecting $E$ and $E'$ |
| $U_E$ | set of upstream carriers of $E$ |
| $r_E$ | traffic load from $E$ to $E'$ |
| $f_E^{L_x}$ | percentage of peering traffic load that $E$ sends to $L_x$ |
| $f_E$ | load balancing vector of $E$, $f_E = (f_E^{L_1}, f_E^{L_2}, ..., f_E^{L_M})$ resulting from $G^E$ |
| $r_E^{L_x}$ | load via carrier $L_x$, $r_E^{L_x} = f_E^{L_1} * r_E$ |
| $N(L_xL'_x)$ | the number of available paths for routing traffic from $L_x$ to $L'_x$ |
| $\varphi_p(L_xL'_x)$ | performance cost associated with the path via $L_xL'_x$ of $E$ |
| $K_E$ | scaling factor for the performance cost wrt other $G_E$ cost components |
| $G$ | the threshold game between $E$ and $L_x$ |
| $T_{L_x}$ | the strategy set of $L_x$, i.e. the set of all potential threshold choices in $G_{L_x}$ |
| $\delta_{L_x}$ | the cost function of $L_x$ in the threshold game $G$ |
| $g_{L_x}$ | the function assigning the load balancing ratio to peering links |
| $T_E$ | the strategy set of $E$, i.e. the set of all potential threshold choices in $G_E$ |
| $\theta_E$ | the cost function of $E$ in threshold game $G$ |
| $g_E$ | the function assigning the load balancing ratio to carrier path |
| $R(\tau_{L_x})$ | the set of optimal responses of $E$ for strategy $\tau_{L_x}$ of $L_x$ |

Table 5.11: Mathematical notations

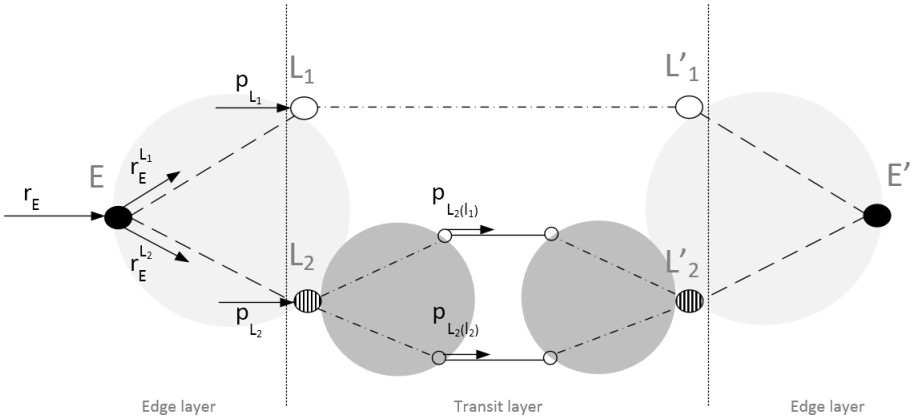



Figure 5.6: A example of 1 edge network pair connecting via 2 pairs of peering carrier

### 5.2.2 Threshold game

In the multipath equilibrium routing frameworks presented in the previous chapters, a network makes its choice of potential threshold $\tau$ before constructing the routing cost game, and choice that does not need to be coordinated with the other peering network. Such a choice of $\tau$ has an impact on the set of equilibria; therefore different choices of $\tau$ could result in different load balancing vectors. Considering $G^{L_x}$, the load balancing vector $f_{L_x}$ varies accordingly to the value of $\tau_{L_x}$ decided by $L_x$. Besides that, the load balancing ratio $f_{L_x(l_i)}$ on a peering link $l_i$ of $L_x$ is also determined by the performance cost $\phi_c(L_x(l_i))$, and hence the potential value, as explained in the previous chapters. In other words, $f_{L_x(l_i)}$ could be expressed as a function of $\tau_{L_x}$ and $\phi_c(L_x(l_i))$. Let $T_{L_x}$ denote the set of all potential threshold choices available in the game $G_{L_x}$, and let $g_{L_x} : T_{L_x} \times \mathbb{R} \to \mathbb{N}$ be a function assigning the load balancing ratio to peering links, i.e., $f_{L_x(l_i)} = g_{L_x}(\tau_{L_x}, \phi_c(L_x(l_i)))$. Likewise, in $G_E$ let $T_E$ denote the set of all possible choices of $\tau_E$, and let $g_E : T_E \times \mathbb{R} \to \mathbb{N}$ be a function assigning the load balancing ratio $f_E^{L_x}$ of a carrier path via $L_x$ as a function of the values of $\tau_E$ and $\varphi_p(L_x L'_x)$. Thus, $f_E^{L_x} = g_E(\tau_E, \varphi_p(L_x L'_x))$.

As previously mentioned, supposing both edge and transit routing games are played, the amount of traffic shift over the peering links does not only depend on the threshold choice $\tau_{L_x}$ of $L_x$ in the transit routing game, but it is also determined by the potential threshold $\tau_E$ in the edge routing game. This is also true for the amount of traffic experienced by $E$ when its traffic demand toward $E'$ changes. In such a strategic context, rationality assumption implies that both a carrier and its downstream edge network fine tune their choices of potential threshold to reduce the amount of traffic shift. The resulting non binding cross-layer interaction between them can be modeled as a threshold non-cooperative game.

The threshold game can be defined as $G(L_x, E; T_{L_x}, T_E; \delta_{L_x}, \theta_E)$ in which $L_x$ and $E$ are the two player sets; $T_{L_x}$ and $T_E$ their strategy sets, so that each strategy $\tau_{L_x} \in T_{L_x}$ and $\tau_E \in T_E$ indicates a threshold choice; $\delta_{L_x}$ denotes the cost function of $L_x$, and $\theta_E$ the cost function of $E$ in $G$. In the following, we define these cost functions in more detail.

Given a routing game involving $(L_x, L'_x)$, when the inter-carrier traffic load changes, the objective of $L_x$ to reduce the amount of traffic shift among its peering links can be defined as a cost function $\delta_{L_x} : T_{L_x} \times T_E \to \mathbb{N}$ computed as:

$$\delta_{L_x}(\tau_{L_x}, \tau_E) = \max_{l_i \in H_{L_x}} |\hat{f}_{L_x(l_i)} - f_{L_x(l_i)}| \tag{5.5}$$

Where $f_{L_x(l_i)}$ and $\hat{f}_{L_x(l_i)}$ denotes the load balancing ratio on the peering link $l_i$ of $L_x$ before and after the inter-peering traffic demand change, respectively.

At the edge network layer, the cost function $\theta_E : T_E \times T_{L_x} \to \mathbb{N}$ is defined to express the objective of $E$ to minimize the amount of traffic shift among its upstream carrier paths; we have:

$$\theta_E(\tau_E, \tau_{L_x}) = \max_{L_x \in U_E} |\hat{f}_E^{L_x} - f_E^{L_x}| \tag{5.6}$$

Where $\hat{f}_E^{L_x}$ and $f_E^{L_x}$ denote the load balancing ratio on the carrier path via $L_x$ before and after the variation of traffic load $r_E$ from $E$ towards $E'$.

Taking into account the hierarchical nature of the cross-layer decision-making framework, our threshold game $G(L_x, E; T_{L_x}, T_E; \delta_{L_x}, \theta_E)$ is a form of Stackelberg [40] or leader-follower game [41], in which the carrier network $L_x$ plays the role of a leader and edge network $E$ acts as its follower (and likewise for $L'_x$ and $E'$). Let $R(\tau_{L_x}) \subset T_E$ denotes the set of optimal responses of the follower (edge network) for each strategy choice $\tau_{L_x}$ made by the leader (carrier network) $L_x$.

In the resulting leader-follower game, for leader $L_x$, a strategy $\tau^*_{L_x} \in T_L$ is called a (Stackelberg) threshold equilibrium strategy if

$$\max_{\tau_E \in R(\tau^*_{L_x})} \delta_{L_x}(\tau^*_{L_x}, \tau_E) = \min_{\tau_{L_x} \in T_L} \max_{\tau_E \in R(\tau_{L_x})} \delta_{L_x}(\tau_{L_x}, \tau_E) \tag{5.7}$$

## 5.3 Conclusions

We draw in this chapter the natural evolution of the mathematical modeling of the routing problems addressed in the previous two chapters. Open works in this topic are first the numerical simulation of the proposed cross-layer equilibrium routing framework and then its experimental evaluation through implementation in real open-source systems.

# Chapter 6

# Multipath strategies for Internet security: a measurement study

Multipath communications at the Internet scale have been a myth for a long time, with no actual protocol being deployed at large scale. In the previous chapters we discussed how one can enhance existing routing systems at the network IP layer (i.e., BGP, LISP) to explicitly select Internet paths to assign to aggregate of application flows, and even single flows. Recently, the Multipath Transmission Control Protocol (MPTCP) extension was standardized and is undergoing rapid adoption in many different use-cases, from mobile to fixed access networks, from data-centers to core networks. Its adoption by the Apple iOS is available and under completion, and the adoption by the Linux kernel is forthcoming. Among its major benefits – i.e., reliability thanks to backup path rerouting, throughput increase thanks to link aggregation, and confidentiality being more difficult to intercept a full connection – the latter has attracted lower attention.

In this chapter we investigate how explicit multipath forwarding strategies can enhance Internet connection confidentiality. We take as primary reference technology the one of MPTCP as it focuses on single connection rather than on IP aggregate, and because it may not require network support, but our investigation also covers network configurations with a forwarding protocol operating at the network edges and able to explicitly select paths at the transport layer flow level such as those discussed in the previous chapters. We want to determine how robust can MPTCP, or such explicit flow-level forwarding protocols, be to exploit multiple Internet-scale paths and decrease the probability of Man-in-the-Middle (MITM) attacks. By analyzing the Autonomous System (AS) level graph, we identify which countries and regions show a higher level of robustness against MITM AS-level attacks, for example due to core cable tapping or route hijacking practices.[1]

[1]The content of this article was published in [42]. An extended version was submitted to Elsevier

## 6.1 Introduction

The Multipath Transmission Control Protocol (MPTCP) [8] is an extension of TCP to concurrently use multiple network paths for a given connection. Among many proposals to support these features at the transport layer, MPTCP is considered as the one having attracted the largest interest and deployment [43]. One of the main reasons for this success is the incremental deployability adopted in its design, with the required signaling transparently reusing existing features of the TCP options.

As already detailed in Chapter 2.6, MPTCP employs multiple 'subflows' to route traffic from a source to a destination in an IP network via different network interfaces and/or TCP ports at the transmitting and/or receiving endpoints. Subflow IP traffic can then be routed independently in the network segment. However, besides the usage of multiple network interfaces at the source or destination, the presence of flow-level load-balancers sensible to port numbers,or multipath proxies aware of the network topology [44] can differentiate the route followed by the subflow packets.

Among the motivations pushed forward in support of MPTCP, there are (i) bandwidth aggregation, i.e., the increased network bandwidth offered to a connection; (ii) connection reliability, i.e., the possibility to use an alternative path in case of failure along the primary path or at the primary network interface level; (iii) communication confidentiality, i.e., the decreased ability for a Man-in-the-Middle (MITM) attacker to intercept all the traffic of a same connection. While the first two aspects above have been largely explored in the last decade, the latter was marginally studied to date. In this chapter, we report the results of an extensive measurement campaign aimed at assessing the degree of confidentiality one can expect using MPTCP. In particular, we focus on confidentiality from large-scale, i.e., Autonomous System (AS) level, MITM interception, i.e., looking at the empirical probability that a single connection can be intercepted by an organization or an attacker able to capture all the traffic going through an AS on a given direction (most of Internet communications being asymmetric). Such attacks can happen either by remote access to routing devices of an AS or even by Border Gateway Protocol (BGP) route hijacking.

In our analysis, we focus on the case of MPTCP-capable source devices using two edge providers, analyzing measurement results on a geographical basis to identify which countries and regions MPTCP may grant higher confidentiality with respect to large-scale MITM threats.

An important assumption of our analysis is that the MPTCP scheduler behavior of endpoints or multipath converters can be tuned so that it does not only look for throughput maximization, but also for path diversity exploitation for increased confidentiality, as

investigated in [45]. Solutions offering programmability of the MPTCP scheduler are making surface, as notably [46, 47].

It is worth noting that, despite we refer to MPTCP as our reference multipath transport-layer protocol, our study can apply as well to other functionally equivalent protocols, such as for instance multipath QUIC (Quick User Datagram Protocol Internet Connections) [48].

## 6.2 Internet MITM Attacks

In Internet-scale communications, MITM attacks can happen when the attacker gains access to all the traffic transiting through an AS, or at least a portion of it that is enough to reconstruct the transmitted data. In practice, it can be possible by optical layer or BGP route hijacking MITM attacks.

At the optical layer, an attacker is able to split cables by using fiber optical taps, as described in [49], with a low probability of being detected if peculiar strategies are adopted as explained in [50, 51]. Moreover, one can intercept the traffic by exploiting coupling and out-of-the-fiber light propagation phenomena [52], despite the fact that this is particularly challenging when performing wavelength-division-multiplexing.

At the BGP layer, MITM attacks exploit the natural way BGP works, stealthily hijacking Internet routes to modify or capture the traffic before it reaches the destination. These BGP-based MITM attacks have been quite deeply studied for about twenty years; in a recent survey [53] we have a detailed description of such attacks, their effects as well as the mitigation and defense strategies. This type of attack gained special attention in 2008, when a major provider in central Asia hijacked Youtube traffic to apply local policies. In the same year, a practical BGP MITM attack was demonstrated during the DefCon hacking conference [54]: authors successfully intercepted traffic bound for the conference network and redirected it to a system they controlled before routing it back to DefCon. A recent notable attack happened in 2014, attackers injected BGP routes to redirect traffic from Bitcoin miner nodes to a compromised host [55]; it was estimated that at least $83,000 worth of Bitcoins, Dogecoins, HoboNickels, and Worldcoins were stolen over a period of four months. More recently, in 2017 all traffic heading to Visa, MasterCard and other service providers was hijacked for a short period of few minutes [56]. The actual cost of such BGP incidents could be even more than what have been reported. Notable ones are documented in [57, 58]; often they are not reported because they cannot be always detectable, they have limited scope, last for a short time etc.

At the transport layer, the advent of MPTCP raised new security specification questions and challenges [59, 60]. In [61], cryptography based solutions are proposed against

eavesdropping. The authors in [60] present an analysis of residual threats in the MPTCP signaling and also propose some fixes. Recently, an extension of MPTCP to secure multipath communications was proposed in [62], offering authentication and encryption mechanisms not only to the connection but also to single TCP options. This prevents different types of MITM attacks where an attacker could force all the traffic to be sent only over the path under his control by hijacking the traffic and erasing the MP_CAPABLE option.

In general, most of the works at the state of the art aim at either investigating security threats for MPTCP or proposing solutions for them. It is worth mentioning the rising interest in using MPTCP to further enhance confidentiality when using Internet over-the-top Virtual Private Networks (VPN) services such as ToR and OnionCat [63]: MPTCP is used in the upstream direction from the client to many gateways accessible via the VPN, on the way to the server, thus increasing the confidentiality level of the connection. Nevertheless, such practices can have a gain which can be hard to assess: how can you ensure the upstream source-destination traffic does follow disjoint paths, hence decreasing MITM efficiency, if not at the router-level, at the AS level? In this work, for the first time at the state of the art to the best of our knowledge, we attempt to provide a response to such questions.

## 6.3 Methodology

In this section, we first give a description on the datasets used for constructing a representative AS-level graph of Internet, the basis for our analysis. Then, we describe our approach for computing the number of valid vertex-disjoint paths between two arbitrary nodes over the constructed graph. Finally, we detail how we evaluate path diversity at different geographical scopes. The datasets we employed as well as our scripts are given in [64] for the sake of reproducibility.

### 6.3.1 Graph construction

We extract 2015 data from [65], the latest dataset available, couple the AS-level topology with the inter-AS relationship data to form a new dataset containing all the AS links along with their frequency of occurrence and relationship type. Comparing with other resources [66] [67], the topological data extracted from [65] revealed to be more reliable and able to capture a broadened view of the Internet topology. Indeed, it integrates data not only from Routeviews [68], but also from other resources such as RIPE RIS [69]. Moreover, the traceroute-based approach employed in [66] has known issues [70] when converting router-level paths into AS-level. The inter-AS relationship data from [65] is

extracted monthly from the Cyclops database [71], which combines BGP data with Internet eXchange Point (IXP) data and adopts inference techniques proposed in [70].

Employing measurements over a long period allows us to capture inter-domain connection dynamics as well as inter-AS economic relationships. For instance, in a one month period, only 85% of inter-AS links appear more than 20 days, the remaining links with lower frequency of occurrence being those used for backup operations or during BGP convergence periods. For the sake of consistency, we removed these unstable links.

### 6.3.2 Path diversity computation

The problem with selecting all the paths connecting two nodes over a graph that satisfies given routing properties is often referred to as policy-compliant path diversity computation in the literature [72, 73]. The common approach [72] is to convert the original graph into a type-of-relationship (ToR) graph [74], i.e., a directed graph in which the relationship between two adjacent vertexes is expressed via the direction of the edge connecting them, then maximizing the total number of vertex-disjoint paths between nodes in this graph. However, the time-complexity experienced in such methods is relatively high hence intractable for a graph as big as the AS graph.

We introduce a novel path search algorithm leveraging the scale-free characteristics [75] of the input AS graph (i.e., a graph with relatively few hubs capturing the majority of the paths) to optimize the execution time. In such a scale-free graph, the diameter (i.e., the length of the longest path among all the shortest paths) is not too high. Thus, the average path length (measured in number of AS hops) connecting any pair of nodes in the AS-level graph of Internet is around 5 as of today [76] (a bit lower with IPv6).

Searching for paths in a scale-free graph with a reasonable diameter is not a too complex problem when adopting breadth/depth-first search algorithms with a limited depth. From the constructed AS graph $G$, the breadth-first search algorithm in Alg. 1, can be applied to discover all the policy-compliant paths between two nodes $s$ and $d$, in a reasonable time. Starting from the origin $s$, the algorithm explores every adjacent node $n$ of $s$. A queue $P$ is introduced to keep track of the explored paths; initially, it includes all the paths from $s$ to $n$. Following these paths, the algorithm continues discovering the adjacent nodes to look for destination $d$. For a path $p$ dequeued from $P$, the last node $n$ is extracted, all of its neighbors are checked in sequence to determine the valid next hops towards $d$. Once a neighbor is determined as valid, link to that neighbor will be added into the current path forming a new valid path toward destination. This new explored path is then enqueued into $P$ for the next discovering phase. A node is considered as valid once the path through it does not violate the valley-free routing property [77]; we express such policy-compliant

path (i.e., a path that complies with the valley-free routing policy), using the following regular expression $c2p * p2p?p2c*$ [73] in which $c2p$, $p2p$ and $p2c$ denote the relationship between interconnected nodes (where ? means that you can have one or none p2p link).

It is worth noting that, within $G$ links are labeled according to their inferred relationship. For example, assuming that $n_1$, $n_2$, $n_3$ are the three neighbors of node $s$, in which $s$ is customer ('$c$') of $n_1$, provider ('$p$') of $n_2$ and peer with $n_3$; the links $(s, n_1)$, $(s, n_2)$, and $(s, n_3)$ are labeled as '$c2p$', '$p2c$' and '$p2p$', respectively. With these labels, the preceding regular expression defined for policy-compliant path then could be leveraged to determine the validity of next hop toward the destination. For instance, taking the customer-type neighbors among the neighbors of $s$ (i.e., $n_2$), and looking at their neighbors $x$ in turn, those $(n_2, x)$ links are not validated if they are either $c2p$ or $p2p$ because a customer is not expected to grant transit towards its other provider(s) to one among its providers, and a customer is not expected to give access to its peer(s) to its provider(s). By checking the labels of links along the explored path, the validity of next hops can be determined. Once a valid path is discovered, it is enqueued into $P$ for the next discovering phase. The same exploration and validation processes are repeated for all the paths in $P$ until reaching destination $d$ or the path length goes over a given threshold $\tau$.

The path validation logic is executed at run-time, i.e. right after discovering a new path toward destination a validation process is triggered, to ensure that non-compliant paths are detected at the early stage, thus avoiding wasting time exploring invalid paths. By reducing the number of paths needed to be explored in the following phases, the search space is continuously optimized. Moreover, a proper choice of $\tau$ not only limits the time and space complexity, but can also avoid selecting long paths which should be avoided in current routing practice.

As a result of the path search algorithm, policy-compliant paths between two endpoints may share common nodes. To get the final set of vertex-disjoint paths, we run a simple off-line filtering linear algorithm to capture the shortest disjoint paths. Since the original list of valid paths turned out to be quite small most of the time and already sorted, the complexity of such a filtering operation is negligible.

### 6.3.3 Source-destination pairs

Within the constructed AS-level graph, an end-to-end connection over the Internet could be simulated by simply attaching two end-hosts as virtual nodes into AS nodes of the original graph. Simulating a multipath connection requires at least one of these two virtual nodes to be multi-homed. For instance, a multi-homed device can be emulated by adding a new node, then linking it with at least two AS nodes. The connection from that multi-

**Algorithm 1:** Path Search Algorithm

```
input  : source s, destination d, graph g, threshold τ
output: ValidPathSet
VisitedNodes ⟵ ∅
queue.append([s])
while queue not empty do
    path ⟵ queue.pop()
    v ⟵ path.LastNode()
    if v ∉ VisitedNodes then
        for n ∈ v.NeighborSet do
            if n ∉ VisitedNodes and (label(v,n)='p2c' or label(v,n)='p2p') then
                for x ∈ n.NeighborSet do
                    if label(n,x)='c2p' or label(n,x)='p2p' then
                        g.RemoveEdge(n,x)
                    end
                end
            end
            NewPath ⟵ list(path)
            NewPath.append(n)
            if n = d then
                ValidPathSet.append(NewPath)
            end
            if length(NewPath) = τ + 1 then break
            queue.append(NewPath)
        end
        VisitedNode.add(v)
    end
end
```

homed source node to any other virtual destination node forms a multipath transport-layer communication. Our approach for emulating multipath communication can therefore be simply referred to as a process of such source-destination pair selection. In the following, we define the target set of AS nodes which we consider for attaching the end hosts. A simulation process is then described in details explaining which communication scenarios are covered in our study.

The current Internet ecosystem is composed of more than 60 thousand ASes, out of which the large majority are stub ASes, i.e., ASes that are only origin or destination ASes. About 13% are Tier-3 or small Tier-2 ASes, we arbitrary define in this study as those appearing at most in the third from last position and at least penultimate position in BGP AS paths; we refer to such ASes as 'edge provider' ASes, which can be considered as a representative set of national Internet Service Provider (ISPs), or 'eyeball' ASes (hence excluding Internet carriers and stub ASes).

Rather than taking into account all possible communications, we target the connections among hosts at the edges, i.e., hosts connects to the edge provider ASes, performing connections using multiple sub-flows. Considering connections between hosts in different countries, we precisely address the MITM robustness of Internet connections crossing multiple ASes. To precisely determine which communications to cover in our study, we define a target set of source-destination pairs that addresses, in a reasonable yet arbitrary way, the communications that may be more sensitive to communication privacy. Our choice of source-destination pairs is as follows:

- the source is interconnected to two edge providers in a country.
- the destination is not multi-homed, i.e., it is reachable via a single ISP, the one given by the best BGP path from each source edge provider, and belongs to an AS at another country than the one of the source.

Figure 6.1 illustrates an example of how we simulate multipath communications accordingly the above policy. For each two arbitrary edge provider ASes in a same country, one source is created (i.e., a dual-homed source). For each edge provider in another country, one destination is paired with the source. Such a pair dual-homed source - single-homed destination defines the two endpoints of a multipath communication. Listing all pairs, i.e., combining a given source with every destination, all possible (international) communications of a dual-homed host can be covered.

Besides reducing the number of pairs to a reasonable and treatable number (requiring about one week of computation), it is worth noting that, in such a way, we consider communication in a single direction: from source to destination. That is, under such a

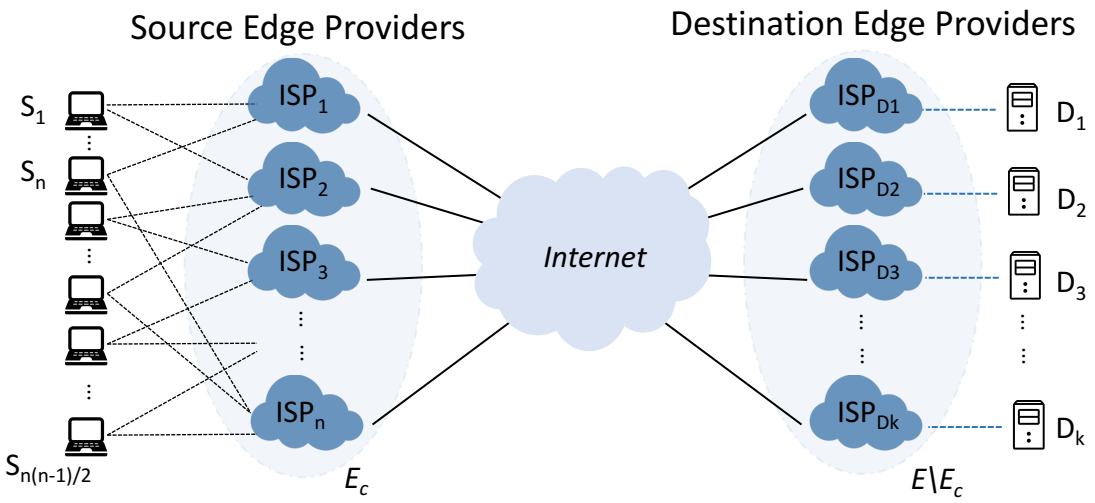



Figure 6.1: Representation of the source-destination pair selection process.

path election strategy, we cover the case when a multi-homed device *uploads* to a single-homed server, as well as the case when a single-homed device *downloads* contents from multi-homed servers.

The scenarios that are not covered in our study include: (i) multi-homed devices *downloading* from single-homed server; (ii) single-homed devices *uploading* contents to multi-homed servers; (iii) a multi-homed device communicating with another multi-homed device. A dual analysis, quite expensive computationally, covering these additional cases may be performed as well in future works.

### 6.3.4 MiTM robustness metric aggregations

The ability to split traffic over different paths allows a multipath protocol to secure its communication against the MiTM attacks. Thus, the chance for an attacker to capture all the traffic sent by a source is reduced in proportion to the number of disjoint paths between source and destination. Path diversity is therefore a proper indicator to evaluate the MiTM robustness of a multipath communication.

Rather than considering the robustness against MiTM attacks of every connection individually, we are more interested in evaluating such robustness at the end-host level, thus measuring the degree of robustness offered by a multipath-capable source device to secure its data sending over the Internet. With regard to the aforementioned approach for source-destination pair selection, we define the *source-specific MiTM robustness metric* as the average number of disjoint paths over all the destination edge providers that are in a different country than the source. Such a metric can be considered as a level of

unlikelihood that a MiTM attack takes place for that source configuration; the higher the value of the robustness metric, the more difficult it is for an attacker to capture traffic from that source. Moreover, aggregating results from all the sources within a given country we can obtain a *source country-specific MiTM robustness metric*. Such a definition allows us to characterize the robustness level offered by different source countries to multipath communications.

As another way to aggregate the MiTM robustness metric computation, we also study a country-level source-destination based aggregation, i.e., leading to a robustness metric for a pair of source and destination countries. Given a source (a pair of edge providers in a country) and a destination country, its MiTM robustness metric is defined as the average number of disjoint paths from the source over all edge providers belonging to the destination country. Furthermore, by grouping together the results from all the sources within a source country, we can define the *country-pair MiTM robustness metric* for the corresponding pair of countries.

Let us more precisely characterize the aforementioned source-destination pair selection process with respect to the two MiTM robustness metric aggregations we study in the following, i.e., the source country-level one and the country-pair one. We segment the set of edge providers, $E$, in country-specific subsets, $E_c$, where $c$ denotes a country in the set of countries $C$, i.e., $E = \bigcup_{c \in C} E_c$. We employ the AS-to-country mapping given by the CIDR Report [78]. Overall, for a given country $\tilde{c}$, the number of source-destination pairs is therefore equal to:

$$\frac{|E_{\tilde{c}}| \times (|E_{\tilde{c}}| - 1)}{2} \times \sum_{c \neq \tilde{c}} |E_c| \tag{6.1}$$

For a given source and destination countries, $s$ and $d$ respectively, the number of source-destination pairs connecting them is equal to:

$$\frac{|E_s| \times (|E_s| - 1)}{2} \times |E_d| \tag{6.2}$$

Doing so, we target a lower bound, pessimistic analysis, since we only take into consideration international communications and we suppose the destination is not multi-homed. The filter we set on the destination enumeration allows us to target communications that may need a higher level of confidentiality due to their international connotation. Moreover, in this way we also avoid a huge bias potentially due to the fact that a large majority of the AS paths available at the national level are not visible in backbone BGP routing tables such as the Routeviews ones (typically because of Internet exchange points, as recently shown in [79]). We believe having a lower bound stand is more appropriate than an upper bound one, while allowing us to scientifically qualify the value of the relative trends.

## 6.4 Results

We report the results obtained for a set of 147 countries, i.e., those countries from the United Nations statistics [80] that appear to have at least two distinct edge providers officially based in the country; this automatically excludes Greenland territories, very small city-state countries, many African countries and Indonesia. The geographical coverage is given in Figure 6.5. In the following sections, we present the statistics for two different MiTM robustness metric aggregations, the country source specific one and the country pair one.

### 6.4.1 Source country aggregation

Let us recall the measurement approach for source country-specific MiTM robustness analysis:

- For each country, we generate all possible dual-homed sources, i.e., all possible pairs of edge providers. In figure 6.2 we report the distribution of the number of such dual-homed sources over the set of observed countries.
- For each such source configuration, we compute the number of disjoint paths to each destination. For each edge provider that is a different country than the source country, one destination is generated. The distribution of destinations over different countries is presented in figure 6.3.
- For a given source, we compute its corresponding robustness metric by taking the average of the number of disjoint paths over all the destinations.
- For each country, a series of MITM robustness metrics is hence generated, one for each source.

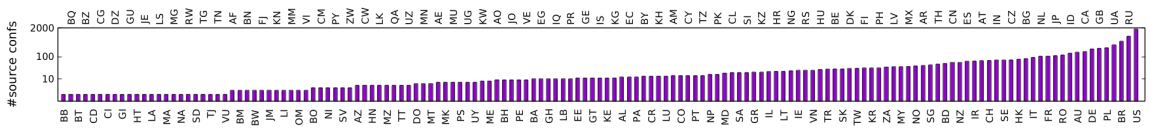


Figure 6.2: Number of source configurations per country

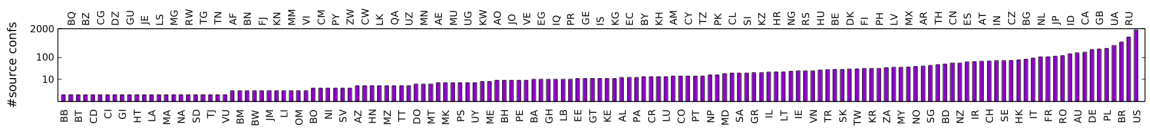
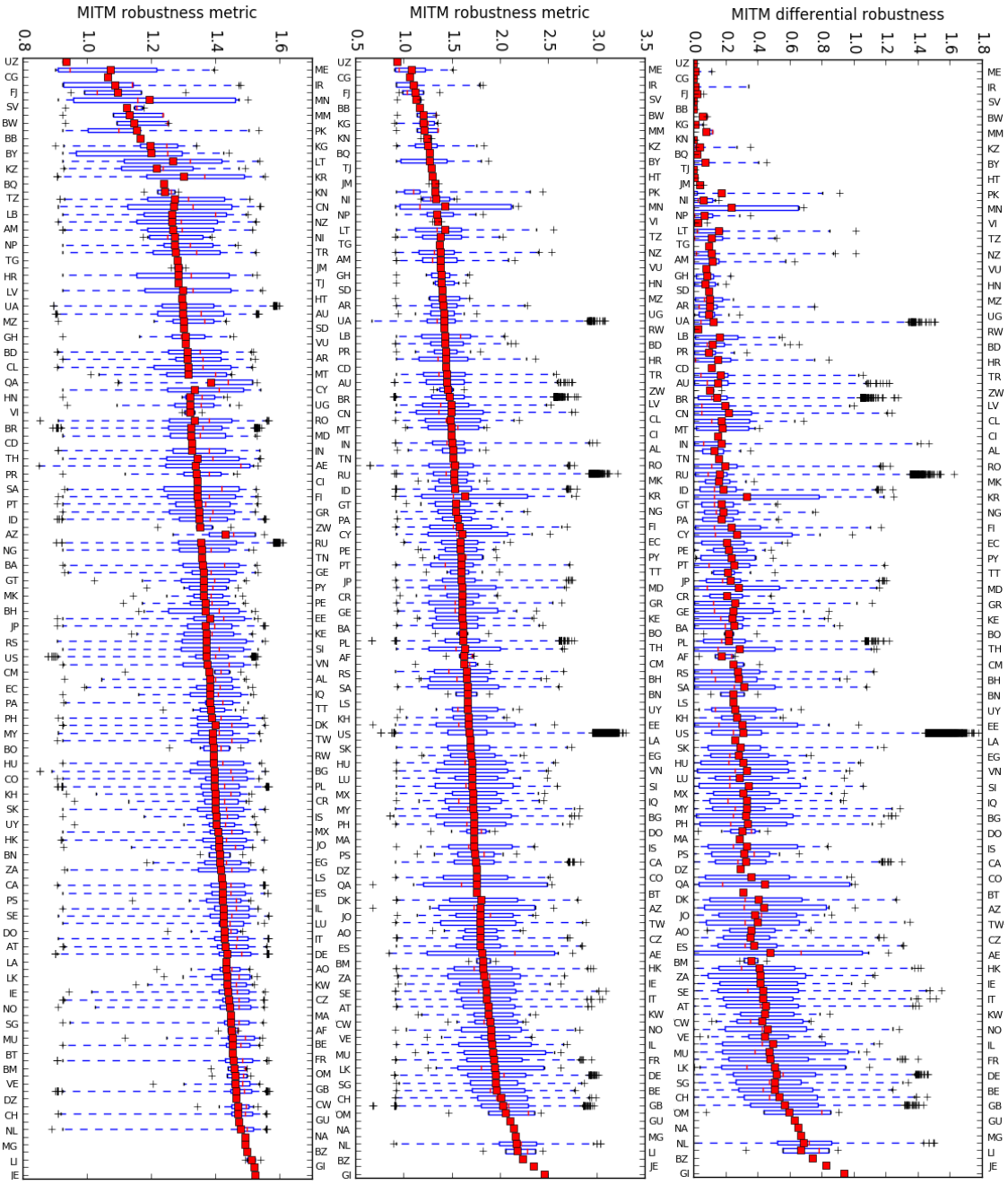


Figure 6.3: Size of the destination set per country

We characterize the resulting series using boxplot distributions (using a 0.1% outlier threshold). Figures 6.2 and 6.3 show that the three countries with the highest number




(a) device view (b) edge provider view (c) differential robustness view

Figure 6.4: MITM robustness distribution for 147 countries.

of source configurations (i.e., those with the highest number of edge provider pairs), e.g., Brazil, US and Russia have also the lowest size of the destination set, which is reasonable given (6.2), while guaranteeing a largely sufficient statistical significance (thousands of entries for each country). We overlay over the boxplots the average of the corresponding series with a red square, order them with increasing averages[2] from left to right. We report the results in Figure 6.4, and with a geographical view in Figure 6.5. We express three different viewpoints:

- *device view* (Figure 6.4a): the MITM robustness is computed with the source node integrated in the AS graph as an 'artificial' node, i.e., the path search algorithm finds the number of AS-disjoint paths from this source node toward the destination. It provides therefore a device view; obviously, in this view the upper bound of the robustness is 2, i.e., the number of edge providers used by the source.

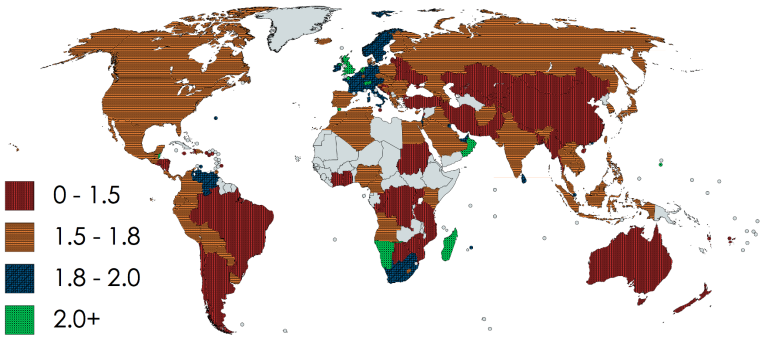
- *edge provider view* (Figure 6.4b): the MITM robustness is computed counting the number of disjoint paths from the first and the second edge provider, then decreased by those paths that share an AS hop. Taking into account such a view, we assume that additional AS paths can be made available to MPTCP subflows acting at the edge providers level, e.g., by forms of flow path steering and load-balancing.

- *differential view* (Figure 6.4c): the differential robustness results, i.e., the edge provider view robustness minus the device view robustness, computed for each source configuration individually. This view more precisely quantifies the gain achievable for MPTCP communications when inter-AS load-balancing is enabled at the edge providers.

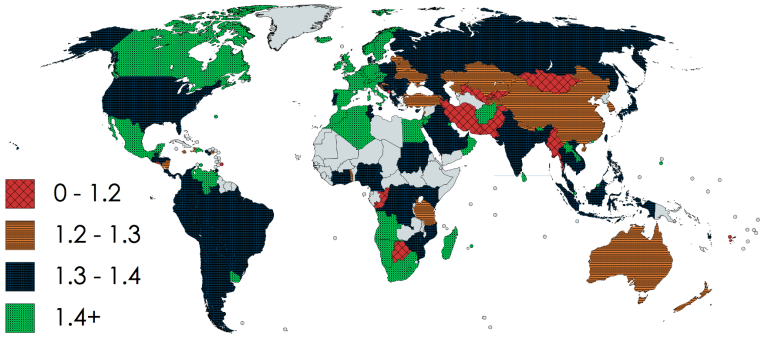
The above viewpoints also reflect different levels of trust on the providers. That is, while the edge provider view assumes MITM attacks do not happen at the source and destination edge providers (i.e., there is a high level of trust on those providers), the device view assumes that attacks can happen at the source edge providers, hence revealing a low level of trust in source direct providers.

As a general assessment, Figure 6.4 shows a distribution to be interpreted. For example, one could consider 1.5 as the rough threshold above which the likelihood of MiTM is to be considered low, and conversely high if lower than 1.5. Only about 5% of the countries show good chances of being robust against MITM from a device viewpoint, while looking at the maximum instead of the average and median values one could speculate that careful choice of the edge providers could make the MiTM likelihood low for a majority of the countries. From an edge provider viewpoint, this ratio grows to roughly 60%, and higher

[2] Average values do include outliers.




(a) device view




(b) edge provider view

Figure 6.5: Countries covered with corresponding MiTM robustness distribution

than 90% looking at the maximum, that is if the edge provider choice can be influenced by confidentiality concerns.

Moreover, the average number of paths connecting a dual-homed node to international destinations has a significant variance depending on the origin country. The average robustness ranges from 1 (and less) to 1.6 from a device viewpoint, and from 1 (and less) to 2.5 from an edge provider viewpoint. It is worth noting that the reason why some minimum, and even average values, are below 1, is the partial view over the Internet topology and the incompleteness of inter-AS relationship inference; in fact, these factors make some destinations unreachable (counted as 0 path), but we left the 0 values in the series to also give an index of the level of topology incompleteness for different countries. In any case, the boxplot median is a metric robust against such outliers to look at.

In addition, observing the distributions in Figure 6.4, we can also remark that:

- Within a country, a high inter-quantile range indicates that the path diversity strongly depends on how the two upstream edge providers are selected for the source.
- The gap between the min and max robustness is another interesting fitness metric to observe. Some countries maintain a small gap (below 1) while others have a very big gap (up to 2). In other words, the deployment of multipath transport-layer

communications for securing international communications in some countries can statistically yield a much better result than in other countries, where this gap is smaller. Particularly interesting is the case of Angola (AO), Venezuela (VE) and Namibia (NA), with small robustness gaps, which may be correlated to the presence of inter-continental cables landing in or close to the country [81].

- The median is mostly higher than the average in the device view, and lower than the average in the edge provider view. This is essentially due to outliers, counted in the average and not in the median.

- From the edge provider viewpoint, the maximum value is higher than 2 in the most of the countries, suggesting that with a proper choice of trusted source providers, one can adopt multipath communications to statistically expect high confidentiality for its communications. Particularly alerting are the cases of Uzbekistan (UZ), Nepal (NP) and Lebanon (LB), with quite low maximum values.

- From the device viewpoint, in most of the cases the maximum robustness is not higher than 1.6, both averages and medians are quite far from the desirable target of 2. Hence, without the support of inter-AS load-balancing at source providers, path diversity from a dual-homed node is reduced significantly, indicating a non negligible probability of paths joining on the way to the destination.

- Considering the differential robustness, we can remark that among the countries that have the lowest device view MITM robustness, those that could most benefit from inter-AS load-balancing practices are Mongolia (MN), Pakistan (PK) and Korea (KR). However, the majority of those countries with low robustness do not improve much the situation going from the device view to the edge provider view.

Looking at macro geographical regions, many European countries seem to grant better security than countries in other regions. In order to look at continental characteristics, the plots in Figure 6.6 show the boxplot results (with 1% outliers) aggregated on a macro-region basis (a and c, sub-continental level) and on a relative position basis (b and d, in terms of seacoast and inland borders). We can remark that:

- Western Europe appears to be the best off, followed by Northern Europe and Northern America. In almost 50% of Western Europe countries there can be 2 disjoint paths from the source edge providers to Internet destinations.

- Central Asia shows the worst robustness, followed by Australia and New Zealand; the reasons are likely network centralization practices and geographical isolation. It

is interesting to notice the relevant gap between Central and South-Eastern/Western Asia.

- Within Europe, Western countries do offer a better diversity over Northern countries, and especially over Eastern and Southern countries. with a small range of variation and a high median value show the best result.
- A high variance is recorded at Southern Asia, Northern Europe and Sub-Saharan Africa, which indicates high differences among the countries within these areas.
- We could not find a strong correlation between the relative continental position, and the robustness metric, yet a positive correlation exists, with countries at the boundaries of oceans, with inter-continental cable landing and that are sea-oriented (most of the border on the coast) that offer higher robustness than fully internal and continental-oriented ones.

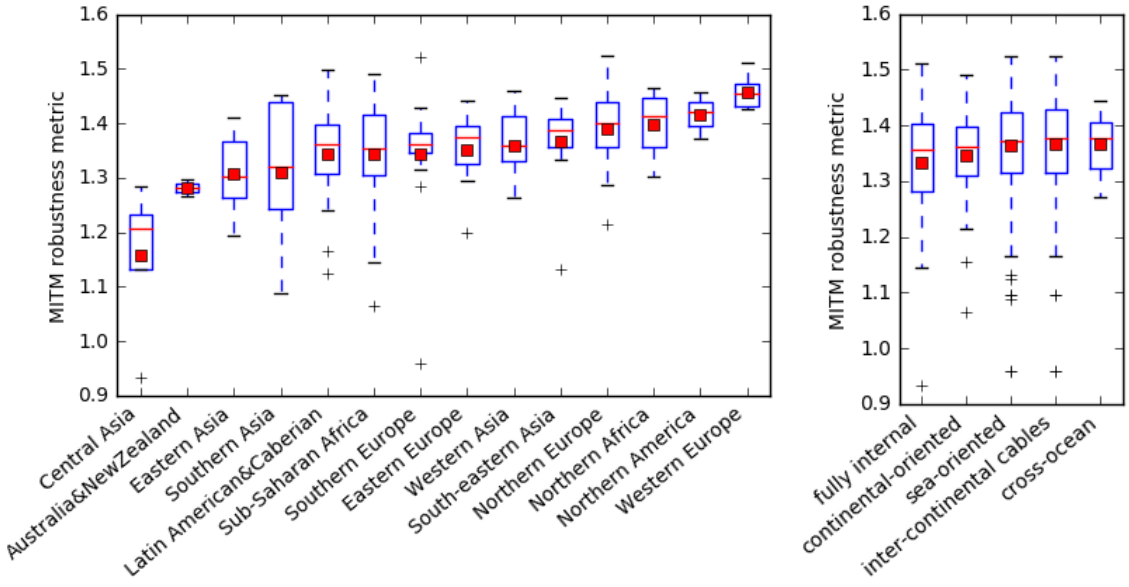
### 6.4.2 Source-destination country pair aggregation

As we may notice, the MiTM robustness level of a multipath communication could be affected not only by the country where the communication starts but also by the choice of upstream providers at that country. Besides that, within a source country, the robustness level for different destination countries can significantly vary. To evaluate this latter aspect further, we perform a source-destination country pair aggregation.

Over the set of 147 countries, we evaluate the robustness metric for 1547 directional country-to-country communication pairs in which the MiTM robustness metric for one pair is computed as follows:

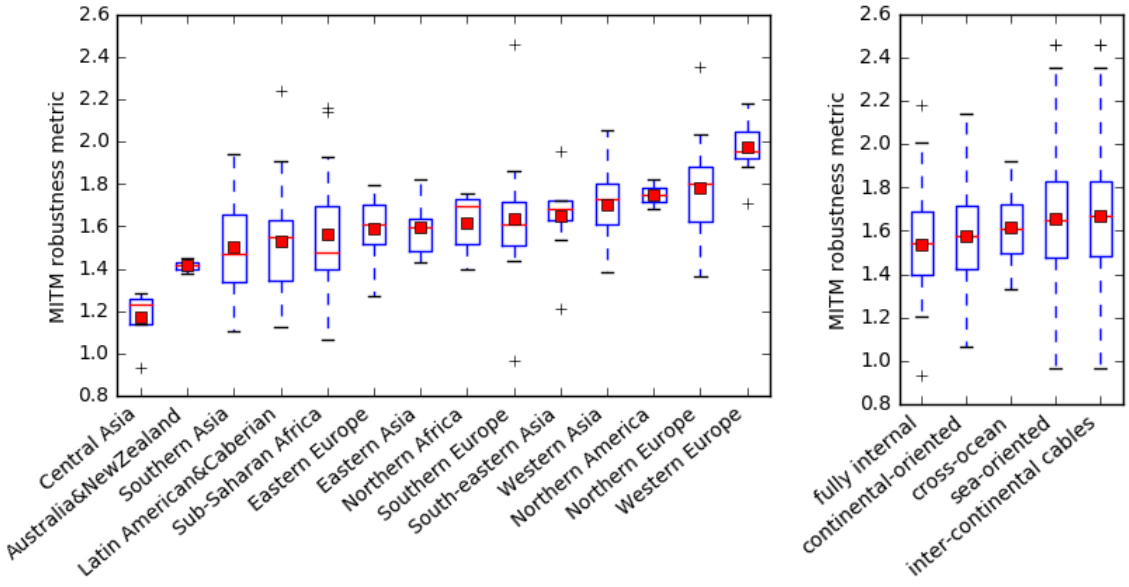
- For a given source country, we generate all possible dual-homed sources, i.e., all possible pairs of edge providers.
- For each such source configuration, we compute the number of disjoint paths to each edge provider located in the destination country.
- For a given source, we take the average of the number of disjoint paths over all the destinations to get its source-destination based MiTM robustness metric.
- For a given source-destination country pair, a series of MiTM robustness metrics, one for each source, is therefore created.

In Figure 6.7, we report the CDF of the average MiTM robustness, for all the 1547 pairs. The high range of variation (between 0.4 and 6) shows us the big robustness gap




(a) device view: macro-regions grouping

(b) position grouping




(c) edge provider view: macro-regions grouping

(d) position grouping

Figure 6.6: MITM robustness metric with continental subregion grouping.

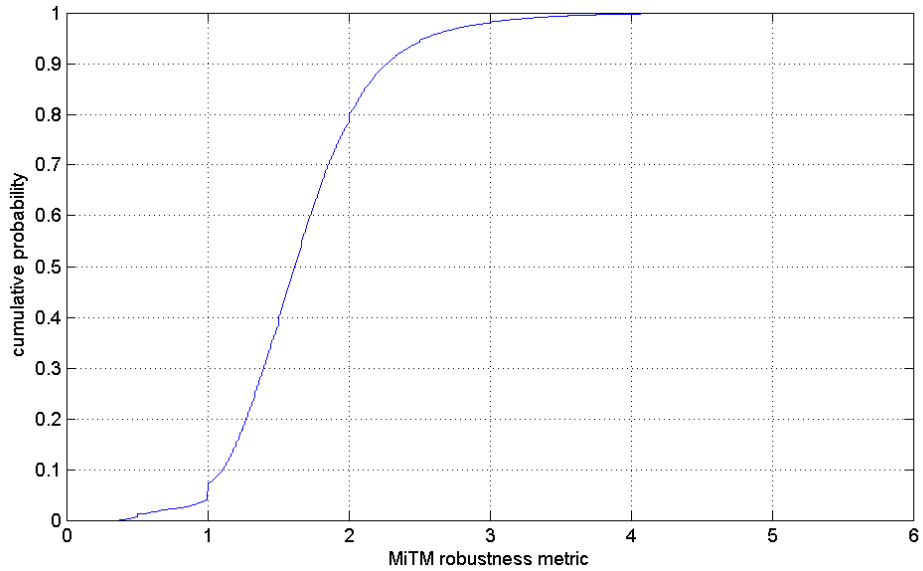


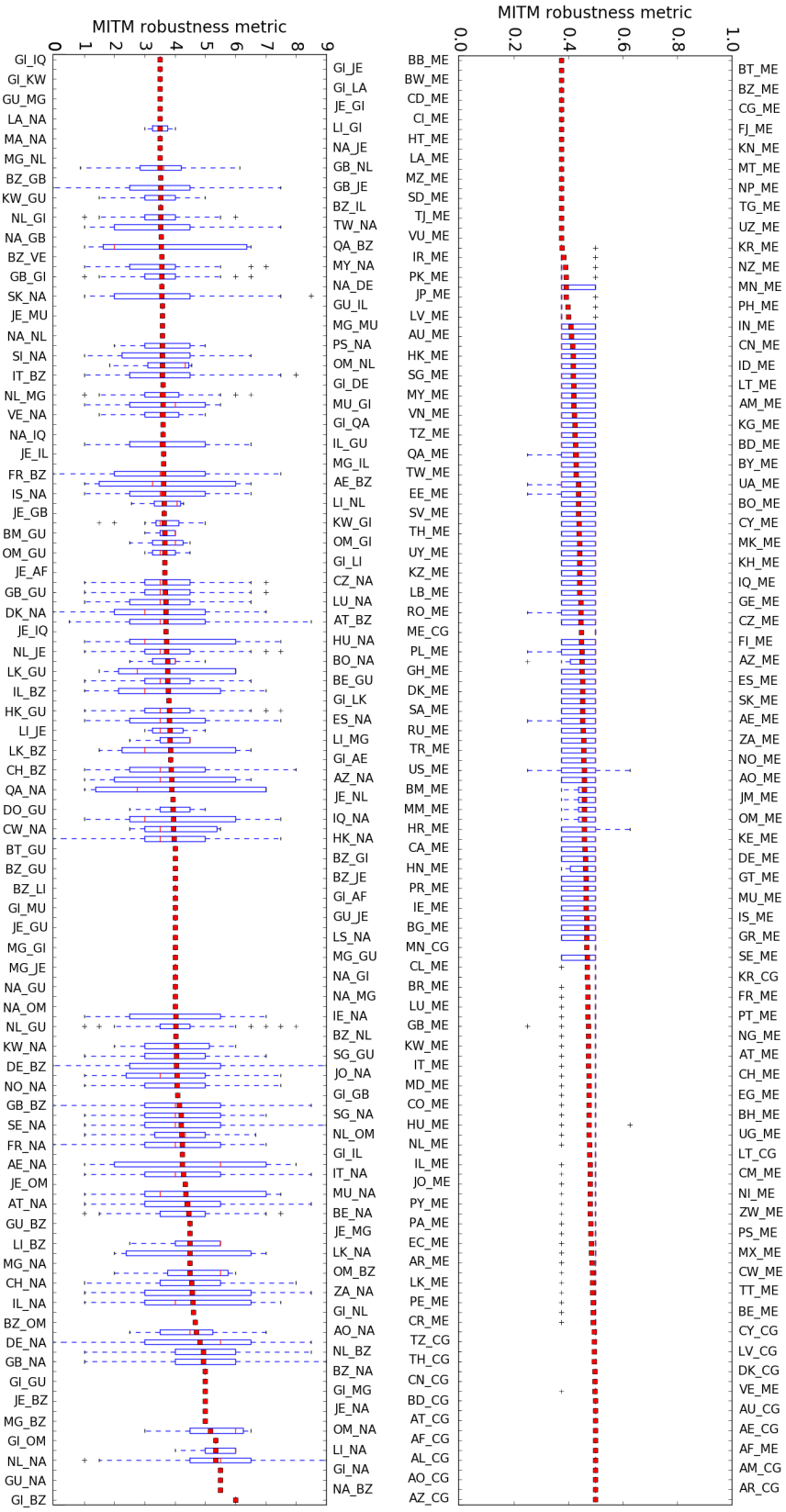

Figure 6.7: CDF of average MiTM robustness for 1547 pairs of source-destination country

between pairs. Only 20% of the country pairs show an average of two or higher. For the remaining pairs, approximately 73% of them have the average range from 1 to 2. The remaining 7% are country pairs with very low robustness, below one; besides the specific context related to a country pair, a factor behind such bad performance can be the already discussed topology view incompleteness.

To better understand the impact caused by different destinations, we further characterize the top 147 and bottom 147 pair in the CDF distribution, i.e., roughly the top 10% and the bottom 10% cases. The results are presented in Figure 6.8, where the country pairs in each group are ordered from left to right with an increasing average (the average do include the outliers). We report the MiTM robustness distribution of each pair using the boxplot (with 0.1% outliers) overlaid with a red square representing the average.

Figure 6.8a reports the MiTM robustness metric distribution for the top 147 country pairs. The high inter-quartile range (IQR) with a pair highlights the strong impact caused by edge providers choice at the source to the robustness metric. Besides that, there are also some source countries, such as Morocco (MA), Madagascar (MG), Gibraltar (GI), Guam (GU), Jersey (JE), Namibia (NA), Liechtenstein (LI) and Belize (BZ), that suffer from the presence of only one edge provider pair; these countries result in pairs with a collapsed robustness point in the box. In addition, within these top 147 pairs, there are some destinations, like Namibia (NA), Guam (GU) and Belize (BZ), that appear to show high sensibility to the destination choice on the MiTM robustness.

In Figure 6.8b, we report the results for the bottom 147 country pairs. The majority




(a) 147 most robust country pairs (b) 147 least robust country pairs

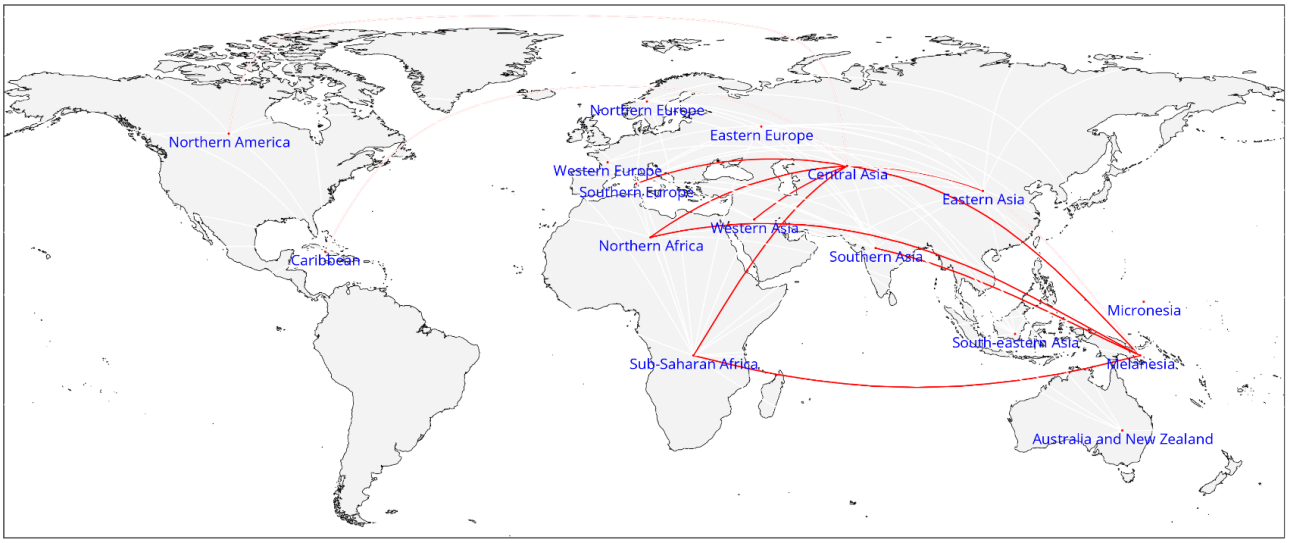
Figure 6.8: MITM robustness distribution for the top and bottom 147 pairs of country (with respect to their average MITM robustness)

of them have Montenegro (ME) as the destination. The second popular destination is Republic of Congo (CG). That highlights again the impact of destination choice on the MiTM robustness level. Unlike the top 10% case, we see a small inter quartile range (IQR) for most of the pairs, showing that even a careful choice on the edge providers at the source country cannot improve much the level of robustness for such connections. In other words, regardless of the origin country as well as the choice of source edge providers, the possibility of employing MPTCP to secure the communications destined to, e.g., Montenegro and Republic of Congo is extremely low.

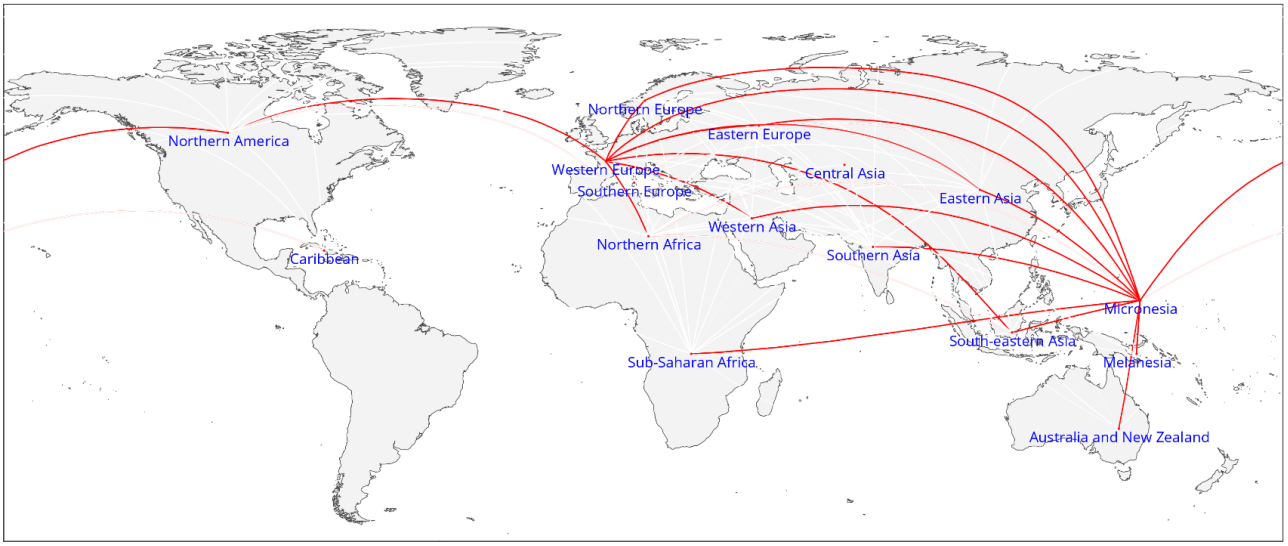
Considering 1 and 2 as the thresholds for very low (zero) and high (sufficient) robustness, respectively, a source-destination pair can be classified as: (1) highly robust against MiTM if it has the average robustness level of at least 2, and (2) weak against the MiTM once maintaining the average of 1 or lower. We visualize the country-to-country communications in these two classes by mapping them into a geographical map in Figure 6.9. To avoid too many lines, we first group countries with respect to their subregion, then converting these country-to-country connections into the corresponding subregion-to-subregion connections. Finally, the subregional connections are expressed using lines with different opacity reflecting the portion of country-to-country communications between subregions having the MiTM robustness level less than or equal to 1 as in Figure 6.9a, and equal to or higher than 2 as in Figure 6.9b.

In Figure 6.9a, we only show the connections between subregions when there are more than 30% of the country-to-country communications with a robustness metric of at most one. For subregion pairs with less than 30% of their country-to-country communications having a robustness metric lower than one, the connection lines are hidden. In other words, the lines point out the subregions where the deployment of MPTCP cannot offer any protection against large-scale MiTM attacks. As presented in the map, the area of Central Asia and Melanesia are the two subregions having the worst performance, most of their MPTCP communications with other subregions are classified as zero-robust. Thus, most of the subregions could not be benefit from the deployment of MPTCP to secure their communications with Central Asia.

In the sub-regional view of the high robustness group presented in Figure 6.9b, we show the connection lines between sub-regions with more than 50% of the country-to-country communications having robustness level of 2 or higher. In such a view, Micronesia and then Western Europe are the two areas that outperform the others in term of MiTM robustness. As depicted in the plot, except for a few low connected regions, like Central Asia, Caribbean and Northern Africa, etc., most of the multipath communications from and to Micronesia can profit from a high level of robustness. It is worth noting that in the




(a) regions with more than 30% of country-to-country communications having at most one path




(b) regions with more than 50% of country-to-country communications having at least two paths

Figure 6.9: Regional view of the source-destination based MiTM robustness

region of Micronesia, Guam is the only country covered by our study. The high robustness result captured for communications from and to this region is therefore directly related to the highly connected network infrastructure of Guam being a crucial node in the Internet cable network [82].

## 6.5 Application scopes

We focused our study on MPTCP-based communications. More precisely, it covers the following cases:

- *MPTCP capable endpoints*: both source and destination, client and server (or vice versa), are MPTCP capable, and the MPTCP communication is not filtered by middle-boxes. As argued in Section 6.3.3, the multi-homed endpoint can be either the server or the client.
- *MPTCP proxied endpoints*: at least one endpoint is not MPTCP capable, but the TCP communications are handled by MPTCP proxies, converting TCP packets into MPTCP packets and vice versa, as explained in [44, 83], possibly routed via Internet disjoint paths as proposed in [84, 85]. The multipath conversion proxies can sit at endpoint premises (customer premises equipment for the client, hypervisor or middle-box at the server) or at the edge provider level borders.

Besides MPTCP-based communications, other protocols offering Internet-scale multi-path, connection flow-level load-balancing could also be covered by our study. The following protocols are either not deployed, or they have only undergone a limited deployment at the Internet scale so far; they are:

- *SCTP*: the Stream-Control-Protocol (SCTP) [86] is another multipath transport protocol absolving the same function as MPTCP, but less deployed than MPTCP due to the limited retrocompatibility.
- *LISP*: the Locator/Identifier Separation Protocol (LISP) [5] is able to perform inter-AS inbound load-balancing by means of encapsulation, routing locator mapping, and appropriate traffic engineering (TE) policy configuration. LISP primary scope is the edge provider one, hence results with the edge provider view are readily applicable. Furthermore, deployment of LISP as an intra-AS TE tool can also allow us to perform inter-AS multipath on the outbound direction as proposed in [87].
- *MultiPath BGP*: in BGP, the routing decision process only allows us to take one route per network prefix. The selected path can be inefficient in terms of global routing.

Recently, forms of *Multipath BGP* were discussed in standardization fora, but finally not standardized; however, some recommendations have been published [88], and implemented by some vendors (see, e.g., [89] and [90]). Such multipath mode can be adopted at the edge provider scope to enable load-balancing at the egress direction. Despite the study [91] on core routing tables reports that in 2010 multipath BGP was practically not used, speculations report that it is used by major cloud providers.

The above protocols are a selection of those protocol communication contexts where load-balancing can affect the AS-path selection. There are also other load-balancing protocols which can potentially influence the egress AS selection as well, as for instance in data-center environments. In the case of MPTCP communications, these protocols, operated at the edge provider view, are able to perform inter-AS load-balancing in such a way that the path diversity exposed in our edge provider view can be made available to MPTCP devices, hence giving them the full potential of MPTCP in terms of communication confidentiality and robustness against MITM attacks.

Finally, additional multipath transport-layer protocols are making surface, as for example the already mentioned multipath extension to the QUIC protocol [48].

## 6.6 Conclusions

We explored in this chapter how Internet path diversity could be exploited by means of multi-path transport-layer protocols such as MPTCP, or even network-layer protocol able to operate at the application flow level, when looking at increased security against man-in-the-middle attacks. We focused on such attacks acting at the autonomous system level, and at the robustness of multipath communications in what appear as a reasonable configuration where at least one endpoint is multi-homed with two edge providers.

We reported extensive, specific and aggregated results for most of the world countries and regions, looking at macro trends that could inspire further research in the area. Results show that, statistically speaking, multipath protocols do not help in guaranteeing robustness against MiTM attacks hence high confidentiality, unless (i) the choice of the edge provider is carefully taken, or (ii) one can rely on inter-AS load-balancing features offered implicitly or explicitly by edge providers. Some continental regions are strongly more robust than others, and there seems to be a positive correlation with inter-continental cable landing proximity. Moreover, the results show that there are countries surprisingly less well connected than one could think of, such as Northern America countries, and countries that are more obviously less robust against such attacks due to network centralization practices.

# Chapter 7

# Scheduling challenges in multipath transport

Two different MPTCP load balancing strategic behaviors were presented and discussed in the previous chapters. In Chapter 2.6, we overviewed an application of multipath equilibrium routing to MPTCP load-balancing. In Chapter 6, we described a strategic behavior to increase confidentiality making explicit strategic use of path diversity in multipath transport. Both require an MPTCP scheduler that differs from the ones existing at the state of the art, which we present in this chapter.

## 7.1 Introduction

In order to leverage the Internet path diversity and hence increase the robustness level for communications against man-in-the-middle attacks – as presented previously in Chapter 6 – a multipath transport layer protocol such as MPTCP needs a scheduling mechanism that balances traffic load over the available paths with the certainty that minimum load balancing ratios are guaranteed on each path. Moreover, being able to implement a load-balancing multipath equibrium distribution as proposed in [3] and resumed in Chapter 2.6 also requires explicit load-balancing over MPTCP subflows. The requirement arising from these two use-cases is therefore a scheduler able to implement and guarantee a load balancing distribution for ensuring connection confidentiality or strategic routing equilibrium.

The default round-robin and least RTT schedulers [92] available from the early Linux kernel implementation of MPTCP could hardly satisfy the above requirement. The least-RTT one risks to send the large majority, if not all, the traffic on a single subflow. The round-robin one could guarantee that multiple subflows are used concurrently, but it can strongly impact the performance [92], and more importantly it offers no control on how

the traffic is distributed among paths, i.e., it cannot provide any guarantee on the load-balancing distribution.

In recent years, in conjunction with the growth of multihoming practices for end devices, the integration of MPTCP in some operating systems, many studies on MPTCP scheduling lead to actual implementations for testing. Most of them address the common challenge of Head -of-Line (HoL) blocking, which arises when MPTCP is employed in an heterogeneous paths environment. Besides that, there are also efforts to enhance connection reliability, to optimize the aggregated throughput [93, 94], to integrate with application data for improving scheduling decision [47], and also to simplify the development of scheduler [46]. However, as of our knowledge and research, there is not an explicit way designed to explicitly control the load balancing over subflows. In other words, we cannot guarantee the amount of traffic load on each subflow.

The requirements from end hosts as well as applications for multipath transport could be very different, e.g., one may want to leverage all the available paths to enhance confidentiality or to improve performance in case of homogeneous paths environment; or one may expect to replicate traffic on multiple paths for reliability reason; or one may opt for monetary cost or power saving, etc. Obviously, there is no one-fit-all scheduler for such a diverse set of requirements. Developing its own scheduler requires application owner to touch the kernel space which could be a quite complex and time consuming task. Another solution is to inject user-defined or application policies into the MPTCP scheduler. However it is also a challenge since subflow signaling in MPTCP is intentionally designed to be transparent from the application layer.

With an aim to provide a simpler and more direct approach for end host as well as application to actively manage the distribution of traffic load over multiple subflows, we design a weighted load-balancing (WLB) scheduler for MPTCP. The design of a WLB allows us to partially fill the gap between the application requirements and the scheduling algorithm. Before going into detail of the proposed solution, we explain in more details the scheduling algorithms proposed in the recent past.

## 7.2 MPTCP schedulers at the state of the art

The research efforts to develop and improve the scheduling of MPTCP have been initiated since the early days of MPTCP, when its default scheduler, the Lowest-Delay-First (LowRTT) one, showed limitations under heterogeneous paths settings. Thus, after path heterogeneity is identified as one of the major reasons for Head of Line blocking (HoL) – causing performance degradation – various scheduling solutions have been introduced to address the problem. Some of them have been implemented for evaluation in practical sce-

narios, while for others only simulation results are published. We provide in the following a selected yet comprehensive review of MPTCP schedulers.

In the early work [95], the default LowRTT scheduler is enhanced with opportunistic retransmission and penalization mechanisms. With significant performance improvements reported, these two mechanisms were then integrated in LowRTT and enabled by default in the current Linux MPTCP implementation.

In order to overcome receive buffer blocking caused by out-of-order delivery, authors in [96] propose a MPTCP scheduler (named OTIAS) that schedules segments for in-order arrival at the receiver. The delivery delay for each segment over subflows is estimated based on the one-way delay of each subflow. The segment is then scheduled to the lowest-delay subflow even if that subflow has no available congestion window. In that case, the segment waits in the send buffer. In other words, for the purpose of arriving in-order at the receive buffer, the data could be transmitted out-of-order.

With an aim to mitigate the HoL blocking effect, i.e., reducing the blocking time at sender, authors in [97] propose the Delay-Aware packet scheduling (DAPS) scheduler. For each data segment, DAPS estimates its delay over every available subflows, based on the RTT measured on each subflow. Relying on that, a scheduling decision is made to ensure that segments are delivered in-order at the receiver buffer. In other words, the proposed scheduler expects that by carefully deciding the number of segments allocated to each subflow, all subflows are able to converge on the same transmission time. DAPS is used in the latter work [98] and compared with other scheduling algorithms.

In 2016, also to minimize HoL blocking, authors in[98] introduce a scheduler designed to prevent the fast subflows from being blocked (named BLEST). Instead of scheduling segments to a slow subflow when the congestion window of the faster ones is not available, BLEST relies on a blocking estimation rate to make its decision. For a subflow, its blocking rate represents the chance of being blocked once a segment is sent on that subflow, and is computed relying on the send window.

In an effort to improve to performance of video streaming application over MPTCP connection, authors in [47] propose a cross-layer scheduler. The main idea is to leverage the data from the streaming application to support the scheduling decision, i.e., prioritize the data segment which is more important than the other. An improvement is reported; however, the proposed solution requires a cross-layer communication between the application and the scheduler which could be quite complicated to have in practice.

Arguing that HoL blocking may not the main reason for performance degradation in some Internet applications consisting of multiple upload/download for relative short duration, i.e., web browsing, video streaming, etc., authors in [94] show that the under

utilization of fast paths can be the major cause of performance degradation. They introduce the Earliest Completion First (ECF) scheduler to decline the opportunity for sending traffic on slow paths, and therefore increase the utilization of fast paths. The decision logic of waiting for fast path or using a slow path plays the key role in the design of ECF and it is built upon subflow RTT estimation, the corresponding bandwidth and the amount of data available to send.

Rather than focus on a specific scheduling problem, authors in [46] define the Programmable Multipath TCP (ProgMP) scheduler. It uses a high level programming model that allows applications to define their own MPTCP scheduler. Instead of touching the kernel, the scheduler can be programmed from the user space. The proposed model aims to simplify and then accelerate the development process of an MPTCP scheduler, thus enabling more scheduling algorithms to be implemented and evaluated. More importantly, it opens the way to a simpler approach to employ application layer data for supporting scheduling decision. Many applications could take advantage from the proposed model to tailor the scheduler to fit with their requirements.

Recently (2017) authors in [93] propose an optimal load balancing (OLB) scheduler that not only prevent HoL blocking in the heterogeneous wireless environment, but also strives for an optimal aggregated throughput solution. Proving by an analysis based on multipath fluid model that load balancing between subflows is the key to achieve throughput optimality, the proposed scheduler focuses on developing a load balancing algorithm that computes the optimal subflow weights to achieve maximum throughput. These subflow weights are updated iteratively, i.e., once an acknowledgement is received at subflow level. Finally, it combines it with a weighted round robin scheduling algorithm to ensure that the computed subflow weights are always respected.

We summarize in table 7.1 the above described schedulers, in a chronological order, with a short description, the problem they addressed and the status of the implementation. Most of these proposed approaches are designed to address specific MPTCP scheduling problems such as HoL blocking or performance degradation, etc. The traffic load on subflows is therefore implicitly controlled by the scheduling logic which is mostly based on subflow characteristics, retrived in-band. Only ProgMP may allow some level of explicit control over the traffic load on subflows. However, the current ProgMP design does not support schedulers developed at the application layer to collect the identification of subflow, i.e., a four-tuple $(src\_ip, dest\_ip, src\_port, dest\_port)$[1], which is important for

[1](source IP address, destination IP address, source port and destination port - the subflow identification is available at transport layer after the connection between end hosts is established.

| Algorithm | Year | Problems to address | Description | Implemented |
|---|---|---|---|---|
| LowRTT +RP | 2012 | HoL blocking | The Lowest-Delay-First scheduler enhanced with opportunistic retransmission and penalization mechanisms | Yes |
| OTIAS | 2013 | HoL blocking | Estimate the delivery delay of segment when sending it on each subflow, then select the subflow with lowest delivery delay | Yes |
| DAPS | 2014 | HoL blocking | Based on delay estimation, segments are scheduled on subflows in such a way that all the subflows converge on the same transmission time | Yes |
| BLEST | 2016 | HoL blocking | Relying on the send window to estimate the blocking rate of slow subflow. If the blocking rate is high, then decline the opportunities to send on slow subflow and wait for the fast subflow | Yes |
| Cross-layer | 2016 | Performance degradation | Scheduling data segments based on the priority defined by the video streaming application | No |
| ECF | 2017 | Performance degradation | Relying on the data queue at send buffer when deciding to send on slow subflow or to wait for the faster one, prioritize flow with earliest completion time | Yes(not open-source) |
| ProgMP | 2017 | Scheduler implementation | High level programming model that allows applications to define their own MPTCP scheduler | Yes |
| OLB | 2017 | HoL blocking, throughput optimization | Relying on subflow weight to make scheduling decision. Weight is updated iretatively to reflect the subflows status | Yes(not open-source) |

Table 7.1: MPTCP scheduling algorithms

assigning a load to subflows.

## 7.3 A weighted load-balancing scheduler

In order to cope with the limitation of the ProgMP, we designed an MPTCP scheduler that makes its scheduling decision based on the load balancing strategy defined by an out-of-band application, i.e., a load balancing distribution that may be computed based on additional metrics than those that can be retrieved via MPTCP signaling (e.g., RTT, loss, sequence numbers, etc). In other words, we want to be able with MPTCP to set the load balancing decision in an arbitrary way, by an arbitrary application, while the scheduling logic is implemented by the scheduler. The application decides the distribution of traffic over subflows according to its own logic, then configures the computed load balancing ratio into the scheduler via a configuration file or an API. The required scheduling algorithm then ensures that the configured load balancing ratio is strictly followed.

### 7.3.1 Design and implementation

There are two main building blocks in the design of our weighted load balancing (WLB) scheduler: (1) the configuration parser and (2) the scheduling logic. The configuration parser is developed with the aim to translate the configurations made by applications at the user space to the scheduler at kernel space. The second component, the scheduling logic, holds the responsibility to distributing segments to subflows while respecting to the distribution ratio defined by the application. In Figure 7.1, we present the general structure of our WLB scheduler with as reference system architecture the one of the current MPTCP implementation in Linux. At the kernel space, both the configuration parser and the scheduling logic are put in one WLB scheduler box. At the user-space, there is a configuration file where application can define and update the weight for each subflow[2]

The scheduling logic of our WLB scheduler is inspired from the idea of a weighted round-robin scheduler. In the proposed design, each subflow maintains a 'weight' attribute and a 'quota' attribute. The subflow weight defines the percentage of total traffic that the application assigns to a subflow, while the role of the quota is to keep track of the number of segments that were already allocated to that subflow in a turn.

A subflow is then classified as 'being used' if the subflow quota is smaller than the weight. A subflow with zero quota means that it is 'totally unused', and when the quota is the same as the weight (with a non-zero weight configured) that subflow is considered as 'fully used'. Note that a fully-used state subflow is no longer marked as being in the

[2]instead of a configuration file, one could design an ad-hoc API, which is left for future work.

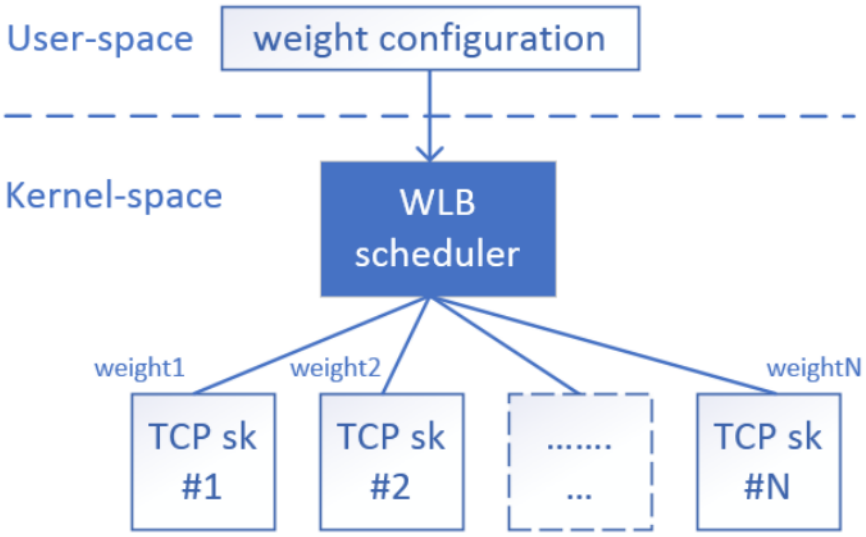



Figure 7.1: Weighted load balancing scheduler design

being-used state. The scheduling algorithm is built upon three occupation states of a subflow. A list of available subflows along with their occupation state is maintained by the scheduler. For each segment received from the application, the scheduler allocates it to a being-used subflow. Once there is no more being-used subflow in the list, a totally-unused one is then selected. Receiving a segment from the scheduler results in an update of the subfow quota (the quota is increased one by one) and its occupation state. When all the subflows are fully-used, their quota is reset to 0 and subflows become totally-unused. For a better illustration, we depict in Figure 7.2 the scheduling logic of WLB scheduler.

We implement our WLB scheduler in the Linux Kernel employing MPTCP code revision v0.91 from [99]. Relying on the modular design architecture of the current implementation, we develop WLB scheduler as a kernel module, and make a minor modification to the source code of MTPCP to integrate the new scheduler. More precisely, in */net/mptcp* we define a new kernel module named *mptcp_wlb.c* which includes the source code for the weighted load balancing scheduler.

The modular design of MPTCP allows developers to implement their own scheduler as a separated module. However, to ensure the compatibility with other components of MTPCP, a common design for the scheduling module is standardized, requiring the implementations of three major functions: $get\_subflow$, $next\_segment$ and $init$. Following the standard design, our scheduling logic is implemented in the $next\_segment$ function, which takes the responsibility for determining which subflow to send a data segment to. The two new subflow attributes to support the scheduling decision in WLB, the quota and weight, are defined as private subflow attributes, i.e., these attributes are only available in that module. The logic for the configuration parser is implemented in the function named

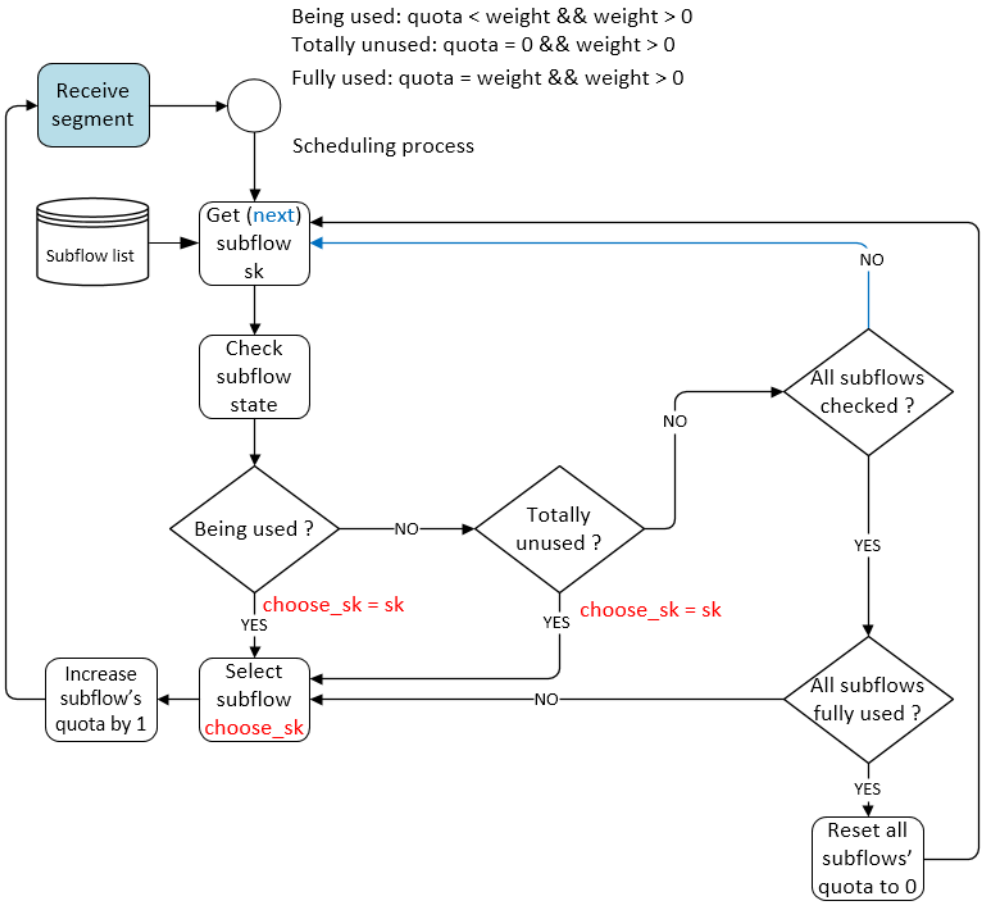



Figure 7.2: Weighted load-balancing scheduling algorithm ('sk' stands for socket; each subflow has its socket). Note that a being-used state can pass to a fully-used state, and one subflow is in only one state at a given time.

*conf_parser* that is called by the *next_segment* function. Finally, to integrate the new scheduler to MPTCP, we define it as instance of the *structmptcp_sched_ops*. During our development process, we follow the same coding standard and naming convention as the other scheduler implementations.

To allow weight configuration at the user space, module parameters for the WLB scheduler module are introduced. After computing its load balancing strategy, the application updates these parameters to configure the corresponding weight for each subflow. At the scheduler, the configuration parser takes responsibility for reading these parameters frequently. Once there is configuration change, the parser updates corresponding subflows weight attribute with the new values.

### 7.3.2 Problems and challenges

By strictly respecting the load balancing strategy defined by an application when distributing the traffic load to subflows, the proposed scheduling algorithm could encounter the problem of performance degradation caused by HoL blocking, or fast path under utilization, if the application does not take into account these problems when determining the subflow weights.

A further direction is to improve the scheduling logic to cope with the performance degradation problem while still respecting the configured weight. Besides the configured subflow weight, other subflow characteristics such as the delivery delay, the blocking estimation, the congestion window size, etc., should be taken into account by the scheduler. In the design of a target scheduling algorithm, traffic patterns should also be considered. Thus, depending on the traffic patterns different scheduling strategies could be applied. Moreover, once the total amount of transmitting data could be predicted or given by the application through configuration file, the decision of scheduler could be improved.

## 7.4 Conclusions

The applications employing MPTCP to leverage the multipath routing architecture may have different requirements for distributing traffic over multiple paths. The modular design of MPTCP implementation allows the customization of scheduler. However, among the schedulers at the state of the art proposed in the recent years, there is not one giving an explicit way for applications to directly control the distribution of traffic over its multiple subflows. With an aim to filling that gap, we introduce WLB, a weighted load balancing scheduler for MPTCP. Our design allows applications (such as those behind the behaviors described in Chapters 2.6 and 6 – to define their own load balancing distribution, while the role of the scheduling logic is to guarantee that this load balancing strategy is always respected. With a simple weighted round-robin design, WLB offers application the control of traffic distribution over subflows.

An open challenge is to design an enhanced version of the scheduler that, by means of an API allows applications to configure the desired load-balancing distribution (instead of using a configuration file) as well as to retrieve parameters from MPTCP in-band signaling so as to better integrate congestion and buffer state information in the load-balancing computation logic.

# Chapter 8

# Conclusions

Experimentation and evaluation is an essential step towards a better understanding not only on the proposed solution for a problem but also on the problem itself. Moving an idea from theory to practice allowed us to shape and reshape novel networking solutions with respect to various practical scenarios and constraints. From an insightful view supported by evaluation results, we could present the advantages as well as drawbacks of new strategic management of path diversity in networks, and we could highlight the way for further research directions.

We started by evaluating the Peering Equilibrium MultiPath (PEMP) routing proposal in real open-source routers. Besides being able to validate experimentally most of the modeling choices, we could revisit some aspects at the light of implementation-specific constraints. Our open source effort in this direction is not ended, and an open perspective is to attempt at upstreaming our modifications in the most recently maintained Quagga fork (e.g., FRR), and also to possibly extend it to other open-source routing and network control systems.

We continued investigating multipath equilibrium routing concept application to edge network traffic engineering, identifying an important feature missing in the LISP protocol, i.e., egress control and policy routing. In this domain, we propose LISP-EC as an extended LISP router behavior to offer outbound traffic control. We could implement it in OpenLISP and experiment its behavior, proving it can lead significant improvement in term of delay when comparing with legacy LISP based approaches. Moreover, our LISP-EC proposal is designed to be fully interoperable with the existing systems.

Addressing a long-term possible Internet routing framework in which edge and transit networks coordinate their routing while managing egress traffic using multipath equilibrium routing, we modeled such a routing interaction using non-cooperative game theory. The proposed modeling leverage on the single-layer multipath equilibrium routing frame-

works applied to BGP and LISP routing, while addressing a specific aspect (the potential threshold configuration affecting the level of path diversity) not definitely addressed in previous works. Further work is needed to add the necessary interfaces to open-source routers or network control-plane platform to support the cross-layer coordination feature we proposed.

We then explored how strategic load balancing decisions could also take place at the terminal or at the connection (transport) level, using multipath transport control protocols such as MPTCP.

First, we perform an analysis to understand at which extent, geographically, using multipath transport or network level communications at the terminal and edge network layers one can increase confidentiality in the current Internet. Internet path diversity could be exploited by means of multi-path transport-layer protocols such as MPTCP, or even network-layer protocol able to operate at the application flow level, when looking at increased security against connection interception Internet-scale attacks. We evaluate through a measurement campaign the ability of securing the Internet communication against man-in-the-middle attack. The results pinpoint countries regions in the geographical Internet where multipath routing would not be helpful as of their current interconnection to the Internet, while pinpointing a large number of countries where it would be effective. We contribute a geographical map showing the robustness level of multipath communications while making our code available.

Enabling the proposed multipath strategies against Internet-scale main-in-the-middle attacks implies having an explicit control on the multipath scheduler able to impose a certain level of load-balancing over the available paths. Moreover, the adoption of a multipath equilibrium load-balancing at the MPTCP level also calls for a scheduler able to ensure load balancing distribution for the sake of connection confidentiality, reliability, etc. In the last chapter, we survey current MPTCP schedulers available with running code, revealing the gap between current schedulers and the explicit load-balancing requirements from the studied applications. Hence we designed an explicit MPTCP scheduler to offer a simpler solution for applications to control the traffic load on each subflow. The resulting scheduler allows us to strictly guarantee a load-balancing distribution, at the expense of the degradation of aggregated throughout, which is the price to pay to meet strict load balancing requirements from multipath-powered applications.

# Software contributions

We list in the following the open source code contributions developed and used for this thesis.

- PEMP-Quagga is a modified version of Quagga routing daemon, with peering equilibrium extensions to the basic BGP routing decision process described in Chapter 3: `https://github.com/routing-games/quagga`.
- Quagga-ext - an extended version of open source network routing software Quagga, to include the new TLVs in both BGP and OSPF daemons: `https://github.com/lip6-lisp/quagga-ext`
- LIP6-LISP OpenLISP Control Plane extension - an extended version of the router control plane with egress control and equilibrium routing capability described in Chapter 4: `https://github.com/routing-games/control-plane`.
- LIP6-LISP OpenLISP Data Plane extension - an extended version of the router with egress control capability and equilibrium routing capability described in Chapter 4: `https://github.com/routing-games/data-plane`.
- MPTCP weighted load balancing scheduler - an extended version of Linux implementation of MPTCP with an explicit load balancing scheduler described in Chapter 7: `https://github.com/routing-games/mptcp`

# Publications

## International conferences with peer review

1. Ho Dac Duy Nguyen and Stefano Secci. "Equilibrium routing: From theory to practice". In: *23rd International Conference on Telecommunications, ICT 2016, Thessaloniki, Greece, May 16-18, 2016*. 2016, pp. 1–7

2. Ho Dac Duy Nguyen and Stefano Secci. "LISP-EC: Enhancing LISP with egress control". In: *2016 IEEE Conference on Standards for Communications and Networking, CSCN 2016, Berlin, Germany, October 31 - November 2, 2016*. 2016, pp. 281–287

3. Ho Dac Duy Nguyen, Chi-Dung Phung, Stefano Secci, Benevid Felix, and Michele Nogueira. "Can MPTCP secure Internet communications from man-in-the-middle attacks?" In: *13th International Conference on Network and Service Management, CNSM 2017, Tokyo, Japan, November 26-30, 2017*. 2017, pp. 1–7

4. Matthieu Coudron, Ho Dac Duy Nguyen, and Stefano Secci. "Enhancing buffer dimensioning for Multipath TCP". in: *7th International Conference on the Network of the Future, NOF 2016, Búzios, Brazil, November 16-18, 2016*. 2016, pp. 1–7. DOI: 10.1109/NOF.2016.7810142

## Submitted

5. Ho Dac Duy Nguyen, Chi-Dung Phung, Stefano Secci, Benevid Felix, and Michele Nogueira. "MPTCP robustness against large-scale man-in-the-middle attacks". In: *Elsevier Computer Networks (submitted)* (2018)

# References


[1] Stefano Secci et al. "Peering equilibrium multipath routing: a game theory framework for internet peering settlements". In: *IEEE/ACM Trans. Netw.* 19.2 (2011), pp. 419–432.

[2] Stefano Secci, Kunpeng Liu, and Bijan Jabbari. "Efficient inter-domain traffic engineering with transit-edge hierarchical routing". In: *Computer Networks* 57.4 (2013), pp. 976–989.

[3] Stefano Secci et al. "Performance-Cost Trade-Off Strategic Evaluation of Multipath TCP Communications". In: *IEEE Trans. Network and Service Management* 11.2 (2014), pp. 250–263.

[4] Yakov Rekhter and Tony Li. *A Border Gateway Protocol 4 (BGP-4)*. RFC 1654. IETF technical report. July 1995.

[5] D. Farinacci et al. *The Locator/ID Separation Protocol (LISP)*. RFC 6830. IETF technical report. Jan. 2013.

[6] *Quagga Routing Software Suite. Online.* URL: `https://www.nongnu.org/quagga/`.

[7] *OpenLISP control plane. Online.* URL: `http://github/lip6-lisp/control-plane`.

[8] A. Ford et al. *TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6824. IETF technical report. Jan. 2013.

[9] Albert-László Barabási, Réka Albert, and Hawoong Jeong. "Scale-free characteristics of random networks: the topology of the world-wide web". In: *Physica A: statistical mechanics and its applications* 281.1-4 (2000), pp. 69–77.

[10] Daniel Walton et al. *Advertisement of multiple paths in BGP*. RFC 7911. 2016.

[11] Qiuyu Peng, Anwar Walid, and Steven H. Low. "Multipath TCP algorithms: theory and design". In: *ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS '13, Pittsburgh, PA, USA, June 17-21, 2013*. 2013, pp. 305–316.

[12] Costin Raiciu et al. “Improving datacenter performance and robustness with multipath TCP”. In: *Proceedings of the ACM SIGCOMM 2011 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Toronto, ON, Canada, August 15-19, 2011*. 2011, pp. 266–277.

[13] Ariel Orda, Raphael Rom, and Nahum Shimkin. “Competitive routing in multiuser communication networks”. In: *IEEE/ACM Trans. Netw.* 1.5 (1993), pp. 510–521.

[14] Rainer Feldmann et al. “Nashification and the Coordination Ratio for a Selfish Routing Game”. In: *Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 - July 4, 2003.Proceedings*. 2003, pp. 514–526.

[15] Raúl H. Etkin, Abhay Parekh, and David Tse. “Spectrum sharing for unlicensed bands”. In: *IEEE Journal on Selected Areas in Communications* 25.3 (2007), pp. 517–528.

[16] Yannis A. Korilis, Aurel A. Lazar, and Ariel Orda. “Architecting Noncooperative Networks”. In: *IEEE Journal on Selected Areas in Communications* 13.7 (1995), pp. 1241–1251.

[17] Se-Young Yun and Alexandre Proutière. “Distributed Proportional Fair Load Balancing in Heterogenous Systems”. In: *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, Portland, OR, USA, June 15-19, 2015*. 2015, pp. 17–30.

[18] Hisao Kameda and Eitan Altman. “Inefficient Noncooperation in Networking Games of Common-Pool Resources”. In: *IEEE Journal on Selected Areas in Communications* 26.7 (2008), pp. 1260–1268.

[19] Ratul Mahajan, David Wetherall, and Thomas E. Anderson. “Negotiation-Based Routing Between Neighboring ISPs”. In: *2nd Symposium on Networked Systems Design and Implementation (NSDI 2005), May 2-4, 2005, Boston, Massachusetts, USA, Proceedings*. 2005.

[20] Wen Xu and Jennifer Rexford. “MIRO: multi-path interdomain routing”. In: *Proceedings of the ACM SIGCOMM 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Pisa, Italy, September 11-15, 2006*. 2006, pp. 171–182.

[21] Mahajan Ratul, David Wetherall, and Thomas Anderson. “Towards Coordinated Interdomain Traffic Engineering”. In: *Proceedings of HotNets-III 2007*. Aug. 2004.

[22] Ratul Mahajan, David Wetherall, and Thomas E. Anderson. "Mutually Controlled Routing with Independent ISPs". In: *4th Symposium on Networked Systems Design and Implementation (NSDI 2007), April 11-13, 2007, Cambridge, Massachusetts, USA, Proceedings.* 2007.

[23] E. Nordmark and M. Bagnulo. *Shim6: Level 3 Multihoming Shim Protocol for IPv6.* RFC 5533. IETF technical report. June 2009.

[24] R. Moskowitz et al. *Host Identity Protocol.* RFC 5201. IETF technical report. Apr. 2008.

[25] Cédric de Launois, Olivier Bonaventure, and Marc Lobelle. "The NAROS Approach for IPv6 Multihoming with Traffic Engineering". In: *Quality for All, 4th COST 263 International Workshop on Quality of Future, Internet Services, QoFIS 2003, Stockholm, Sweden, October 1-2, 2003, Proceedings.* 2003, pp. 112–121.

[26] Matthieu Boutier and Juliusz Chroboczek. "Source-specific routing". In: *Proceedings of the 14th IFIP Networking Conference, Networking 2015, Toulouse, France, 20-22 May, 2015.* 2015, pp. 1–9.

[27] F. Baker and P. Savola. *Ingress Filtering for Multihomed Networks.* RFC 3704. IETF technical report. Mar. 2004.

[28] Ho Dac Duy Nguyen and Stefano Secci. "Equilibrium routing: From theory to practice". In: *23rd International Conference on Telecommunications, ICT 2016, Thessaloniki, Greece, May 16-18, 2016.* 2016, pp. 1–7.

[29] Renata Teixeira et al. "Impact of hot-potato routing changes in IP networks". In: *IEEE/ACM Trans. Netw.* 16.6 (2008), pp. 1295–1307.

[30] *Routing games LIP6 open source project. Online.* URL: `https://routing-games.lip6.fr`.

[31] *The BIRD Internet Routing Daemon Project. Online.* URL: `http://bird.network.cz/`.

[32] Athina Markopoulou et al. "Characterization of failures in an operational IP backbone network". In: *IEEE/ACM Trans. Netw.* 16.4 (2008), pp. 749–762.

[33] Alberto Medina et al. "BRITE: An Approach to Universal Topology Generation". In: *9th International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2001), 15-18 August 2001, Cincinnati, OH, USA.* 2001, p. 346.

[34] A.L. Barabási and R. Albert. "Emergence of scaling in random networks". In: *Science* 286.5439 (1999), p. 509.

[35] Tian Bu and Donald F. Towsley. “On Distinguishing between Internet Power Law Topology Generators”. In: *Proceedings IEEE INFOCOM 2002, The 21st Annual Joint Conference of the IEEE Computer and Communications Societies, New York, USA, June 23-27, 2002*. 2002, pp. 638–647.

[36] Ho Dac Duy Nguyen and Stefano Secci. “LISP-EC: Enhancing LISP with egress control”. In: *2016 IEEE Conference on Standards for Communications and Networking, CSCN 2016, Berlin, Germany, October 31 - November 2, 2016*. 2016, pp. 281–287.

[37] Andra Lutu and Marcelo Bagnulo. “The Tragedy of the Internet Routing Commons”. In: *Proceedings of IEEE International Conference on Communications, ICC 2011, Kyoto, Japan, 5-9 June, 2011*. 2011, pp. 1–5.

[38] Tian Bu, Lixin Gao, and Don Towsley. “On characterizing BGP routing table growth”. In: *Proceedings of the Global Telecommunications Conference, 2002. GLOBECOM ’02, Taipei, Taiwan, 17-21 November, 2002*. 2002, pp. 2185–2189.

[39] *Optimized BGP multi-homing commercial solution. Online*. URL: `http://border6.com/?q=solutions`.

[40] H. von Stackelberg. *The Theory of the Market Economy*. Oxford University Press, 1952.

[41] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory. The Science of Microfabrication*. MIT Press, 1994.

[42] Ho Dac Duy Nguyen et al. “Can MPTCP secure Internet communications from man-in-the-middle attacks?” In: *13th International Conference on Network and Service Management, CNSM 2017, Tokyo, Japan, November 26-30, 2017*. 2017, pp. 1–7.

[43] O.Bonaventure, C.Paasch, and G.Detal. *Use Cases and Operational Experience with Multipath TCP*. RFC 8041. IETF technical report. Jan. 2017.

[44] D. Behaghel et al. *Extensions for Network-Assisted MPTCP Deployment Models*. Internet-Draft. Mar. 2017. URL: `https://tools.ietf.org/id/draft-boucadair-mptcp-plain-mode-10.txt`.

[45] Matthieu Coudron, Ho Dac Duy Nguyen, and Stefano Secci. “Enhancing buffer dimensioning for Multipath TCP”. In: *7th International Conference on the Network of the Future, NOF 2016, Búzios, Brazil, November 16-18, 2016*. 2016, pp. 1–7. DOI: `10.1109/NOF.2016.7810142`.

[46] Alexander Frömmgen et al. “A programming model for application-defined multipath TCP scheduling”. In: *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference, Las Vegas, NV, USA, December 11 - 15, 2017*. 2017, pp. 134–146.

[47] Xavier Corbillon et al. “Cross-layer scheduler for video streaming over MPTCP”. In: *Proceedings of the 7th International Conference on Multimedia Systems, MMSys 2016, Klagenfurt, Austria, May 10-13, 2016*. 2016, 7:1–7:12.

[48] Quentin De Coninck and Olivier Bonaventure. “Multipath QUIC: Design and Evaluation”. In: *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies, CoNEXT 2017, Incheon, Republic of Korea, December 12 - 15, 2017*. 2017, pp. 160–166.

[49] K. Witcher. *Extensions for Network-Assisted MPTCP Deployment Models*. White Paper. SANS Institute, 2005. URL: https://www.sans.org/reading-room/whitepapers/physical/fiber-optics-security-vulnerabilities-1648.

[50] M. Furdek and N. Skorin-Kapov. *Physical-Layer Attacks in Transparent Optical Networks, Optical Communications Systems*. http://www.intechopen.com/books/optical-communications-systems/physical-layer-attacks-in-transparent-optical-networks. 2012.

[51] K. Shaneman and S. Gray. “Optical Network Security: Technical Analysis of Fiber Tapping Mechanisms and Methods for Detection and Prevention”. In: *Proceedings of the IEEE MILCOM 2004*. 2004, pp. 711–716.

[52] J.S. White and A.W. Pilbeam. “An analysis of coupling attacks in high-speed fiber optic networks”. In: *Proceedings of the Enabling Photonics Technologies for Defense, Security, and Aerospace Applications VII*. 2011.

[53] Mauro Conti, Nicola Dragoni, and Viktor Lesyk. “A Survey of Man In The Middle Attacks”. In: *IEEE Communications Surveys and Tutorials* 18.3 (2016), pp. 2027–2051.

[54] A.Pilosov and T.Kapela. *Stealing The Internet - An Internet-Scale Man In The Middle Attack*. 2017. URL: https://www.defcon.org/images/defcon-16/dc16-presentations/defcon-16-pilosov-kapela.pdf.

[55] A. Greenberg. *Hacker Redirects Traffic From 19 Internet Providers to Steal Bitcoins*. 2014. URL: https://www.wired.com/2014/08/isp-bitcoin-theft.

[56] A. Toonk. *BGPstream and The Curious Case of AS12389*. 2017. URL: https://bgpmon.net/bgpstream-and-the-curious-case-of-as12389/.

[57] Sharon Goldberg. “Why is it taking so long to secure internet routing?” In: *Commun. ACM* 57.10 (2014), pp. 56–63.

[58] A. U. Prem Sankar et al. “B-Secure: A Dynamic Reputation System for Identifying Anomalous BGP Paths”. In: *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications - FICTA 2016, Volume 1*. 2016, pp. 767–775.

[59] M. Bagnulo. *Threat Analysis for TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6181. RFC Editor, Mar. 2011. URL: `https://tools.ietf.org/rfc/rfc6181.txt`.

[60] M. Bagnulo et al. *Analysis of Residual Threats and Possible Fixes for Multipath TCP (MPTCP)*. RFC 7430. RFC Editor, July 2015. URL: `https://tools.ietf.org/rfc/rfc7430.txt`.

[61] Dong-Yong Kim and Hyoung-Kee Choi. “Efficient design for secure multipath TCP against eavesdropper in initial handshake”. In: *Proceedings of the 2016 International Conference on Information and Communication Technology Convergence (ICTC 2016)*. 2016.

[62] Mathieu Jadin et al. “Securing multipath TCP: Design & implementation”. In: *2017 IEEE Conference on Computer Communications, INFOCOM 2017, Atlanta, GA, USA, May 1-4, 2017*. 2017, pp. 1–9.

[63] Hackdopi. *Hackdopi*. URL: `http://hackdopi.wikidot.com`.

[64] Lip6-MPTCP. *LIP6-MPTCP open source project repository*. URL: `https://github.com/lip6-mptcp`.

[65] AS-level Topology Archive. *AS-level Topology Archive*. 2015. URL: `http://irl.cs.ucla.edu/topology`.

[66] CAIDA. *Archipelago (Ark) measurement infrastructure*. URL: `http://www.caida.org/projects/ark`.

[67] IRR. *Internet Routing Registries*. URL: `http://www.irr.net`.

[68] Routeviews. *University of Oregon Route View Projects*. URL: `http://www.routeviews.org`.

[69] RIPE RIS. *RIPE Routing Information Service*. URL: `https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris`.

[70] Ricardo V. Oliveira et al. “The (in)completeness of the observed internet AS-level structure”. In: *IEEE/ACM Trans. Netw.* 18.1 (2010), pp. 109–122.

[71] Cyclops. *Cyclops*. URL: https://cyclops.cs.ucla.edu.

[72] Thomas Erlebach et al. "Connectivity Measures for Internet Topologies on the Level of Autonomous Systems". In: *Operations Research* 57.4 (2009), pp. 1006–1025.

[73] Rowan Klöti et al. "Policy-compliant path diversity and bisection bandwidth". In: *2015 IEEE Conference on Computer Communications, INFOCOM 2015, Kowloon, Hong Kong, April 26 - May 1, 2015*. 2015, pp. 675–683.

[74] Giuseppe Di Battista et al. "Computing the types of the relationships between autonomous systems". In: *IEEE/ACM Trans. Netw.* 15.2 (2007), pp. 267–280.

[75] Réka Albert and Albert-László Barabási. "Statistical mechanics of complex networks". In: *CoRR* cond-mat/0106096 (2001). URL: http://arxiv.org/abs/cond-mat/0106096.

[76] M. Kühne. *Update on AS Path Lengths Over Time*. 2012. URL: https://labs.ripe.net/Members/mirjam/update-on-as-path-lengths-over-time.

[77] Lixin Gao. "On inferring autonomous system relationships in the internet". In: *IEEE/ACM Trans. Netw.* 9.6 (2001), pp. 733–745.

[78] T. Bates, P. Smith, and G. Huston. *CIDR Report*. 2017. URL: http://www.cidr-report.org.

[79] Bernhard Ager et al. "Anatomy of a large european IXP". In: *ACM SIGCOMM 2012 Conference, SIGCOMM '12, Helsinki, Finland - August 13 - 17, 2012*. 2012, pp. 163–174.

[80] UNSD. *Standard country or area codes for statistical use (M49)*. 2017. URL: http://unstats.un.org/unsd/methods/m49/m49regin.htm.

[81] Cable Map. *Greg's Cable Map*. 2017. URL: http://cablemap.info.

[82] Nicole Starosielski et al. "Critical Nodes, Cultural Networks: Re-mapping Guam's Cable Infrastructure". In: *Amerasia Journal* 37.3 (2011), pp. 18–27. URL: https://doi.org/10.17953/amer.37.3.m70071273lt62754.

[83] M. Boucadair et al. *Network-Assisted MPTCP: Use Cases, Deployment Scenarios and Operational Considerations*. Internet-Draft. Dec. 2016. URL: https://tools.ietf.org/id/draft-nam-mptcp-deployment-considerations-01.txt.

[84] Matthieu Coudron et al. "Cross-layer cooperation to boost multipath TCP performance in cloud networks". In: *IEEE 2nd International Conference on Cloud Networking, CloudNet 2013, San Francisco, CA, USA, November 11-13, 2013*. 2013, pp. 58–66.

[85] Yacine Benchaib, Stefano Secci, and Chi-Dung Phung. "Transparent Cloud Access Performance Augmentation via an MPTCP-LISP Connection Proxy". In: *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems, ANCS 2015, Oakland, CA, USA, May 7-8, 2015*. 2015, pp. 201–202.

[86] M. Tuexen and R. Stewart. *Stream Control Transmission Protocol (SCTP) Chunk Flags Registration*. RFC 6096. RFC Editor, Jan. 2011. URL: https://tools.ietf.org/html/rfc6096.

[87] Xavier Misseri, Jean-Louis Rougier, and Damien Saucez. "Internet routing diversity for stub networks with a Map-and-Encap scheme". In: *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*. 2012, pp. 2861–2866.

[88] A. Lange. *Issues in Revising BGP-4*. Internet-Draft. Mar. 2012. URL: https://tools.ietf.org/id/draft-ietf-idr-bgp-issues-06.txt.

[89] Juniper. *Configuring BGP to Select Multiple BGP Paths*. JUNOS 8.2 Routing Protocols Configuration Guide in Chapter 33 BGP Configuration Guidelines pp:571-572. URL: https://www.net.t-labs.tu-berlin.de/teaching/ss08/RL_labcourse/docs/04-juniper-bgp.pdf.

[90] Cisco. *BGP Best Path Selection Algorithm*. Cisco guide, Troubleshooting tech notes, Document ID:13753. Sept. 2016. URL: https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html.

[91] Eric Elena, Jean-Louis Rougier, and Stefano Secci. "Characterisation of AS-level path deviations and multipath in Internet routing". In: *Next Generation Internet (NGI), 2010 6th EURO-NF Conference on, Paris, France, June 2-4, 2010*. 2010, pp. 1–7.

[92] Christoph Paasch et al. "Experimental evaluation of multipath TCP schedulers". In: *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop, CSWS '14, Chicago, Illinois, USA, August 18, 2014*. 2014, pp. 27–32. DOI: 10.1145/2630088.2631977. URL: http://doi.acm.org/10.1145/2630088.2631977.

[93] Kae Won Choi et al. "Optimal load balancing scheduler for MPTCP-based bandwidth aggregation in heterogeneous wireless environments". In: *Computer Communications* 112 (2017), pp. 116–130. DOI: 10.1016/j.comcom.2017.08.018. URL: https://doi.org/10.1016/j.comcom.2017.08.018.

[94] Yeon-sup Lim et al. “ECF: An MPTCP Path Scheduler to Manage Heterogeneous Paths”. In: *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies, CoNEXT 2017, Incheon, Republic of Korea, December 12 - 15, 2017*. 2017, pp. 147–159. DOI: 10.1145/3143361.3143376. URL: http://doi.acm.org/10.1145/3143361.3143376.

[95] Costin Raiciu et al. “How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP”. In: *Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2012, San Jose, CA, USA, April 25-27, 2012*. 2012, pp. 399–412. URL: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/raiciu.

[96] Fan Yang and Paul D. Amer. “Work in progress: Using one-way communication delay for in-order arrival MPTCP scheduling”. In: *9th International Conference on Communications and Networking in China, Maoming, China, August 14-16, 2014*. 2014, pp. 122–125. DOI: 10.1109/CHINACOM.2014.7054271. URL: https://doi.org/10.1109/CHINACOM.2014.7054271.

[97] Nicolas Kuhn et al. “DAPS: Intelligent delay-aware packet scheduling for multipath transport”. In: *IEEE International Conference on Communications, ICC 2014, Sydney, Australia, June 10-14, 2014*. 2014, pp. 1222–1227. DOI: 10.1109/ICC.2014.6883488. URL: https://doi.org/10.1109/ICC.2014.6883488.

[98] Simone Ferlin et al. “BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks”. In: *2016 IFIP Networking Conference, Networking 2016 and Workshops, Vienna, Austria, May 17-19, 2016*. 2016, pp. 431–439. DOI: 10.1109/IFIPNetworking.2016.7497206. URL: https://doi.org/10.1109/IFIPNetworking.2016.7497206.

[99] C. Paasch and S. Barre. *Multipath TCP in the Linux Kernel*. URL: https://www.multipath-tcp.org.

[100] Ho Dac Duy Nguyen et al. “MPTCP robustness against large-scale man-in-the-middle attacks”. In: *Elsevier Computer Networks (submitted)* (2018).