Optimal deployment of security virtual functions in Software-Defined Networks (SDN)

Sonia Haddad-Vanier[1], Celine Gicquel[2], Alexandros Papadimitriou[3]

[1] SAMM, Université Paris I Panthéon Sorbonne, France
`sonia.vanier@univ-paris1.fr`

[2] LRI, CNRS - Université Paris Saclay, France
`gicquel@lri.fr`

[3] Orange Labs Products & Services, France
`alexandros.papadimitriou@orange.com`

Abstract

We study the optimal deployment of security network virtual functions to counter distributed denial of service attacks in telecommunication networks. We propose a mathematical programming formulation for this combinatorial optimization problem and develop a decomposition approach to solve it. Preliminary computational results carried out on medium-size randomly generated instances are presented.

Keywords : *telecommunication network, facility location, mathematical programming.*

1 Problem description and mathematical modeling

Telecommunication companies are a major target for cyber-attacks because they operate critical infrastructures that are widely used to communicate and store large amounts of sensitive data. Among the most devastating attacks are the distributed denials of service (DDoS). A distributed denial of service is a type of security attacks in which multiple compromised computer systems attack a target, such as a server or a website, and cause a denial of service for users of the targeted resource. The monumental growth of multimedia content, the explosion of cloud computing, the increasing use of mobile phones contribute to reinforcing the frequency, variability and dangerousness of DDoS attacks.

However, the emergence of new networking technologies such as Software-Defined Network (SDN) and Network Function Virtualization (NFV) provide a new opportunity to design innovative DDoS mitigation solutions [1]. Namely, network services such as security mechanisms can now be deployed as virtualized network functions (VNF). Their flexibility and reactivity allow to postpone the determination of the DDoS defense architecture to be used after the attack is detected, its target identified and its volume estimated. The defense mechanisms can thus be placed and sized based on detailed information on the on-going attack. However, instantiating VNFs is costly and consumes computing resources in the network: it is thus necessary to carefully select their location and numbers. The purpose of this work is to develop an optimization framework capable of optimally deploying a defense architecture based on VNFs in order to mitigate the effect of an on-going DDoS attack and prevent the malicious flow from reaching its target.

The network topology is modeled by a digraph $G = (V, E)$ in which V, the set of nodes, represents specific equipment in the network and E, 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 of each link e, denoted by b_e.

The illegitimate traffic corresponding to the DDoS attack is represented as a set A of attacks: attack $a \in A$ corresponds to an illegitimate traffic of ψ^a Mbps between a source $s^a \in V$ and a target $t^a \in V$. The main difficulty here is that, as the traffic routing is done dynamically by algorithms controlled not by the internet service provider (ISP) but by the service providers, the routing of the malicious flow in the network is not known by the ISP when he has to make the VNF placement decisions. Let P^a be the set of all potential paths between s^a and t^a for attack a.

VNFs are used to stop the illegitimate traffic before it reaches its target. A VNF will be instantiated on a node $v \in V$ of the network and will filter the malicious flow. There are N types of VNFs available. A VNF of type n is characterized by its filtering capacity ϕ^n, its cost K^n and its computing resources consumption. R types of computing resources (CPU, memory, etc.) are considered. The amount of computing resource r required by the instantiation of one VNF of type n is denoted by γ^{rn}. The amount of computing resource r available at node v is denoted by Cap_v^r.

The objective of the proposed mathematical programming model is to decide on 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 total cost and complying with the limitations on the computing resources.

We introduce the following decision variables:
- x_v^n : number of VNFs of type n placed at node v,
- φ_v^a: filtering capacity dedicated for attack a at node v,
- f_p^a: illegitimate flow routed on path $p \in P^a$ of attack a,
- z_v^a: binary variable. $z_v^a = 1$ if some filtering of the malicious flow corresponding to attack a is carried out at node v, and $z_v^a = 0$ otherwise.

This leads to the bi-level programming formulation (1)-(6) displayed below.

The objective (1) of the leader problem, i.e. of the ISP, is to minimize the total costs of the deployed VNF. Constraints (2) ensure that the VNFs installed at node v do not consume more than the available computing capacity for each computing resource. Constraints (3) allocate the filtering capacity installed at node v to the identified attacks. Constraints (4) state that no filtering of attack a can take place at its target t^a. Finally, Constraints (5) translate the fact that we seek to avoid any damage to the attack targets by stopping all the malicious flow before it reaches them. Here, D represents the amount of malicious flow that will be able to reach the targets in the worst case, i.e. for the (up to now unknown) malicious flow routing which is the worst for the current filtering capacity allocation decided by the ISP.

Note how D is computed as the optimal objective value of a second sub-problem which can be seen as the follower sub-problem of the bi-level programming model. Given the filtering capacity allocated to stopping each attack $a \in A$, i.e. given $\varphi_v^a, \forall v \in V, a \in A$, decided by the leader problem, the sub-problem looks for the routing that will maximize the total amount of unfiltered malicious flow.

$$Z_{BL}^* = \min \sum_{v \in V} \sum_{n \in N} K^n x_v^n \tag{1}$$

$$\sum_{n \in N} \gamma^{rn} x_v^n \leq Cap_v^r \qquad \forall v \in V, \forall r \in R \tag{2}$$

$$\sum_{a \in A} \varphi_v^a \leq \sum_{n \in N} \phi^n x_v^n \qquad \forall v \in V \tag{3}$$

$$\varphi_{t^a}^a = 0 \qquad \forall a \in A \tag{4}$$

$$D = 0 \tag{5}$$

$$(x, \varphi) \in \mathbb{Z}_+^{NV} \times \mathbb{R}_+^{AV} \tag{6}$$

where D, the total damage inflicted to the attack targets, is the optimal value of sub-problem:

$$D = \begin{cases} \max \sum_{a \in A} \Big(\sum_{p \in P^a} f_p^a - \sum_{v \in V} z_v^a \varphi_v^a \Big) & (7) \\ \sum_{a \in A} \sum_{e \in p, p \in P^a} f_p^a \le b_e \qquad \forall e \in E & (8) \\ \sum_{p \in P^a} f_p^a \le \psi^a \qquad \forall a \in A & (9) \\ z_v^a \ge \frac{\sum_{v \in p \in P^a} f_p^a}{\psi^a} \qquad \forall a \in A,\ \forall v \in V & (10) \\ z_v^a \in \{0,1\} \qquad \forall a \in A,\ \forall v \in V & (11) \end{cases}$$

2 Solution approach and preliminary computational results

Solution approach
We propose a new iterative solution approach based on a decomposition of the problem into two smaller sub-problems: one for the internet service provider and one for the attacker. This decomposition algorithm simulates a two player game between the ISP, who places the filtering VNFs with the aim to minimize his cost, and the attacker who aims to maximize the damage inflicted to the targets. More precisely, at each iteration, the ISP first decides where to locate the VNFs taking into account information about the attack routing used in the previous iterations by the attacker. Based on these placement decisions, the attacker then seeks to route the malicious flow through the network so as to maximize the amount of flow which will reach its target without being filtered. If no malicious flow could reach its target, the algorithm stops and the current VNF placement is considered optimal. If some malicious flow is not filtered, a set of filtering constraints (one for each attack a) is added to the leader problem to enable the ISP to update his VNF placement decisions.

The filtering constraints generated at the end of iteration i are obtained as follows. For each attack a:
1. Record the set of paths currently used by the attacker to route the flow, i.e. the set $\overline{P}_i^a \subset P^a$ of paths p such that $\overline{f}_p^a > 0$ in the current solution of the attacker.
2. Add the filtering constraint $\sum_{v \in V(\overline{P}_i^a)} \phi_v^a \ge \sum_{p \in \overline{P}_i^a} \overline{f}_p^a$ to the leader problem. Here, $V(\overline{P}_i^a)$ denotes the set of nodes belonging to at least one of the paths of $\overline{P}_i^a$.

Note that the attacker sub-problem is a mixed-integer linear program involving path-flow variables f_p^a whose number increases exponentially fast with the network size. To address this issue, we develop a heuristic solution approach based on column generation for the attacker sub-problem. Thus, based on the current placement of the VNFs, we first solve the linear relaxation of a restricted attacker sub-problem in which only one variable f_p^a (the one corresponding to the shortest path between s^a and t^a in terms of hops) is considered. We then iteratively generate additional variables f_p^a and solve again the linear relaxation of the restricted master problem until no more variable f_p^a with a negative reduced cost can be found. Note that, in our case, the pricing problem corresponds to finding a shortest path in a directed graph with non-negative edge weights and can thus be solved in polynomial time. Finally, we reintroduce the integrality constraints on variable z and solve the restricted master problem as a mixed-integer linear program.

Preliminary computational results
We now discuss the results of preliminary computational results carried out to assess the proposed solution approach and compare it with the previously published heuristic solution approach LCG presented in [2].

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. More precisely, we used 4 internet network topologies from the Internet Topology Zoo library and one topology

			LCG		DEC			
Topology	V	E	$Cost$	$Time$(s)	$Cost$	$\#IT$	$\#FC$	$Time$(s)
Goodnet	17	31	1473	0.3	858	6	22	1.2
BICS	33	48	1690	0.6	754	12	51	3.2
IntelliFiber	73	96	1737	2.4	1040	10	44	5.2
Free	120	167	1560	4.5	1014	6	22	6.6
Cogentco	197	245	1950	12.5	767	24	99	44.8

TAB. 1: Numerical results

corresponding to the network of the French company Free. $R = 2$ types of computing resources were taken into account at each node: the number of CPUs and the memory. We considered three types of nodes: small nodes with $Cap = (4, 32)$, medium nodes with $Cap = (40, 160)$ and large nodes with $Cap = (400, 1600)$. A single type of VNF 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 $K^1 = 130$. The number of source-target pairs was set to $|A| \in \{2, 5\}$. In each instance, we considered $|A|$ different sources and a single targeted node, which were randomly selected. The intensity of each attack (in Mbps) was randomly generated following the normal distribution $\mathcal{N}(50, 25)$.

For each considered topology and value of $|A|$, we randomly generated 5 instances, leading to a set of 50 medium-size instances. Each generated instance was solved with the LCG algorithm presented in [2] and with the solution approach described above. The mathematical programs were solved using the solver CPLEX 12.8.9 with the default settings. All tests were run on an Intel Core i5 (1.9GHz) with 16 Gb of RAM, running under Windows 10. For each solution approach and each topology, we report in Table 1 $Cost$, the average cost of the VNF placement, and $Time$ the average computation time. For the decomposition (DEC) algorithm, we also report $\#IT$, the average number of iterations and $\#FC$ the average number of filtering constraints added to the network manager problem.

Results from Table 1 first show that the proposed approach provides VNFs placement solutions which are significantly less costly than the ones provided by the LGC algorithm: the average cost is thus divided by a factor of 1.9 when using algorithm DEC instead of algorithm LGC. This is mainly explained by the fact that algorithm LGC assumes that the potential flow on each possible path between the source and the target of an attack is equal to ψ^a, the total amount of the attack a, whereas in practice, this value is limited by the bandwidth of the arcs belonging to this path. Algorithm LGC thus largely overestimates the real amount of the attack flow on each path, which leads to placing more VNFs than what is actually needed. Moreover, we note that the improvement in the solution quality is obtained at the expense of a reasonable additional computational effort: the average computation time is namely increased from 4.1s with algorithm LGC to 12.3s with algorithm DEC.

References

[1] E.B. Fernandez A.M. Alwakeel, A.K. Alnaim. A survey of network function virtualization security. *SoutheastCon 2018*, pages 1–8, 2018.

[2] S Haddad-Vanier, C Gicquel, L Boukhatem, K Lazri, and P Chaignon. Virtual network functions placement for defense against distributed denial of service attack. In *Proceedings of the 8th International Conference on Operations Research and Enterprise Systems*, pages 142–150, 2019.