Optimal deployment of virtual network functions for securing telecommunication networks against distributed denial of service attacks: a robust optimization approach

Céline Gicquel, Sonia Vanier, Alexandros Papadimitriou

Optimal deployment of virtual network functions for securing telecommunication networks against distributed denial of service attacks: a robust optimization approach

Céline Gicquel[1], Sonia Vanier[2], Alexandros Papadimitriou [3]

[1] Laboratoire Interdisciplinaire des Sciences du Numérique
Université Paris Saclay, France
[2] Laboratoire d'Informatique de l'Ecole Polytechnique
Université Paris 1, France
[3] Orange Labs Products & Services, France

Abstract: Distributed Denial of Service (DDoS) cyberattacks represent a major security risk for network operators and internet service providers. They thus need to invest in security solutions to protect their network against DDoS attacks. The present work focuses on deploying a network function virtualization based architecture to secure a network against an on-going DDoS attack. We assume that the target, sources and volume of the attack have been identified. However, due to 5G network slicing, the exact routing of the illegitimate flow in the network is not known by the internet service provider. We seek to determine the optimal number and locations of virtual network functions in order to remove all the illegitimate traffic while minimizing the total cost of the activated virtual network functions. We propose a robust optimization framework to solve this problem. The uncertain input parameters correspond to the amount of illegitimate flow on each path connecting an attack source to the target and can take values within a predefined uncertainty set. In order to solve this robust optimization problem, we develop an adversarial approach in which the adversarial sub-problem is solved by a Branch & Price algorithm. The results of our computational experiments, carried out on medium-size randomly generated instances, show that the proposed solution approach is able to provide optimal solutions within short computation times.

Keywords: Telecommunication networks, Cybersecurity, Distributed denial of service, Network function virtualization, Robust optimization, Adversarial approach, Mixed-integer linear programming, Branch & Price, Column Generation.

1 Introduction

Distributed Denial of Service (DDoS) attacks are among the top threats to network operators and internet service providers (ISPs). A distributed denial of service is a type of cyberattack in which multiple compromised computer systems attack a target, such as a server or a website, and cause a denial of service for its legitimate users. DDoS flooding attacks are often launched through the use of botnets. A botnet is a network of user computers or Internet of Things (IoT) devices that are remotely controlled by a hacker through malwares. Under the direction of the hacker, an army of botnets can launch a DDoS attack against a target by simultaneously sending to it a large amount of traffic or service requests. The flood of incoming messages, connection requests or malformed packets exhausts the resources of the target and forces it to slow down or even shut down, thereby preventing it to provide service to its legitimate users.

In recent years, the number, intensity and diversity of DDoS attacks have increased dramatically. Thus, in 2016, the BBC website was targeted by a DDoS attack of more than 600 Gbps and was unavailable for a few hours (Khandelwal, 2016). More recently, Amazon announced that its AWS Shield service mitigated a 2.3Tbps DDoS attack in February 2020 (AWS, 2020). There is also a continuous appearance of new attack vectors, i.e. new techniques enabling hackers to launch a DDoS attack, and new combinations of attack vectors: see e.g. the recent report provided in (Netscout Systems, 2020) and (FBI, 2020). This trend is likely to continue and even accentuate in the near future. Namely, with the development of the Internet of Things, systems based on smart devices (such as sensors) connected to the Internet are widely deployed. This increases the vulnerability of networks and the number of potential DDoS targets: see among others Rahimi et al. (2018), Akpakwu et al. (2018), Fysarakis et al. (2016) and Silva et al. (2020). Furthermore, as mentioned e.g. by Grawe (2020), the COVID-19 pandemic has forced organizations to accelerate their digital transformation plans, thus further increasing the attack surface for hackers and criminals.

DDoS attacks can be very damaging for the organization they target. For instance, a survey carried out in 2017 by the cybersecurity company Kapersky Lab estimated the average cost of a DDoS attack for large (1000+) businesses to be around \$2.3 millions (Berard, 2018). This cost mainly comprises the cost incurred in fighting the attack and restoring service, the investment in an offline or back-up system while online services are unavailable, the loss of revenue or business opportunities and the loss of trust from customers and partners.

Many DDoS mitigation solutions have been proposed to protect organizations' networks, servers and services. The traditional approach consists in deploying specialized hardware security appliances that are fixed in terms of strength, functionality and capacity. This means in particular that the location and capacity (in terms of the volume of malicious traffic it can process) of the defense appliances are determined in advance, before the DDoS attacks actually take place. As explained e.g. by Fayaz et al. (2015), companies are thus forced to over provision by deploying appliances capable of handling a high but predefined volume of attack at several points in the network. A second approach consists in using an external cloud-based DDoS protection service. In this case, when under attack, all the incoming traffic to the targeted service is diverted towards a cloud scrubbing center managed by a third party. In the scrubbing center, the traffic is inspected and only the legitimate traffic is routed back towards its destination. These cloud-based services are more flexible and scalable than dedicated hardware appliances. They however raise concerns relative to customers' privacy violation and often lead to increased latency (Alharbi and Aljuhani, 2017).

Network Function Virtualization (NFV) is a recent network architecture concept in which network functions (e.g. network address translation, firewalling, domain name service, etc.) are implemented as software and deployed as virtual machines running on general purpose commodity hardware (Jakaria et al., 2016). Virtualization increases manageability, reliability and performance of the network and allows a flexible and dynamic implementation of the network services, which significantly reduces the cost of the infrastructure and simplifies the deployment of new services. These numerous benefits have convinced operators to largely embrace virtualization of network functions: see e.g. Donovan (2014) and Savi (2018).

NFV offers new possibilities to counter DDoS attacks. In particular, its flexibility and reactivity allows to postpone the DDoS defense deployment after the attack is detected. This allows to place adapted defense mechanisms where they are needed and to launch them depending on the scale of the attack (Fayaz et al., 2015). Moreover, NFV-based mitigation approaches do not require the use of an external service provider, which reduces the privacy and latency issues encountered by cloud-based DDoS mitigation.

As mentioned e.g in Alharbi and Aljuhani (2017), Silva et al. (2020) and Jakaria et al. (2016), NFV is a promising technology to mitigate DDoS attacks. However, in order to fully leverage its potential, some difficulties should be overcome. First, virtual network functions (VNFs) are instantiated on virtual machines. These virtual machines consume the limited computing resources (CPU, memory,...) of the servers on which they run. When designing an NFV-based infrastructure to counter an on-going DDoS attack in a network, these limitations in the available computing resources should be taken into account. The number of VNFs which can be instantiated at each node of the network depends on the resources of the servers located at this node. Second, each VNF has a limited filtering capacity and can thus remove only part of the attack flow. The filtering capacity of a VNF corresponds to the maximum amount of malicious flow an instance of this VNF can stop. If the malicious flow going through a VNF is larger than its filtering capacity, the excess malicious flow is forwarded in the network and may thus reach its target. This translates into the fact that, in order to stop all the malicious traffic of an attack, several VNFs may have to be placed at different nodes on the paths used to route the flow between its source and its target. A carefully optimized VNF placement strategy taking into account both the limited computing resources in the network and the limited filtering capacity of a VNF is thus needed.

In the present work, we focus on the deployment of an architecture based on the NFV technology to secure a network against DDoS attacks. We assume that the on-going attack has been detected and that its ingress points, its volume and its target have been identified. Based on this information, we seek to determine the optimal number and location of VNFs in order to remove all the illegitimate traffic while trying to minimize the total cost of the activated VNFs.

We take here the perspective of an internet service provider (ISP) aiming at providing a DDoS mitigation service to its customers in a 5G network. Among the key features of 5G networks is network slicing: see e.g. Vyakaranam and Krishna (2018). Network slicing is an architecture in which the physical network infrastructure managed by an ISP is partitioned into multiple virtual independent networks termed slices. Each slice is an iso-

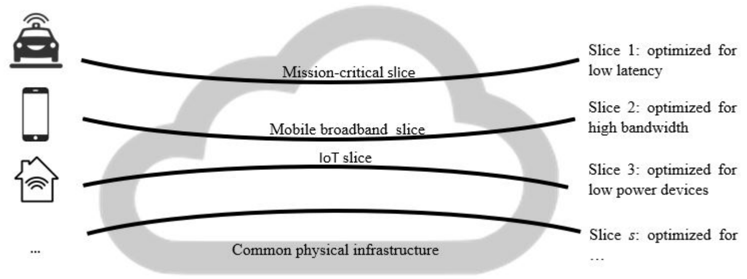
Mission-critical slice
Mobile broadband slice
IoT slice
Common physical infrastructure
...
Slice 1: optimized for low latency
Slice 2: optimized for high bandwidth
Slice 3: optimized for low power devices
Slice s: optimized for ...

Figure 1: 5G network slicing

lated end-to-end network which is lent by the ISP to a single customer and is adapted to meet the specific requirements of this customer in terms of quality of service (bandwidth, reliability, latency, etc.). See Figure 1 for a graphical illustration of 5G network slicing. Network slicing thus provides an opportunity to the ISP to flexibly configure its physical network so as to simultaneously fulfill quality-of-service requirements that may strongly vary from one customer to the next. However, on each slice of the network, the routing of the flow will not be managed anymore by the ISP but by its customer which will rely on its own proprietary routing algorithms. This significantly enhances the difficulty for the ISP of providing a DDoS mitigation service as it will not control the exact routing of the malicious flow that needs to be stopped.

Our main contributions are thus threefold. First, we present a robust optimization (RO) model to optimally design an NFV-based DDoS mitigation infrastructure in the context of 5G network slicing. This model explicitly takes into account the fact that the ISP is not aware of the exact routing of the attack flow. This is done by considering the malicious flow routing as an input parameter of the optimization problem which is subject to uncertainty. To the best of our knowledge, this is the first time such a robust optimization model is investigated to design a DDoS mitigation infrastructure in 5G networks. Second, we propose an efficient algorithm to solve the robust optimization problem. This algorithm relies on an adversarial approach which decomposes the problem into a master problem and an adversarial sub-problem. The master problem seeks to optimally place the filtering VNFs while taking into account a limited number of possible malicious flow routings. The adversarial sub-problem aims at finding the worst flow routing for a given VNF infrastructure and is used to generate new routings, i.e. new constraints, to be taken into account in the master problem. Moreover, as the adversarial sub-problem involves an exponential number of decision variables, we develop a Branch & Price algorithm to solve it in a computationally efficient way. Third, we provide the results of computational experiments carried out on medium-size randomly generated instances. These results show that the proposed solution algorithm is able to efficiently provide optimal or near-optimal solutions within short computation times.

The paper is organized as follows. We first review the related literature in Section 2. We then provide in Section 3 a formal description of the problem, discuss its modeling as a robust optimization problem and present a complexity analysis. We describe in Section 4 the adversarial solution approach proposed to solve this RO problem. Numerical results carried out on medium-size randomly generated instances are provided in Section 5. Finally, Section 6 gives a conclusion and some research perspectives.

2 Related works

We provide in this section a brief overview of the works closely related to ours. We first discuss papers proposing NFV-based infrastructures for DDoS mitigation. We then consider papers dealing with the optimal placement of virtual network functions in a network for generic cases and focus on two recent works studying the optimal placement of VNFs in a network for the specific case of DDoS mitigation. Finally, we review the literature on the network flow interdiction problem as this problem shares some common features with our problem.

2.1 NFV-based infrastructures for DDoS mitigation

NFV-based infrastructures to counter DDoS attacks are investigated in several recent papers. Fung and McCormick (2015) propose a solution based on request prioritization to protect an online application server from a DDoS attack. The incoming requests to the servers are categorized into two priority levels: requests from trusted sources are assigned a high priority and are guaranteed to be served whereas requests from untrusted sources are assigned a low priority and will be served based on the resource availability on the server. The proposed architecture makes use of a VNF for priority assignment and flow dispatching. Another widely used mitigation strategy against DDoS attacks is flow filtering: see e.g. Silva et al. (2020). Basically, flow filtering consists in analyzing the information contained in the headers of the data packets to block the malicious flow. The filtering process thus exploits information such as the source and destination IP addresses, the origin and destination ports or the network layer protocol to identify malicious packets and drop them. Jakaria et al. (2016), Rashidi et al. (2018) and Jakaria et al. (2019) investigate a DDoS mitigation framework in which this filtering process is carried out by VNFs which are dynamically allocated as needed depending on the volume of the attack. More precisely, their framework aims at protecting an online product server against a specific type of DDoS attacks, termed SYN floods, which exploit some weak points of the TCP internet protocol. This framework involves a dispatcher/load balancer which receives the incoming packets from the internet and distributes them to filtering VNFs instantiated on commodity servers. These VNFs verify the source IP address of each packet, drop the packet in case it is illegitimate or forward it to the product server in case its source is white-listed. Finally, Fayaz et al. (2015) and Alharbi and Aljuhani (2017) propose DDoS mitigation infrastructures in which VNFs may have a variety of functions depending on the type of the attack.

2.2 Optimal placement of virtual network functions

In their survey on network function placement, Li and Qian (2016) distinguish between two types of placement problems. The first one corresponds to the case where independent network functions, i.e. functions which do not interact with one another, should be placed in the network. The second one, called service chaining, applies when each flow must traverse a predefined sequence of network functions (such as firewall $\rightarrow$ intrusion detection system $\rightarrow$ proxy) between its ingress point and its destination point in the network. Note that the problem under study in this work belongs to the first type of problem as we consider a single type of network functions. We refer the reader to Demirci and Sagiroglu (2019) for a general overview of the literature on the optimal placement of virtual network functions and focus in what follows on the specific context of DDoS mitigation.

To the best of our knowledge, there are only two works dealing with the problem of optimally placing VNFs in a network to counter an on-going DDoS attack. Fayaz et al. (2015) develop the Bohatei system based on NFV and SDN (software-defined networking). Their system includes a resource manager which determines the type, number and location of VNFs to be instantiated based on the available information on the ingress points, target, type and volume of the on-going attack so as to minimize the costs related to the malicious flow traffic. They consider a case in which the mitigation of each type of DDoS attack (e.g. SYN food, DNS amplification or UDP flood) is a multi-step process requiring the use of different types of network functions. They formulate the underlying optimization problem as a mixed-integer linear program and solve it using a two-step heuristic. Jakaria et al. (2019) consider an architecture involving two types of VNFs, namely dispatchers and filtering agents, to counter SYN flood attacks. They deploy these VNFs through virtual machines running on commodity servers. The objective is to process all the incoming traffic while using a minimum number of commodity servers. Their mathematical model is formulated as a constraint satisfaction (SAT) problem (Apt, 2003). It takes into account the limited computing resources of each commodity server, the limited bandwidth of the links between the dispatchers and the filtering agents and the relation between the packet filtering rate of a VNF and the computing resources allocated to the virtual machine on which it is instantiated. Note that, contrary to the problem under study here, both Fayaz et al. (2015) and Jakaria et al. (2019) assume in their problem modeling that the flow of the attack, once detected, can be flexibly routed towards the launched virtual machines.

2.3 Network flow interdiction problem

In the network flow interdiction problem, an attacker and a defender take measurements on a capacitated network. The defender seeks to maximize the flow through the network, while the attacker suppresses some arcs to minimize the maximum flow. Each arc has a removal cost. Thus, the goal for the attacker is to select a subset of arcs to remove without exceeding a fixed budget. The network interdiction problem is known to be an NP-complete problem: see Phillips (1993) and Wood (1993). However, it can be solved in polynomial time for certain categories of graphs such as planar graphs (Phillips, 1993; Wollmer, 1964).

Different classes of the network flow interdiction problem are studied in the literature: see e.g. Church et al. (2004). Baffier et al. (2018) investigate an adaptive network interdiction flow problem. The defender aims to maximize the flow value and the attacker seeks to minimize the remaining flow value by removing a set of k links. The goal is to find a robust flow against any k edge attack. A bilevel optimization framework is developed to address this problem. Naoum-Sawaya and Ghaddar (2017) also formulate the problem as a bi-level mixed-integer program. An iterative cutting plane algorithm is proposed and implemented in a branch-and-cut approach. Lim and Smith (2007) study problems with discrete and continuous interdictions. They describe a linearized model to optimize the discrete network interdiction problem and compare it to a penalty model. For the continuous case, they describe an optimal partitioning algorithm as well as a heuristic procedure to estimate the optimal value of the objective function. Altner et al. (2010) propose two classes of polynomially separable valid inequalities for the Maximum Flow Network Interdiction Problem. An approximation factor-preserving reduction from a simpler interdiction problem is also developed. Lei et al. (2018) consider maximum flow interdiction problem under interdiction-effect uncertainties. The problem is characterized as a Stackelberg game. They consider risk-neutral and risk-averse behaviors of the two players. Five bi-level/tri-level programming models for different risk-preference combinations are investigated. An application of the network interdiction problem to security issues is studied by Guo et al. (2016). The problem is to optimally interdict illegal network flow in the context of the containment of the flow of drugs through the US-Mexico border patrol. A Stackelberg game model for network interdiction flow with a single source-destination flow is presented. The proposed solution approach is based on column generation and constraint generation algorithm. Fu and Modiano (2019) propose a new paradigm for network interdiction that models scenarios. The interdiction is performed through injecting bounded-value flows to maximally reduce the throughput of the residual network. They study two problems under the paradigm: deterministic flow interdiction and robust flow interdiction. An algorithm with logarithmic approximation ratio is developed.

Note that the present works investigates a defender-attacker problem which significantly differs from the network flow interdiction problem. Namely, in our case, the defender, i.e. the internet service provider, allocates security resources without changing the network topology. Virtual functions are deployed on network nodes in order to suppress attacking flows but this security mechanism does not remove any component (node or link) in the network. Our goal is to minimize the costs of VNFs deployment while gradually eliminating the malicious flows, rather than destroying links to prevent the attacker from reaching its target. This implies significant differences in the mathematical formulations of the problem. The methodologies proposed in the existing works can therefore not be directly applied to our problem.

3 Problem description and mathematical modeling

In this section, we first describe in a more formal way the optimization problem under study and present the proposed robust optimization model. We then provide a small illustrative example. Finally, we discuss the use of aggregated filtering constraints in the problem formulation and study its complexity status.

3.1 Problem definition

The network topology is modeled by a digraph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ in which $\mathcal{N}$, the set of nodes, represents specific equipment in the network and $\mathcal{L}$, the set of arcs, corresponds to the links that can be used to route the traffic. The routing of the traffic in the network is limited by the bandwidth b_l of each link l. In practice, part of this bandwidth is used to route the legitimate traffic in the network. In the present work, for the sake of simplicity, we assume that the bandwidth consumed by the legitimate traffic is negligible as compared to the one consumed by the illegitimate traffic. We thus consider that the illegitimate traffic may use all the bandwidth of a link if needed.

The illegitimate traffic corresponding to the on-going DDoS attack is represented as a set $\mathcal{A}$ of attacks: attack $a \in \mathcal{A}$ corresponds to an illegitimate traffic of F^a Mbps between a source $s^a \in \mathcal{N}$ and the target $t \in \mathcal{N}$ of the DDoS attack. Source nodes, $\{s_a, a \in \mathcal{A}\}$, are network access nodes (also termed gateways) managed by the ISP. They are able to compute the number of incoming packets and to detect suspicious traffic entering the network. In contrast, the target node t is a strategic node belonging to an external network managed by a customer which subscribed to a security service provided by the ISP. The ISP must thus secure this node against the on-going DDoS attack but it is not allowed to install any software (i.e. to deploy VNFs) on this node. The malicious traffic corresponding to the attack thus has to be stopped before it reaches t.

As explained in the introduction, in the present work, we consider the case in which an ISP lends slices of its physical network infrastructure to its customers and each of these customers uses its own flow routing algorithms to route the flow on the slice assigned to it. The result is that, by the time the ISP has to decide on the NFV-based DDoS mitigation infrastructure, it does not know the exact routing of the malicious flow to be stopped. Let $\mathcal{P}^a$ be the set of all potential paths between s^a and t for attack a. $\mathcal{N}^{a,p}$ (resp. $\mathcal{L}^{a,p}$) denotes the set of nodes (resp. the set of links) belonging to path $p \in \mathcal{P}^a$ and $\mathcal{P}^a(n)$ denotes the subset of paths of $\mathcal{P}^a$ going through node n. The amount of malicious flow of attack $a \in \mathcal{A}$ on path $p \in \mathcal{P}^a$, denoted by $\tilde{f}^{a,p}$, is thus subject to uncertainty. However, even if the exact value of parameter $\tilde{f}^{a,p}$ is unknown, there are some restrictions on its potential value. Namely, we know that the total amount of malicious flow routed on the paths belonging to $\mathcal{P}^a$ may not be greater than F^a, the amount of illegitimate traffic of attack a. Moreover, the malicious flow routing must comply with the limited bandwidth of each link. These two pieces of information should be exploited as best as possible to avoid using more network resources than necessary for the DDoS attack mitigation.

In the considered DDoS mitigation framework, VNFs are used to filter and stop the illegitimate traffic before it reaches its target. A VNF instantiated on a node $n \in \mathcal{N}$ of the network can be seen as a software running on the server located at node n and filtering the flow going through n. As explained in Section 2, this filtering process mainly consists in selectively stopping unwanted traffic by exploiting the information contained in the header of each data packet. This information can be the source, destination, port or routing protocol of the data packet to be processed. The filtering capacity of a VNF corresponds to the number of packets it can receive and process per second: if the malicious flow the VNF has to handle is larger than its filtering capacity, the excess flow is forwarded in the network and may thus reach its target. This filtering capacity is linked to the amount of computing resources consumed by the VNF on the server where it is instantiated. Indeed, data packets arriving at the VNF are first extracted and stored in memory. They then undergo several processing cycles on the available CPUs in order to analyze their content. Thus, the number of CPUs allocated to the VNF strongly limits its packet processing rate. Moreover, widely used filtering rules consist in analyzing the destination of a set of packets and storing them in memory. If there are too many packets targeting the same destination at the same time, these packets are considered as suspicious and are discarded.

Consequently, the filtering process requires some memory to implement the malicious traffic filtering rules. The set of available VNF types is described by $\mathcal{V} = \{1, ..., V\}$. A VNF of type v is characterized by its filtering capacity ϕ^v, its cost K^v and its computing resources consumption. The set of computing resources (CPU, memory, etc.) is denoted by $\mathcal{R} = \{1, ..., R\}$. Let k^{rv} be the amount of computing resource r required by the instantiation of one VNF of type v and Cap_n^r the amount of computing resource r available at node n.

Table 3.1 summarizes the notation used to describe the input parameters of the various mathematical models throughout the paper.

The optimization problem consists in identifying the location and number of VNFs to be placed in the network so as to stop all the malicious flow before it reaches its target, and this whatever its routing through the network, while minimizing the cost of the instantiated VNFs and complying with the limitations on the computing resources.

3.2 Mathematical formulation

We propose to handle this optimization problem using a robust optimization (RO) approach. A robust optimization problem is an optimization problem in which some parameters are subject to uncertainty. In a RO problem, the uncertainty on the input parameters is not described in terms of probability distributions but rather by means of an uncertainty set containing all the possible values that these parameters may take. Solving a RO problem consists in finding a solution which is feasible for any realization of the uncertain parameters in the uncertainty set and which provides the best possible value of the objective function. The reader is referred to Gorissen et al. (2015) for a practical introduction on robust optimization.

In the present case, the routing of the malicious flow in the network is not known by the ISP. The amount of malicious flow of attack $a \in \mathcal{A}$ on path $p \in \mathcal{P}^a$, $\tilde{f}^{a,p}$, can thus be seen as an uncertain input parameter for the problem of optimally placing VNFs to counter the DDoS attack. However, as mentioned in Subsection 3.1, even if the exact value of parameter $\tilde{f}^{a,p}$ is unknown, its value should comply with two restrictions. First, for each attack a, the total flow routed in the network may not be larger than the total attack traffic, i.e. we have $\sum_{p \in \mathcal{P}^a} \tilde{f}^{a,p} \leq F^a$ for each attack $a \in \mathcal{A}$. Second, the flow routed on each link l of the network may not exceed the bandwidth b_l of this link. We thus have $\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a \text{ s.t. } l \in \mathcal{L}^{a,p}} \tilde{f}^{a,p} \leq b_l$ for each link l.

$\mathcal{A}$	Set of all on-going attacks to be stopped
$\mathcal{C}$	Collection of restricted sets of paths
$\mathcal{G}$	Graph representing the telecommunication network
$\mathcal{L}$	Set of links used to route the traffic in the network
$\mathcal{L}^{a,p}$	Set of all links belonging to path $p \in \mathcal{P}^a$
$\mathcal{N}$	Set of nodes, i.e. of pieces of equipment in the network
$\mathcal{N}^{a,p}$	Set of all nodes belonging to path $p \in \mathcal{P}^a$
$\mathcal{N}(\tilde{f})$	Subset of nodes through which part of the malicious flow transits when it is routed according to routing $\tilde{f}$
$\mathcal{P}^a$	Set of all paths between s^a and t
$\mathcal{P}^a(n)$	Subset of paths in $\mathcal{P}^a$ such that $n \in \mathcal{N}^{a,p}$
$\mathcal{P}^a_R$	Restricted set of paths for attack a
$\mathcal{R}$	Set of computing resources
$\mathcal{U}$	Uncertainty set
$\mathcal{U}_R$	Restricted uncertainty set
$\mathcal{V}$	Set of available VNF types
A	Number of attacks
b_l	Bandwidth of link l
Cap^r_n	Amount of computing resource r available at node n
F^a	Total illegitimate traffic of attack a
$\tilde{f}^{a,p}$	Unknown amount of malicious flow of attack $a \in \mathcal{A}$ routed on path $p \in \mathcal{P}^a$
$\tilde{f}$	Unknown routing of the DDoS attack ; $\tilde{f} = \{\tilde{f}^{a,p} \text{ s.t. } a \in \mathcal{A}, p \in \mathcal{P}^a\}$
$\overline{f}$	Given routing belonging to the uncertainty set $\mathcal{U}$
K^v	Cost of instantiating a VNF of type v
k^{rv}	Amount of resource r required to instantiate a VNF of type v
R	Number of computing resources
s^a	Source, i.e. ingress point, of attack a
t	Target common to all on-going attacks
V	Number of available VNF types
$\overline{x}$	Given placement of the filtering VNFs in the network
ϕ^v	Filtering capacity of a VNF of type v

Table 1: Notation for the input parameters used in the various mathematical models

Problem RVNFD	
x_n^v	Number of VNFs of type v placed at node n
Problem TP	
$d_n^{a,p}$	Amount of filtering capacity placed at node n allocated to stopping the malicious flow relative to attack a and routed on path p
Problem $DMP(\mathcal{U}_R)$	
x_n^v	Number of VNFs of type v placed at node n
Problems $AP(\overline{x})$, $RAP(\overline{x},\mathcal{C})$ and $\underline{RAP}(\overline{x},\mathcal{C})$	
$f^{a,p}$	Amount of flow related to attack a routed on path p
z_n	$z_n = 1$ if some malicious flow transits through n, to 0 otherwise

Table 2: Notation for the decision variables used in the various mathematical models

This means that the uncertain malicious flow routing, $\tilde{f} = \{\tilde{f}^{a,p}$ s.t. $a \in \mathcal{A}, p \in \mathcal{P}^a\}$, belongs to the uncertainty set $\mathcal{U}$ defined by:

$$\begin{aligned} \mathcal{U} = \{\tilde{f} \geq 0 | \sum_{p \in \mathcal{P}^a} \tilde{f}^{a,p} \leq F^a, & \qquad \forall a \in \mathcal{A} \\ \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a \text{ s.t. } l \in \mathcal{L}^{a,p}} \tilde{f}^{a,p} \leq b_l, & \qquad \forall l \in \mathcal{L}\} \end{aligned}$$

Note that the first restriction on $\tilde{f}$ is expressed as an inequality rather than as an equality. Namely, in some cases, it may not be possible to route all the malicious flow of the attack in the network due to the limited bandwidth of the network links. In these cases, expressing the restriction as an equality would lead to an empty uncertainty set. For the RO problem, this would mean that there is no malicious flow routed in the network, i.e. no malicious flow to be stopped by the VNF-based infrastructure, whereas in practice part (but not all) of the attack flow will be routed in the network.

We introduce the integer decision variables x_n^v which represent the number of VNFs of type v placed at node n: see Table 2 for a summary of the decision variables used in the various mathematical models investigated in the paper.

Using the previously introduced notation,the robust virtual network function deployment problem, which will be denoted by Problem RVNFD, is formulated as follows:

$$Z^* = \min \sum_{v\in\mathcal{V}} \sum_{n\in\mathcal{N}} K^v x_n^v \tag{1}$$

$$\sum_{v\in\mathcal{V}} k^{rv} x_n^v \leq Cap_n^r \qquad \forall n \in \mathcal{N}, \forall r \in \mathcal{R} \tag{2}$$

$$\sum_{n\in\mathcal{N}(\tilde{f})} \sum_{v\in\mathcal{V}} \phi^v x_n^v \geq \sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a} \tilde{f}^{a,p} \qquad \forall \tilde{f} \in \mathcal{U} \tag{3}$$

$$x_t^v = 0 \qquad \forall v \in \mathcal{V} \tag{4}$$

$$x_n^v \text{ integer} \qquad \forall n \in \mathcal{N}, \forall v \in \mathcal{V} \tag{5}$$

The objective (1) is to minimize the total costs of the deployed VNFs. Constraints (2) ensure that the VNFs installed at each node n do not consume more than the available computing capacity for each computing resource. Constraints (3) translate the fact that we seek to avoid any damage to the target by stopping all the malicious flow before it reaches it. In Constraints (3), $\mathcal{N}(\tilde{f}) = \{n \in \mathcal{N} \setminus \{t\} | \sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a(n)} \tilde{f}^{a,p} > 0\}$ represents the subset of nodes n through which part of the malicious flow transits when considering the flow routing $\tilde{f}$. Constraints (3) impose that, for each possible routing $\tilde{f}$, the total filtering capacity installed on the nodes traversed by a strictly positive amount of malicious flow in the routing $\tilde{f}$, i.e on the nodes belonging to $\mathcal{N}(\tilde{f})$, is larger than the total malicious flow actually routed through the network in $\tilde{f}$. Constraints (4) forbid any filtering at the targeted node. Note that Constraints (3) are robust constraints that should hold for any flow routing belonging to the uncertainty set $\mathcal{U}$.

3.3 Small illustrative example

Before discussing some theoretical aspects relative to the formulation and complexity of Problem RVNFD, we provide a small illustrative example to facilitate the understanding of the proposed models and methods.

Let us consider a small network $\mathcal{G}$ including $|\mathcal{N}| = 5$ nodes and $|\mathcal{L}| = 5$ links, each one with a bandwidth of $b_l = 15$Mbsp. The malicious flow corresponding to the on-going DDoS attack enters the network at a gateway located at node 1 and targets a critical customer node located at node 5: we thus have $A = 1$, $s_1 = 1$ and $t = 5$. We consider $R = 1$ computing resource corresponding to the number of CPUs available at each node: we

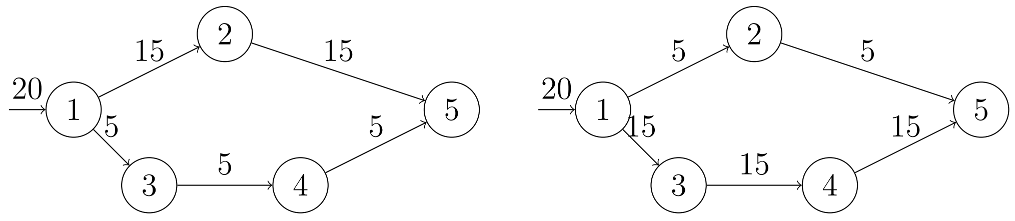

Figure 2: Small illustrative example: two possible routings for the malicious flow

have $Cap^1_n = 4$ CPUs available at node $n \in \{1, 2\}$ and $Cap^1_n = 2$ CPUs available at node $n \in \{3, 4\}$. There is a single type of filtering VNF, i.e. $V = 1$. Each instantiated VNF has a filtering capacity of $\phi_1 = 5$Mbps and requires $k^{1,1} = 2$ CPUs.

Figure 2 displays two possible routings of the malicious flow. This one may use $|\mathcal{P}^1| = 2$ paths from node 1 to reach its target: path $p = 1$ corresponds to $1 \to 2 \to 5$ and path $p = 2$ to $1 \to 3 \to 4 \to 5$. The routing displayed on the left correspond to $\tilde{f}_{left} = (\tilde{f}^{1,1}_{left}, \tilde{f}^{1,2}_{left}) = (15, 5)$, the routing displayed on the right to $\tilde{f}_{right} = (\tilde{f}^{1,1}_{right}, \tilde{f}^{1,2}_{right}) = (5, 15)$. $\tilde{f}_{left}$ and $\tilde{f}_{right}$ are two elements (in fact two extreme points) of the uncertainty set $\mathcal{U}$.

By solving Problem RVNFD for this small instance, we obtain the VNF placement shown in Figure 3. It consists in placing two VNFs at nodes 1 and 2 (i.e. $x^1_1 = x^1_2 = 2$) and one VNF at nodes 3 and 4 (i.e. $x^1_3 = x^1_4 = 1$.) Finally, Figure 4 presents how the malicious flow may be filtered by the instantiated VNFs in case it is routed according to $\tilde{f}_{left}$ (see the network on the left) or according to $\tilde{f}_{right}$ (see the network on the right). Note how, in both cases, all the malicious flow is filtered and stopped before it reaches the target located at node 5.

3.4 Discussion on the aggregated attack filtering constraints

Constraints (3) can be seen as aggregated attack filtering constraints ensuring that the total filtering capacity installed on the set of nodes traversed by $\tilde{f}$ is larger than the total malicious flow routed through the network. As such, they do not guarantee that the filtering capacity installed on each potential path $p \in \mathcal{P}^a$ of each attack a is enough to stop all the flow related to attack

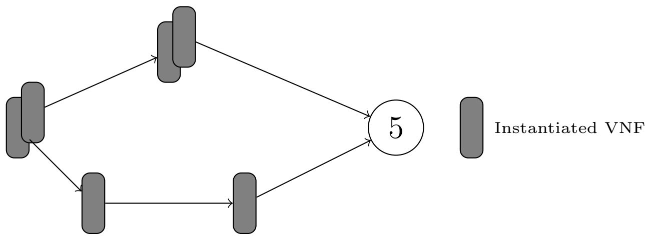

Figure 3: Small illustrative example: robust VNF placement

Figure 4: Small illustrative example: malicious flow filtering for two possible routings

a routed on this path, i.e. that the filtering capacity installed on each path $p \in \mathcal{P}^a$ is larger than $\tilde{f}^{a,p}$. However, we show in what follows that, for any feasible solution $\overline{x}$ of Problem RVNFD and any flow $\overline{f}$ belonging to $\mathcal{U}$, we can find at least one allocation of the filtering capacity installed at each node n to the flows going through n such that all the malicious traffic can be filtered. This can be done by solving the following transportation problem denoted by TP.

Let $d_n^{a,p}$ be the decision variable representing the amount of filtering capacity installed at node n allocated to stopping the malicious flow routed on path $p \in \mathcal{P}^a, a \in \mathcal{A}$.

$$Z^*_{TP} = min \sum_{n \in \mathcal{N}} \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a(n)} d_n^{a,p} \tag{6}$$

$$\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a(n)} d_n^{a,p} \leq \sum_{v \in \mathcal{V}} \phi^v \overline{x}_n^v \qquad \forall n \in \mathcal{N} \setminus \{t\} \tag{7}$$

$$\sum_{n \in \mathcal{N}^{a,p} \setminus \{t\}} d_n^{a,p} \geq \overline{f}^{a,p} \qquad \forall a \in \mathcal{A}, \forall p \in \mathcal{P}^a \tag{8}$$

$$d_n^{a,p} \geq 0 \qquad \forall n \in \mathcal{N}, \forall a \in \mathcal{A}, \forall p \in \mathcal{P}^a(n) \tag{9}$$

The objective (6) seeks to minimize the total amount of filtering capacity used to stop the malicious flow. Constraints (7) ensure that, at each node n, the total amount of filtering capacity allocated to stop the flow routed on each path $p \in \mathcal{P}^a$ of each attack a going through node n is not larger that the amount of filtering capacity available at node n. Constraints (8) guarantee that, for each attack a and each path $p \in \mathcal{P}^a$ used to route the attack in $\overline{f}$, the total amount of filtering capacity dedicated to p on the nodes belonging to it is large enough to stop all the malicious flow routed on p before it reaches t.

Proposition 1. *If $\overline{x}$ is a feasible solution of Problem RVNFD and $\overline{f}$ a flow belonging to the uncertainty $\mathcal{U}$, there exists at least one feasible solution for Problem TP, i.e. one allocation of the installed filtering capacity to the paths used by the attacks such that the total filtering capacity allocated to each path p of each attack a is larger than $\overline{f}^{a,p}$.*

Proof. The proof is done by contradiction.

Let assume that Problem TP is unfeasible. It means that there exists a subset of attacks $\mathcal{A}' \subset \mathcal{A}$ and a subset of paths $\mathcal{P}'^a \subset \mathcal{P}^a$ for each attack $a \in \mathcal{A}'$ such that $\sum_{n\in\mathcal{N}'}\sum_{v\in\mathcal{V}}\phi^v\overline{x}_n^v < \sum_{a\in\mathcal{A}'}\sum_{p\in\mathcal{P}'^a}\overline{f}^{a,p}$ where $\mathcal{N}' = \cup_{a\in\mathcal{A}',p\in\mathcal{P}'^a}\mathcal{N}^{a,p}$. In other words, it exists a subset of paths $\mathcal{P}'^a, a \in \mathcal{A}'$, such that the total filtering capacity installed on the nodes belonging to $\mathcal{N}'$ is insufficient to stop the flow going through these nodes.

Let us consider the routing f' defined by: $f'^{a,p} = \overline{f}^{a,p}$ if $a \in \mathcal{A}'$ and $p \in \mathcal{P}'^a$ and $f'^{a,p} = 0$ otherwise. We have $\mathcal{N}(f') = \mathcal{N}'$. As f' belongs to the uncertainty set $\mathcal{U}$ and $\overline{x}$ is a feasible solution of Problem RVNFD, the constraint $\sum_{n\in\mathcal{N}(f')}\sum_{v\in\mathcal{V}}\phi^v\overline{x}_n^v \geq \sum_{a\in\mathcal{A}'}\sum_{p\in\mathcal{P}'^a} f'^{a,p}$ should hold. This is in contradiction with the strict inequality written above.

□

In other words, solving Problem TP provides a VNF placement ensuring that all the malicious flow of the attack will be stopped provided we use an allocation of the filtering capacity to the paths actually used by the attack which complies with Constraints (7)-(9). Lemma 1 guarantees that such an allocation exists. However, solving Problem RVNFD does not guarantee that any allocation of the installed filtering capacity to the paths actually used by the attack will enable the ISP to block all the malicious flow.

3.5 Complexity analysis

Proposition 2. *Problem RVNFD is NP-hard, even if the uncertainty set $\mathcal{U}$ contains a finite and discrete set of potential routings.*

Proof. The proof is done by reduction from the minimum set covering problem.

Consider an instance I' of the minimum set covering problem. I' includes N potential location sites (indexed by $n = 1, ...N$) for the facilities and D demand points (indexed by $\delta_1, ..., \delta_D$). For each demand point δ_d, $d = 1, ..., D$, we define the subset of potential location sites, $\mathcal{N}(\delta_d) \subset \{1, ..., N\}$, which may cover it. The objective of the minimum set covering problem is to cover all demand points while minimizing the total number of opened facilities.

This instance of the minimum set covering problem can be transformed into an instance I of Problem RVNFD as follows. The graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ has $N + 1$ nodes and N links. The nodes indexed by $n = 1...N$ correspond to nodes where VNFs may be instantiated by the ISP and the node indexed by $N+1$ corresponds to the target of the attack: we thus have $\mathcal{N} = \{1, ..., N+1\}$.

There is a link $l \in \mathcal{L}$ between each node indexed by $n = 1...N$ and the node indexed by $N + 1$. Each link has a bandwidth equal to $b_l = 1$. The DDoS attack enters the network at $A = N$ ingress points corresponding to the nodes indexed by $n = 1...N$ (i.e. $s^a = a$ for $a = 1...N$) and targets node $N + 1$ (i.e. $t = N + 1$). We set $F^a = \frac{1}{A}$ for each attack a.

Each attack a may thus use a single path to reach the target: for each a in $\mathcal{A}$, $|\mathcal{P}^a| = 1$ and the path indexed by $(a, 1)$ corresponds to $a \to t$. A routing in the network is thus a vector $f_d = (f_d^{1,1}, ..., f_d^{a,1}, ..., f_d^{A,1})$ describing the flow of malicious traffic on the single path of each attack. For each demand point $\delta_d, d = 1...D$, of the minimum set covering problem, we add a routing f_d in the discrete uncertainty set $\mathcal{U}_D$ with $f_d^{a,p} = \frac{1}{A}$ if node a belongs to $\mathcal{N}(\delta_d)$ and $f_d^{a,p} = 0$ otherwise.

We consider a single computing resource ($R = 1$) with $Cap_n^1 = 1$ for each node n in $\{1, ..., N\}$. There is a single type of VNF indexed by $v = 1$ with a cost equal to $K^1 = 1$, a filtering capacity ϕ^1 equal to 1 and a consumption of the computing resource $k^{1,1}$ equal to 1. The total amount of malicious flow in any routing $f_d \in \mathcal{U}_D$, $\sum_{a \in \mathcal{A}} F^a$, is less than or equal to 1. Consequently, placing a VNF on any node belonging to $\mathcal{N}(f_d)$ suffices to ensure that the aggregated filtering constraints (3) will be satisfied for routing f_d.

Moreover, as the sets $\mathcal{N}(\delta_d)$ and $\mathcal{N}(f_d)$ coincide for each d, a demand point δ_d will be covered in instance I' as long as a VNF is instantiated on a node belonging to $\mathcal{N}(f_d)$ in instance I. As a consequence, determining, for instance I, the minimum cost VNF placement enabling to stop all the malicious flow, whatever its routing $f_d \in \mathcal{U}_D$, provides the minimum set of potential location sites covering all demand points in instance I'. Solving instance I' of Problem RVNFD thus provides a solution to instance I of the minimum set covering problem.

As the minimum set covering problem is known to be NP-hard (see e.g. Korte and Vygen (2012)), the results follows.

□

Figure 5 illustrates this reduction on a small instance I' of the minimum set cover problem with $N = 3$ potential location sites (represented as dashed nodes indexed from 1 to 3) and $D = 4$ demand points represented by the nodes denoted by δ_1 to δ_4. We have $\mathcal{N}(\delta_1) = \{1, 2\}$, $\mathcal{N}(\delta_2) = \{1, 2, 3\}$, $\mathcal{N}(\delta_3) = \{2\}$ and $\mathcal{N}(\delta_4) = \{3\}$. The corresponding graph is displayed at the top of Figure 5.

This instance of the minimum set covering problem can be transformed into an instance I of Problem RVNFD as follows. The corresponding graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ has $N + 1 = 4$ nodes and $N = 3$ links: see the bottom part of Figure 5. The DDoS attack enters the network at $A = 3$ ingress points corresponding to the nodes indexed by $n = 1...3$ (i.e. $s^1 = 1$, $s^2 = 2$ and $s^3 = 3$) and targets node $t = 4$. We set $F^a = \frac{1}{A} = 0.33$ for each attack a. A routing $f_d = (f_d^{1,1}, f_d^{2,1}, f_d^{3,1})$ describes the amount of malicious flow routed on the single path $a \rightarrow 4$ that may be used by each attack a to reach the target. For each demand point $\delta_d, d = 1...D$, of the minimum set covering problem, we add a routing f_d in the discrete uncertainty set $\mathcal{U}_D$ such that $f_d^{a,p} = \frac{1}{A}$ if node a belongs to $\mathcal{N}(\delta_d)$ and $f_d^{a,p} = 0$ otherwise. This gives $f_1 = (0.33, 0.33, 0)$, $f_2 = (0.33, 0.33, 0.33)$, $f_3 = (0, 0.33, 0)$ and $f_4 = (0, 0, 0.33)$. We thus have $\mathcal{N}(\delta_d) = \mathcal{N}(f_d)$ for each $d = 1...D$. We set $R = 1$, $Cap_n^1 = 1$ for $n = 1..3$, $V = 1$, $K^1 = 1$, $\phi_1 = 1$ and $k^{1,1} = 1$ as described in the proof of Proposition 2.

The optimal solution of instance I' consists in placing a VNF at nodes 2 and 3 as this suffices to ensure that the aggregated filtering constraints (3) will be respected for all routings in the discrete uncertainty set $\mathcal{U}_D$. This gives an optimal solution of instance I which consists in opening a facility at the potential sites 2 and 3.

4 Solution approach

As explained e.g. by Gorissen et al. (2015), Problem RVNFD may seem intractable as such as the number of constraints (3) is infinite. Two main ways have been proposed in the literature to handle this difficulty.

The first one consists in applying reformulation techniques which result in the formulation of a deterministic problem with a finite number of constraints: see e.g. Bertsimas and Sim (2004). In our case, the use of these reformulation techniques is not possible. Namely, the worst case reformation of Constraints (3) would lead to the following expression:

$$\min_{\tilde{f} \in \mathcal{U}} \sum_{n \in \mathcal{N}} \mathbb{I}\Big(\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a(n)} \tilde{f}^{a,p} > 0 \Big) \sum_{v \in \mathcal{V}} \phi^v x_n^v - \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a} \tilde{f}^{a,p} > 0 \qquad (10)$$

where $\mathbb{I}\Big(\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a(n)} \tilde{f}^{a,p} > 0 \Big)$ is an indicator function that is equal to one if $\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a(n)} \tilde{f}^{a,p} > 0$ and zero otherwise. The resulting inner minimization problem cannot be formulated as a linear program (but rather as a

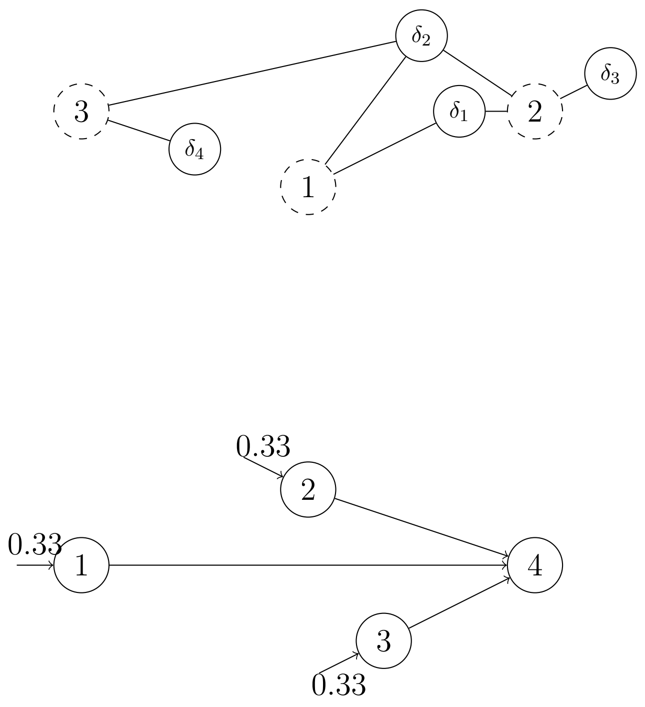

Figure 5: Reduction of an instance of the minimum set covering problem (top) into an instance of Problem RVNFD (bottom)

mixed-integer linear program) due to the presence of this indicator function. It is thus not possible to use the duality theory to reformulate it and obtain a computationally tractable robust counterpart as is commonly done in this type of reformulation approach.

The second possible way of solving a RO problem such as Problem RVNFD consists in applying an adversarial approach. Such approaches are based on the decomposition of the initial problem into a master problem and a sub-problem. The master problem, called the decision maker problem in this context, can be seen as a restricted version of the original RO problem in which only a finite number of extreme points $\mathcal{U}_R \subset \mathcal{U}$ of the uncertainty set (instead of the whole uncertainty set $\mathcal{U}$) are used to express the robust constraints. This problem is a deterministic optimization problem with a finite number of constraints and is thus computationally tractable. The sub-problem is called the adversarial problem. Given the solution provided by the decision maker problem, the adversarial problem seeks to find an extreme point of $\mathcal{U}$ for which this solution is infeasible. If no such extreme point can be found, the current solution of the decision maker problem is optimal for the initial RO problem. If such an extreme point is found, we add it to the restricted set $\mathcal{U}_R$ and reiterate the process. The finite convergence of this algorithm is ensured by the fact that the uncertainty set $\mathcal{U}$ has a finite number of extreme points. Adversarial approaches have been successfully used to solve RO problems arising in a variety of applications: see among others Bienstock and Özbay (2008), Attila et al. (2017), van Hulst et al. (2017) and Agra et al. (2018).

4.1 Adversarial approach

The proposed adversarial approach thus iteratively solves the decision maker problem and the adversarial sub-problem. At each iteration, the decision maker problem is solved using the current restricted uncertainty set $\mathcal{U}_R$ and provides a placement of the VNFs $\overline{x}$ which is optimal for this restricted uncertainty set. $\overline{x}$ being given, the adversarial problem is solved to find the worst-case routing of the malicious flow for the VNF placement described by $\overline{x}$, i.e. to find an extreme point of $\mathcal{U}$ wich maximises the infeasibility of $\overline{x}$ if it exists. In case such an extreme point is found, we update the restricted uncertainty set $\mathcal{U}_R$ by adding the newly found routing $\overline{f}$ and go on to the next iteration. Otherwise, $\overline{x}$ is feasible for all extreme points of $\mathcal{U}$, the current VNF placement $\overline{x}$ is optimal and the algorithm stops.

4.1.1 Decision maker sub-problem

The decision maker problem, denoted by $DMP(\mathcal{U}_R)$, can be formulated as follows:

$$Z^*_{DMP}(\mathcal{U}_R) = \min \sum_{v\in\mathcal{V}} \sum_{n\in\mathcal{N}} K^v x^v_n \tag{11}$$

$$\sum_{v\in\mathcal{V}} k^{rv} x^v_n \leq Cap^r_n \qquad \forall n \in \mathcal{N}, \forall r \in \mathcal{R} \tag{12}$$

$$\sum_{n\in\mathcal{N}(\tilde{f})} \sum_{v\in\mathcal{V}} \phi^v x^v_n \geq \sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a} \tilde{f}^{a,p} \qquad \forall \tilde{f} \in \mathcal{U}_R \tag{13}$$

$$x^v_t = 0 \qquad \forall v \in \mathcal{V} \tag{14}$$

$$x^v_n \text{ integer} \qquad \forall n \in \mathcal{N}, \forall v \in \mathcal{V} \tag{15}$$

Problem $DMP(\mathcal{U}_R)$ thus displays the same structure as the initial RO problem but the number of Constraints (13) is now finite. Moreover, as will be shown by the numerical experiments provided in Section 5, in practice, the cardinality of $\mathcal{U}_R$, and as a consequence the number of Constraints (13) involved in the formulation, remain rather limited when implementing the adversarial approach. Problem $DMP(\mathcal{U}_R)$ can thus be directly solved by a mixed-integer linear programming solver with a reasonable computational effort.

4.1.2 Adversarial sub-problem

Let us now focus on the adversarial sub-problem. In order to formulate it, we introduce the following decision variables:
- $f^{a,p}$: amount of malicious flow of attack a routed on path $p \in \mathcal{P}^a$,
- $z_n \in \{0, 1\}$: $z_n = 1$ if there is a positive amount of malicious flow transiting through node n, 0 otherwise.

Given the current VNF placement $\overline{x}$, the maximum amount of malicious flow which can reach its target can be found by solving the following mixed-integer linear program, denoted by $AP(\overline{x})$.

$$Z^*_{AP}(\overline{x}) = \max \sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a} f^{a,p} - \sum_{n\in\mathcal{N}\setminus\{t\}} (\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n) z_n \tag{16}$$

$$\sum_{p\in\mathcal{P}^a} f^{a,p} \leq F^a \qquad \forall a \in \mathcal{A} \tag{17}$$

$$\sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a \text{ s.t. } l\in\mathcal{L}^{a,p}} f^{a,p} \leq b_l \qquad \forall l \in \mathcal{L} \tag{18}$$

$$\sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a(n)} f^a_p \leq (\sum_{a\in\mathcal{A}} F^a) z_n \qquad \forall n \in \mathcal{N}\setminus\{t\} \tag{19}$$

$$f^{a,p} \geq 0 \qquad \forall p \in \mathcal{P}^a \tag{20}$$

$$z_n \in \{0,1\} \qquad \forall n \in \mathcal{N} \tag{21}$$

The linear variables f thus describe the worst-case routing of the malicious flow for the VNF placement $\overline{x}$. Constraints (17) ensure that, for each attack, the total amount of flow of attack a routed through the network is smaller that the total amount of flow of the attack F^a. Note that due to the limited bandwidth of the network links, it might not be possible to route all the flow of attack a through the network: Constraints (17) are thus formulated as inequalities rather than as equalities. Constraints (18) guarantee that the flow routed on each link does not exceed its bandwidth. In other words, Constraints (17), (18) and (20) make sure that the solution of problem $AP(\overline{x})$ provides a flow f belonging to the uncertainty set $\mathcal{U}$.

The objective function (16) seeks to maximize the amount of malicious flow which will reach its target, i.e. which will not be filtered by a VNF between its source and its target. Note that the filtering capacity $\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n$ placed at node n can stop part of the malicious flow only if there is a positive flow routed through node n, i.e. only if $\sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a(n)} f^a_p > 0$. $\sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a} f^a_p - \sum_{n\in\mathcal{N}\setminus\{t\}} (\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n) z_n$ thus computes the total amount of unfiltered flow as the difference between the total flow routed through the network, $\sum_{a\in\mathcal{A}} \sum_{p\in\mathcal{P}^a} f^a_p$, and the total amount of 'active' filtering capacity, $\sum_{n\in\mathcal{N}\setminus\{t\}} (\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n) z_n$. This 'active' filtering capacity is given by the sum of the filtering capacities installed at the nodes n through which a positive amount of malicious flow transits. Constraints (19) ensure that, for each node n, variable z_n is equal to 1 as soon as there is some positive amount of malicious flow which is routed through node n.

Note that, similar to what is done in Constraint (3) of the initial RO problem, in the objective function (16) of the adversarial sub-problem, the total amount of unfiltered flow is computed in an aggregate manner, i.e. by looking at the total routed flow and at the total active filtering capacity on all nodes of the network. In a feasible solution of problem $AP(\overline{x})$, this might lead to an underestimation of the malicious flow which will reach its target. Namely, we may have a subset of nodes $\mathcal{N}'$ such that the total flow routed through the nodes $n \in \mathcal{N}'$, $\sum_{a\in\mathcal{A}} \sum_{p\in\cup_{n\in\mathcal{N}'}\mathcal{P}^a(n)} f^a_p$, is smaller than the total filtering capacity placed on these nodes, $\sum_{n\in\mathcal{N}'}(\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n)$. In this case, the actual filtering taking place at some of the nodes $n \in \mathcal{N}'$ is not equal to $\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n$ but to a smaller value. More precisely, the total filtering taking place on the subset of nodes $\mathcal{N}'$ is equal to $\sum_{a\in\mathcal{A}} \sum_{p\in\cup_{n\in\mathcal{N}'}\mathcal{P}^a(n)} f^a_p$ rather than to $\sum_{n\in\mathcal{N}'}(\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n)$. This means that the objective function (16) overestimates the actual filtering taking place on the part of the network corresponding to $\mathcal{N}'$ and thus underestimates the amount of unfiltered malicious flow. However, we show in what follows that such a situation cannot occur in an optimal solution of $AP(\overline{x})$.

Proposition 3. *Any optimal solution of $AP(\overline{x})$ provides the worst-case routing for the given VNF placement $\overline{x}$.*

Proof. Let us consider a solution of $AP(\overline{x})$ in which there is at least one subset of nodes $\mathcal{N}'$ such that the total flow routed through the nodes $n \in \mathcal{N}'$, $\sum_{a\in\mathcal{A}} \sum_{p\in\cup_{n\in\mathcal{N}'}\mathcal{P}^a(n)} f^a_p$, is smaller than $\sum_{n\in\mathcal{N}'}(\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n)$. We show that this solution cannot be optimal for $AP(\overline{x})$.

It is namely possible to build another feasible solution of $AP(\overline{x})$ by setting to 0 the flow on all the paths belonging to $\cup_{n\in\mathcal{N}'}\mathcal{P}^a(n)$ and by setting z_n to 0 for all nodes $n \in \mathcal{N}'$. The objective value of the obtained solution will be increased by $\sum_{n\in\mathcal{N}'}(\sum_{v\in\mathcal{V}} \phi^v \overline{x}^v_n) - \sum_{a\in\mathcal{A}} \sum_{p\in\cup_{n\in\mathcal{N}'}\mathcal{P}^a(v)} f^a_p > 0$, i.e. will be strictly larger than the one of the initial solution. This latter can therefore not be optimal. □

4.2 Resolution of the adversarial sub-problem

The adversarial sub-problem $AP(\overline{x})$ is a mixed-integer linear program which could theoretically be solved directly by a mathematical programming solver. However, the number of paths that could possibly be used to route the malicious flow of a given attack a between its source s^a and the target t, and as a consequence the number of flow variables $f^{a,p}$, grows exponentially fast with the network size.

This difficulty may be overcome by using a column generation technique. In a column generation algorithm, we start solving problem $AP(\overline{x})$ with a restricted number of flow variables (i.e. of columns), which provides an initial feasible solution. This initial solution is then improved by iteratively adding new flow variables (i.e. by generating new columns) to the formulation of the problem until no more improving flow variables can be found.

Let $RAP(\overline{x}, \mathcal{C})$ be a restricted version of problem $AP(\overline{x})$ in which only a subset of the flow variables $f^{a,p}$ are explicitly considered. Here, $\mathcal{C}$ denotes a collection of subsets of paths. More precisely, we have $\mathcal{C} = \{\mathcal{P}^a_R, a \in \mathcal{A}\}$ where $\mathcal{P}^a_R \subset \mathcal{P}^a$ is the restricted subset of potential paths available for attack a taken into account in the problem formulation.

$RAP(\overline{x}, \mathcal{C})$ can be formulated as follows:

$$Z^*_{RAP}(\overline{x}, \mathcal{C}) = \max \sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a_R} f^a_p - \sum_{n \in \mathcal{N} \setminus \{t\}} (\sum_{v \in \mathcal{V}} \phi^v \overline{x}^v_n) z_n \tag{22}$$

$$\sum_{p \in \mathcal{P}^a_R} f^a_p \leq F^a \qquad \forall a \in \mathcal{A} \tag{23}$$

$$\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a_R \text{ s.t. } l \in \mathcal{L}^{a,p}} f^a_p \leq b_l \qquad \forall l \in \mathcal{L} \tag{24}$$

$$\sum_{a \in \mathcal{A}} \sum_{p \in \mathcal{P}^a_R(n)} f^a_p \leq (\sum_{a \in \mathcal{A}} F^a) z_n \qquad \forall n \in \mathcal{N} \setminus \{t\} \tag{25}$$

$$f^a_p \geq 0 \qquad \forall p \in \mathcal{P}^a_R \tag{26}$$

$$z_n \in \{0, 1\} \qquad \forall n \in \mathcal{N} \tag{27}$$

Note that $RAP(\overline{x}, \mathcal{C})$ displays the same structure as $AP(\overline{x})$ but the objective and constraints are expressed using a limited number of flow variables $f^{a,p}$, namely those corresponding to paths belonging to the restricted subset $\mathcal{P}^a_R$, for each attack $a \in \mathcal{A}$.

The column generation process, i.e. the process of adding new flow variables $f^{a,p}$, relies on the linear relaxation, denoted by $\underline{RAP}(\overline{x}, \mathcal{C})$, of $RAP(\overline{x}, \mathcal{C})$.

More precisely, at each iteration of the column generation algorithm, in order to identify improving flow variables to be added to the formulation, we first solve $\underline{RAP}(\overline{x}, \mathcal{C})$ with the current collection of path subsets $\mathcal{C}$. We then solve the pricing problem for each attack $a \in \mathcal{A}$. It consists in finding a flow variable $f^{a,p}$ with a positive reduced cost, i.e. a flow variable whose inclusion in the linear programming formulation might lead to an improvement of the objective function, or determining that no such variable exists. If at least one improving flow variable is found, we carry on with a new iteration of the algorithm. If no such variable is found, it means that the current solution of $\underline{RAP}(\overline{x}, \mathcal{C})$ is an optimal solution of the $\underline{AP}(\overline{x})$, the linear relaxation of $AP(\overline{x})$, and we stop.

Let α_a be the dual value of Constraint (23) relative to attack a, β_l the dual value of Constraint (24) relative to link l and γ_n the dual value of Constraint (25) relative to node n in the optimal solution of $\underline{RAP}(\overline{x}, \mathcal{C})$. The reduced cost of variable $f^{a,p}$ is given by $rc^{a,p} = 1 - (\alpha_a + \sum_{l \in \mathcal{L}^{a,p}} \beta_l + \sum_{n \in \mathcal{N}^{a,p}} \gamma_n)$.

Given an attack $a \in \mathcal{A}$, solving the pricing problem, i.e. identifying the variable $f^{a,p}$ with the largest reduced cost, thus amounts to finding the path $p \in \mathcal{P}^a$ with the smallest value of $\sum_{l \in \mathcal{L}^{a,p}} \beta_l + \sum_{n \in \mathcal{N}^{a,p}} \gamma_n$. This can be done by looking for the shortest path between s^a and t in the weighted digraph $(\mathcal{N}, \mathcal{L}, w)$ in which each link l has a weight of $w_l = \beta_l + \gamma_{dest(l)}$ where $dest(l)$ is the destination node of link l. This shortest path problem can be solved in polynomial time by Dijskra's algorithm. If a variable $f^{a,p}$ with a positive reduced cost is found, the corresponding path is added to $\mathcal{P}_R^a$ and the collection $\mathcal{C}$ is updated accordingly.

Algorithm 1 provides a formal description of the column generation algorithm used to solve $\underline{AP}(\overline{x})$.

Note that Algorithm 1 solves to optimality the linear relaxation of $AP(\overline{x})$. In order to solve the original adversarial sub-problem $AP(\overline{x})$, which is a mixed-integer linear program, we consider two alternative ways of using it.

The first one corresponds to an exact Branch & Price algorithm. Basically, a Branch & Price algorithm is a Branch & Bound method in which, at each node of the search tree, new variables may be added to the linear programming relaxation. More precisely, the Branch & Price algorithm starts solving the restricted version of the adversarial sub-problem $RAP(\overline{x}, \mathcal{C})$ with an initial collection of path subsets $\mathcal{C}$, using a branch-and-bound method. At each node of the Branch & Bound search tree, we use Algorithm 1 to solve $\underline{AP}(\overline{x})$ and add new columns in the formulation (i.e. new paths in $\mathcal{C}$). When no new column can be generated by Algorithm 1, i.e. when the linear relaxation of the restricted master problem has been solved to optimality at the current Branch & Bound node, we either get an integer feasible solution of the initial problem $AP(\overline{x})$ or we branch on a fractional variable z_n to create new nodes in the search tree and continue with the Branch & Bound

```
input  : A VNF placement $\overline{x}$ and a collection of path subsets $\mathcal{C}$
output: An updated collection of path subsets $\mathcal{C}$
begin
    repeat
        stop ← 0
        solve $\underline{RAP}(\overline{x}, \mathcal{C})$ with a linear programming solver
        get the dual values $(\alpha, \beta, \gamma)$ of Constraints (23)-(25)
        for $l$=1 to $L$ do
            $w_l \leftarrow \beta_l + \gamma_{dest(l)}$
        end
        for $a$=1 to $A$ do
            find the shortest path $p_s$ between $s^a$ and $t$ in $(\mathcal{N}, \mathcal{L}, w)$
            $rc^a_{p_s} \leftarrow 1 - (\alpha_a + \sum_{l \in \mathcal{L}^a_{p_s}} \beta_l + \sum_{n \in \mathcal{N}^a_{p_s}} \gamma_n)$
            if $rc^a_{p_s} > 0$ then
                stop ← 1
                $\mathcal{P}^a_R \leftarrow \mathcal{P}^a_R \cup \{p_s\}$
            end
        end
    until stop = 0;
end
```

Algorithm 1: Column generation algorithm solving $\underline{AP}(\overline{x})$ to optimality

algorithm. The algorithm stops when there are no more open nodes in the search tree.

The second one is a heuristic algorithm. In this case, we first solve $\underline{AP}(\overline{x})$ using Algorithm 1. When Algorithm 1 stops, we get the updated collection of path subsets $\mathcal{C}$, reintroduce the integrality constraints on variables $z_n, n \in \mathcal{N}$, and solve the restricted problem $RAP(\overline{x}, \mathcal{C})$ as a mixed-integer linear program. Note that this algorithm may provide a sub-optimal solution of $AP(\overline{x})$ as the collection of path subsets $\mathcal{C}$ obtained by solving $\underline{AP}(\overline{x})$ may not be the same as the one needed to obtain an optimal solution of $AP(\overline{x})$.

4.3 Summary of the proposed solution approach

The overall proposed solution approach is described by Algorithm 2 for the case where the adversarial sub-problem is solved exactly and Algorithm 3 for the case where the adversarial sub-problem is solved heuristically.

In Algorithms 2 and 3, the restricted uncertainty set $\mathcal{U}_R$ is initialized as an empty set whereas the restricted path subset $\mathcal{P}_R^a$ to be used for each attack $a \in \mathcal{A}$ initially contains a single path, namely the shortest path in terms of hops between the source of attack a and the target. Moreover, note that the subsets $\mathcal{P}_R^a, a \in \mathcal{A}$, are not reinitialized at each iteration of the adversarial algorithm. This means that the improving paths found while solving $AP(\overline{x}^i)$, where $\overline{x}^i$ denotes the solution of $DMP(\mathcal{U}_R)$ found at iteration i of the adversarial algorithm, are part of the initial collection of path subsets provided to Algorithm 1 when it will be used to solve $AP(\overline{x}^j)$, where $\overline{x}^j$ denotes the solution of $DMP(\mathcal{U}_R)$ found at any iteration $j > i$ of the adversarial algorithm. Our preliminary numerical experiments namely showed that this was more computationally efficient than reinitializing the subsets $\mathcal{P}_R^a, a \in \mathcal{A}$, at each iteration of the adversarial algorithm.

```
begin
    $\mathcal{U}_R \leftarrow \emptyset$
    build the weighted digraph $\mathcal{G} = (\mathcal{N}, \mathcal{L}, w)$ with $w_l = 1, \forall l \in \mathcal{L}$
    for $a$=1 to $A$ do
        find the shortest path $p_s$ between $s^a$ and $t$ in $\mathcal{G}$
        $\mathcal{P}_R^a \leftarrow \{p_s\}$
    end
    $\mathcal{C} \leftarrow \{\mathcal{P}_R^a, a \in \mathcal{A}\}$
    repeat
        solve $DMP(\mathcal{U}_R)$ and record the current VNF placement $\overline{x}$
        solve $RAP(\overline{x}, \mathcal{C})$ with a Branch & Price algorithm using
          Algorithm 1 to generate new columns at each node of the
          search tree and record the updated collection of path subsets
          $\mathcal{C}$
        if $Z^*_{RAP}(\overline{x}, \mathcal{C}) > 0$ then
            record the optimal flow routing $\overline{f}$
            $\mathcal{U}_R \leftarrow \mathcal{U}_R \cup \{\overline{f}\}$
        end
    until $Z^*_{RAP}(\overline{x}, \mathcal{C}) \leq 0$;
end
```
Algorithm 2: Solution algorithm with an exact solution of the adversarial sub-problem

```
begin
    $\mathcal{U}_R \leftarrow \emptyset$
    build the weighted digraph $\mathcal{G} = (\mathcal{N}, \mathcal{L}, w)$ with $w_l = 1, \forall l \in \mathcal{L}$
    for $a$=1 to $A$ do
        find the shortest path $p_s$ between $s^a$ and $t$ in $\mathcal{G}$
        $\mathcal{P}_R^a \leftarrow \{p_s\}$
    end
    $\mathcal{C} \leftarrow \{\mathcal{P}_R^a, a \in \mathcal{A}\}$
    repeat
        solve $DMP(\mathcal{U}_R)$ and record the current VNF placement $\overline{x}$
        solve $\underline{RAP}(\overline{x}, \mathcal{C})$ using Algorithm 1 and record the updated
          collection of path subsets $\mathcal{C}$
        solve $RAP(\overline{x}, \mathcal{C})$ as a mixed-integer linear program
        if $Z^*_{RAP}(\overline{x}, \mathcal{C}) > 0$ then
            record the optimal flow routing $\overline{f}$
            $\mathcal{U}_R \leftarrow \mathcal{U}_R \cup \{\overline{f}\}$
        end
    until $Z^*_{RAP}(\overline{x}, \mathcal{C}) \leq 0$;
end
```

Algorithm 3: Solution algorithm with a heuristic solution of the adversarial sub-problem

5 Numerical results

5.1 Instances

We randomly generated a set of medium-size instances of the problem following the indications provided by public data released by different cloud and telecom providers.

Network. We used 4 internet network topologies. The first three ones correspond to three internet networks described in the Internet Topology Zoo library, IntelliFiber ($N = 73$, $L = 96$), Colt Telecom ($N = 153$, $L = 179$) and Cogentco ($N = 197$, $L = 245$): see Knight et al. (2011) and Knight et al. (2013) for more detail. We also used a topology corresponding to the former network of the French company Free ($V = 120$, $E = 167$): see Ferre (2010). Recall that the problem under study arises within the general con-

Topology	N	L	%Low	%Medium	%High
IntelliFiber	73	96	62%	37%	1%
Colt Telecom	153	179	72%	24%	4%
Cogentco	197	245	59%	39%	2%
Free	120	167	66%	28%	6%

Table 3: Percentage of nodes assigned to a low, medium or high computing capacity for each network topology

text of 5G network slicing. As a consequence, we do not consider in our problem the whole physical network installed by the ISP but only the portion of this network, i.e. the virtual network or slice, lent by the ISP to the customer currently undergoing a DDoS attack. Thus, the bandwidth b_l of a link between two nodes does not correspond to the total bandwidth of the physical link installed by the ISP between these nodes but only to the portion of this bandwidth allocated to the virtual network assigned to the customer under attack. This is why we randomly generated values of b_l corresponding to rather small transmission capacities. More precisely, the bandwidth b_l of each link was randomly generated using a discrete distribution with a support equal to $\{4.8, 12, 20, 40, 100\}$ Mbps.

Computing resources. $R = 2$ types of computing resources were taken into account at each node: the number of CPUs and the memory. We considered three types of nodes: low computing capacity with $Cap = (8, 32)$, medium computing capacity with $Cap = (40, 160)$ and high computing capacity with $Cap = (400, 1600)$. In each considered network topology, we assign each node to a type according to its degree. Thus, nodes with a degree less than 2 were assigned a low computing capacity, nodes with a degree between 3 and 5 were assigned a medium computing capacity and nodes with a degree larger than 6 were assigned a high computing capacity. Table 3 provides a summary of the percentage of nodes assigned to each type (low, medium and high computing capacity) for each considered network topology.

VNFs. $V = 1$ type of VNFs was considered requiring $\gamma^{1,1} = 4$ CPUs and $\gamma^{1,2} = 16$ units of memory, providing a filtering capacity of $\phi^n = 16$ Mbps, with a unit cost of $K^1 = 130$.

Attacks. The number of sources was set to $A \in \{5, 10, 15, 20, 30, 40\}$. In each instance, the sources and target of the attack were randomly selected. The intensity F^a of each attack (in Mbps) was randomly generated following the normal distribution $\mathcal{N}(50, 25)$.

For each considered network topology and value of A, we randomly generated 5 instances, leading to a total of 140 instances.

5.2 Results

Each generated instance was solved using Algorithms 2 and 3. In both cases, the decision maker problem was solved as a mixed-integer linear program using the CPLEX 12.8.9 solver with the default settings. The adversarial sub-problem was solved using either the Branch & Price algorithm embedded in the SCIP 7.0.0 solver (Algorithm 2) or the simplex and Branch & Cut algorithms embedded in the CPLEX 12.8.9 solver (Algorithm 3). All tests were carried out on an PC running under Windows 10 equipped with an Intel Core i5-8350U processor (4 cores, frequency of 1.9GHz) and a 16 GB RAM with a 2400MHz speed. Note that the CPLEX 12.8.9 solver, in its default settings, is set to use a number of threads equal to the number of available cores whereas the SCIP 7.0.0 solver is by default single-threaded.

For each algorithm, each network topology and each considered value of A, we report in Table 4 the average value over the 5 corresponding instances of:

- $Cost$: the cost of the optimal VNF placement,
- $\#IT$: the average number of iterations of the algorithm,
- $\#P$: the total number of source-target paths added to $\mathcal{C}$ by column generation over the course of the algorithm,
- $Time$: the total computation time in seconds of the algorithm.

Algorithm 3 is an approximate solution algorithm which may provide a solution which is not feasible for the initial robust optimization problem, i.e. for Problem RVNFD. Indeed, as the adversarial sub-problem is solved heuristically, the amount of malicious flow that will reach the target may be underestimated in some cases so that the filtering constraints (13) added to the decision maker problem may not be tight enough. In order to estimate the impact of this heuristic resolution, we carry out the following post-optimization analysis. We consider the optimal VNF placement $\overline{x}_{app}$ obtained with Algorithm 3. We solve problem $AP(\overline{x}_{app})$ exactly using the Branch & Price algorithm. We then record AD the actual damage, i.e. the amount of malicious flow which will actually reach its target, if we use the VNF placement $\overline{x}_{app}$. We then compute the percentage of total unfiltered flow $\%UF$ as $\%UF = \frac{100AD}{\sum_{a\in\mathcal{A}} F^a}$. We report in Table 4, for each set of 5 instances, $\#Inf$ the number of instances for which the solution obtained with Algorithm 3 was infeasible and $Max\%UF$ the maximum percentage of unfiltered flow.

Results from Table 4 first show that Algorithm 2 is able to provide optimal solutions to the RO problem with a reasonable computational effort. Namely, the average computation time, over the 140 considered instances, is 22s. This performance is mainly explained by the fact that both the number of iterations $\#IT$ of the algorithm (and as a consequence the number of Constraints (13) of $DMP(\mathcal{U}_R)$) and the number of source-target paths $\#P$ generated by column generation (and as a consequence the number of flow variables in $AP(\overline{x})$) stay limited.

However, the computation time of Algorithm 2 exceeds 60s for 10 out of the 140 considered instances. This might be a problem as the decisions on the VNFs deployment should be taken as quickly as possible after the attack detection and identification. The approximate Algorithm 3 might prove useful in such cases. It is namely able to provide optimal solutions of the RO problem for 137 out of the 140 considered instances, and this with an average computation time below 3s and a maximum computation time of 25s. Moreover, for the 3 instances for which the solution provided by Algorithm 3 did not comply with the original robust constraints (3), the amount of malicious flow which could reach the target stays below 3%, which seems acceptable.

6 Conclusion

This paper described a new robust optimization approach for the defense against Distributed Denial of Service (DDoS) attacks in the context of 5G network slicing. More precisely, we considered the problem of optimally deploying virtual network functions in order to stop an ongoing DDOS attack. We assumed that the target, sources and volume of the attack are identified but that the exact routing of the illegitimate traffic on the network is not known. To take into account these uncertainties, we proposed a robust optimization (RO) model and developed an adversarial approach to solve it. This iterative approach is based on the decomposition of the initial problem into

		Algorithm 2				Algorithm 3					
Topology	A	$Cost$	$\#IT$	$\#P$	$Time$	$Cost$	$\#IT$	$\#P$	$Time$(s)	$\#Inf$	$Max\%UF$
IntelliFiber	5	936	9	25	3	936	10	25	1	0	0.00%
	10	1066	13	36	10	1066	11	34	1	0	0.00%
	15	1248	11	45	4	1248	6	36	1	0	0.00%
	20	988	9	35	11	988	8	35	0	0	0.00%
	30	1846	31	130	134	1846	17	107	2	0	0.00%
	40	1950	17	130	31	1950	15	109	1	0	0.00%
Free	5	1092	12	27	10	1092	11	21	2	0	0.00%
	10	1326	10	41	4	1326	11	37	2	0	0.00%
	15	1378	5	60	3	1378	6	58	1	0	0.00%
	20	650	5	28	1	650	4	28	1	0	0.00%
	30	1092	8	71	3	1092	5	71	1	0	0.00%
	40	1352	7	63	4	1352	6	69	1	0	0.00%
Colt	5	1118	17	42	12	1118	19	40	5	1	1,16%
	10	806	7	35	2	806	7	41	2	0	0.00%
	15	1170	20	63	38	1170	19	59	5	0	0.00%
	20	1092	10	63	14	1092	11	59	3	0	0.00%
	30	754	7	71	4	754	4	70	2	0	0.00%
	40	1118	9	66	9	1118	6	40	2	0	0.00%
Cogentco	5	546	13	58	9	546	14	45	5	1	2,88%
	10	754	14	70	24	754	14	50	6	0	0.00%
	15	1222	18	95	20	1222	14	73	5	0	0.00%
	20	910	21	81	47	910	14	75	6	0	0.00%
	30	728	13	84	34	728	9	78	4	0	0.00%
	40	1066	20	101	107	1066	13	87	7	1	0,18%

Table 4: Numerical results

a master problem and a sub-problem. The master problem is a restricted version of the original RO problem in which only a finite number of possible malicious flow routings are used to express the robust constraints. Considering the current placement of VNF provided by the solution of the master problem, the adversarial sub-problem seeks to find a malicious flow routing that maximizes the amount of attack reaching its target. We tested the efficiency of our algorithms on medium-sized randomly generated instances. The results of computation experiments show that our approach is able of providing optimal solutions in short computation times.

Current work suggests several possible directions for future research. In terms of problem solving, it might be possible to further improve the decomposition approach by carrying out a polyhedral study of the problem and developing new valid inequalities to help solving it more quickly. As for the problem modeling, a first research direction could consist in studying a disaggregated formulation of the robust filter constraints. This could ensure that the instantiated VNFs will be able to stop all the malicious flows regardless of the allocation of filtering capacities. It would also be interesting to study how the legitimate traffic, which will consume network resources and whose routing is also unknown, could be taken into account in the model.

Acknowledgement

The authors would like to thank Kahina Lazri and Paul Chaignon (Orange Labs Products & Services) for their highly appreciated help in understanding and modeling the optimization problem. We are also grateful to Claudia D'Ambrosio (Laboratoire d'Informatique de l'École Polytechnique, CNRS, France) and Andrea Lodi (Ecole Polytechnique Montréal, Canada) for their fruitful advice on the problem modeling and solving.

References

Agra, A., Christiansen, M., Hvattum, L. M., and Rodrigues, F. (2018). Robust optimization for a maritime inventory routing problem. *Transportation Science*, 52(3):509–525.

Akpakwu, G. A., Silva, B. J., Hancke, G. P., and Abu-Mahfouz, A. M. (2018). A survey on 5G networks for the Internet of Things: Communication technologies and challenges. *IEEE Access*, 6:3619–3647.

Alharbi, T. and Aljuhani, A. (2017). Holistic DDoS mitigation using NFV. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference CCWC*.

Altner, D. S., Ergun, Ö., and Uhan, N. A. (2010). The maximum flow network interdiction problem: valid inequalities, integrality gaps, and approximability. *Operations Research Letters*, 38(1):33–38.

Apt, K. R. (2003). *Principles of Constraint Programming*. Cambridge University Press.

Attila, Ö. N., Agra, A., Akartunalı, K., and Arulselvan, A. (2017). A decomposition algorithm for robust lot sizing problem with remanufacturing option. In Gervasi, O., Murgante, B., Misra, S., Borruso, G., Torre, C. M., Rocha, A. M. A., Taniar, D., Apduhan, B. O., Stankova, E., and Cuzzocrea, A., editors, *Computational Science and Its Applications – ICCSA 2017*, pages 684–695. Springer International Publishing.

AWS (2020). AWS Shield - Threat landscape report Q1 2020. https://aws-shield-tlr.s3.amazonaws.com/2020-Q1_AWS_Shield_TLR.pdf. Accessed 2020-12-19.

Baffier, J.-F., Poirion, P.-L., and Suppakitpaisarn, V. (2018). Bilevel model for adaptive network flow problem. *Electronic Notes in Discrete Mathematics*, 64:105–114. 8^{th} International Network Optimization Conference - INOC 2017.

Berard, D. (2018). DDoS breach costs rise to over $2M for enterprises finds kaspersky lab report. https://usa.kaspersky.com/about/press-releases/2018_ddos-breach-costs-rise-to-over-2m-for-enterprises-finds-kaspersky-lab-report. Accessed 2020-12-19.

Bertsimas, D. and Sim, M. (2004). The price of robustness. *Operations Research*, 52(1):35–53.

Bienstock, D. and Özbay, N. (2008). Computing robust basestock levels. *Discrete Optimization*, 5(2):389 – 414.

Church, R. L., Scaparra, M. P., and Middleton, R. S. (2004). Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3):491–502.

Demirci, S. and Sagiroglu, S. (2019). Optimal placement of virtual network functions in software defined networks: A survey. *Journal of Network and Computer Applications*, 147:102424.

Donovan, J. (2014). How SDN enabled innovations will impact AT&T's plans to transform it's infrastructure. www.bit.ly/1RQFMko. Accessed 2020-10-01.

Fayaz, S. K., Tobioka, Y., Sekar, V., and Bailey, M. (2015). Bohatei: Flexible and elastic DDoS defense. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 817–832.

FBI (2020). Cyber actors exploiting built-in network protocols to carry out larger, more destructive distributed denial of service attacks. https://dd80b675424c132b90b3-e48385e382d2e5d17821a5e1d8e4c86b.ssl.cf1.rackcdn.com/external/fbi-private-industry-notification-20200721-002.pdf. Accessed 2020-12-19.

Ferre, L. (2010). Free SAS domestic network. https://fr.wikipedia.org/wiki/Free_(entreprise). Accessed 2020-12-19.

Fu, X. and Modiano, E. (2019). Network interdiction using adversarial traffic flows. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1765–1773. IEEE.

Fung, C. J. and McCormick, B. (2015). Vguard: A distributed denial of service attack mitigation method using network function virtualization. In *2015 11th International Conference on Network and Service Management (CNSM)*, pages 64–70.

Fysarakis, K., Askoxylakis, I., Manifavas, C., Soultatos, O., Papaefstathiou, I., and Katos, V. (2016). Which IoT protocol? comparing standardized approaches over a common M2M application. In *2016 IEEE Global Communications Conference (Globecom)*.

Gorissen, B. L., Yanikoglu, I., and den Hertog, D. (2015). A practical guide to robust optimization. *Omega*, 53:24 – 137.

Grawe, K. (2020). Link11 h1 2020 DDoS report reveals a resurgence in DDoS attacks during COVID-19 lockdowns. https://www.link11.com/en/blog/threat-landscape/h1-2020-link11-ddos-report-en/. Accessed 2020-12-19.

Guo, Q., An, B., Zick, Y., and Miao, C. (2016). Optimal interdiction of illegal network flow.

Jakaria, A. H. M., Rahman, M. A., and Fung, C. (2019). A requirement-oriented design of NFV topology by formal synthesis. *IEEE Transactions on Network and Service Management*, 16(4):1739–1753.

Jakaria, A. H. M., Yang, W., Rashidi, B., Fung, C., and Rahman, M. A. (2016). Vfence: A defense against distributed denial of service attacks using network function virtualization. In *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 431–436.

Khandelwal, S. (2016). 602 Gbps! this may have been the largest DDoS attack in history. www.thehackernews.com/2016/01/biggest-ddos-attack.html. Accessed 2020-12-19.

Knight, S., Nguyen, H. X., Falkner, N., Bowden, R., and Roughan, M. (2011). The internet topology zoo. *IEEE Journal on Selected Areas in Communications*, 29(9):1765–1775.

Knight, S., Nguyen, H. X., Falkner, N., Bowden, R., and Roughan, M. (2013). The internet topology zoo. http://www.topology-zoo.org/index.html. Accessed 2020-12-19.

Korte, B. and Vygen, J. (2012). *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag.

Lei, X., Shen, S., and Song, Y. (2018). Stochastic maximum flow interdiction problems under heterogeneous risk preferences. *Computers and Operations Research*, 90:97–109.

Li, X. and Qian, C. (2016). A survey of network function placement. In *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 948–953.

Lim, C. and Smith, J. C. (2007). Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26.

Naoum-Sawaya, J. and Ghaddar, B. (2017). Cutting plane approach for the maximum flow interdiction problem. *Journal of the Operational Research Society*, 68(12):1553–1569.

Netscout Systems (2020). Netscout threat intelligence report. https://www.netscout.com/sites/default/files/2020-02/SECR_001_EN-2001_Web.pdf. Accessed 2020-12-19.

Phillips, C. A. (1993). The network inhibition problem. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '93, pages 776–785, New York, NY, USA. Association for Computing Machinery.

Rahimi, H., Zibaeenejad, A., and Safavi, A. A. (2018). A novel IoT architecture based on 5G-IoT and next generation technologies. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 81–88.

Rashidi, B., Fung, C., and Rahman, M. (2018). A scalable and flexible DDoS mitigation system using network function virtualization. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6.

Savi, J. (2018). With OPNFV, Orange plans a full-scale rollout of network functions virtualization. https://thenewstack.io/orange-relies-opnfv-transform-networks-future/. Accessed 2020-12-19.

Silva, F. S. D., Silva, E., Neto, E. P., Lemos, M., Neto, A. J. V., and Esposito, F. (2020). A taxonomy of DDoS attack mitigation approaches featured by SDN technologies in IoT scenarios. *Sensors*, 20:3078.

van Hulst, D., den Hertog, D., and Nuijten, W. (2017). Robust shift generation in workforce planning. *Computational Management Science*, 14:115–134.

Vyakaranam, N. and Krishna, D. (2018). 5G: Network as a service - how 5G enables the telecom operators to lease out their network.

https://netmanias.com/en/post/blog/13311/5g/5g-network-as-a-service-how-5g-enables-the-telecom-operators-to-lease-out-their-network. Accessed 2020-12-19.

Wollmer, R. (1964). Removing arcs from a network. *Operations Research*, 12(6):934–940.

Wood, R. K. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18.