DoS-Resilient Virtual Networks through Multipath Embedding and Opportunistic Recovery

Rodrigo R Oliveira, Leonardo R Bays, Daniel S Marcon, Miguel C Neves
Luciana S Buriol, Luciano P Gaspary, Marinho P Barcellos
Federal University of Rio Grande do Sul (UFRGS)
Institute of Informatics
Porto Alegre, RS, Brazil
{ruas.oliveira, lrbays, daniel.stefani, mcneves, buriol, paschoal, marinho}@inf.ufrgs.br

ABSTRACT

Network virtualization can potentially limit the impact of attacks by isolating traffic from different networks. However, routers and links are still vulnerable to attacks on the underlying network. Specifically, should a physical link be compromised, all embedded virtual links will be affected. Previous work protects virtual networks by setting aside backup resources. Although effective, this solution tends to be expensive as backup resources usually remain idle. In this paper, we present a novel virtual network allocation approach which explores the trade-off between resilience to attacks and efficiency in resource utilization. Our approach is composed of two complementary strategies, one preventive and the other reactive. The former embeds virtual links into multiple substrate paths, while the latter attempts to reallocate any capacity affected by an underlying DoS attack. Both strategies are modeled as optimization problems. Numerical results show the level of resilience to attacks and the low cost demanded by our approach.

Categories and Subject Descriptors
C.2.0 [Computer-communication networks]: General–Security and Protection

General Terms
Design, Optimization, Security

Keywords
Network Virtualization, Resource Allocation, Denial-of-Service

1. INTRODUCTION

Network virtualization allows multiple virtual networks to be embedded on the same physical substrate. In addition, virtual networks can be isolated from each other, thus allowing them to coexist independently. Hence, one of the main advantages of this paradigm is the use of isolation to limit the scope of attacks. This can be achieved by creating different, isolated virtual networks for each task, so traffic from one virtual network does not interfere with the others [11].

In a network virtualization environment, virtual networks are requested on-demand by different Service Providers (SPs), and embedded in the physical resources of an Infrastructure Provider (InP) [2]. The Virtual Network Embedding (or Allocation/Mapping) phase consists of assigning shares of physical resources to the overlaid virtual nodes and links [5]. This characteristic tends to increase dependency on certain physical resources, allowing an attacker to launch DoS attacks on virtual networks by compromising nodes and links of the underlying physical substrate. Specifically, a successful attack on a physical link affects all embedded virtual links. Note that, although this article focuses on attacks, the proposed solutions can also be applied to failures.

Previous research tackled this problem by setting aside additional resources as backup [12, 3, 8, 13, 14]. Although effective, this strategy may be too expensive since backup resources remain idle (i.e. are wasted) when there are no disruptions. Therefore, these solutions may be of limited applicability, pragmatically leaving virtual networks vulnerable to DoS attacks.

In this paper, we propose a novel embedding approach that improves the resilience of virtual networks without expending additional resources. Our approach is composed of two complementary strategies, one preventive and the other reactive. The preventive strategy, modeled as a Mixed-Integer Program (MIP), attempts to mitigate the initial impact of an attack by embedding each virtual link into multiple paths, thus preventing the virtual links from losing all their capacity. The reactive strategy, modeled as a Linear Program (LP), aims at partially or fully recover any capacity compromised by attacks. To achieve this goal, it uses any unaffected path (if such a path exists) to opportunistically recover the compromised capacity.

The proposed approach has to deal with three fundamental problems: (i) similarity among paths embedding the same virtual link can be exploited to increase the impact of an attack, thus reducing the effectiveness of our approach; (ii) online resource allocation may overload some physical links, causing virtual network requests that depend on those resources to be dropped; and (iii) the selection of paths with distinct end-to-end propagation delays aggravates out-of-order packet delivery. These problems are addressed with penalty functions that aim at achieving path disjointness, load balancing and differential delay minimization (§3.3). Overall, the contributions of this paper are three-fold:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

$ACM’13 March 18-22, 2013, Coimbra, Portugal.
Copyright 2013 ACM 978-1-4503-1656-9/13/03 ...$10.00.
Joint modeling of the three factors of the problem (§3). We combine the relative impact of path disjointness, load balancing and differential delay, thus achieving equilibrium between virtual network survivability and efficient resource allocation.

Formulation of the strategies as optimization problems (§4). The first strategy (multipath embedding) is modeled as a MIP, while the second (opportunistic capacity recovery) is modeled as a LP. These formulations generate optimal solutions in accordance to the predefined priorities (see above).

Analysis of the trade-off addressed by our solution (§5). Experiments show that the proposed approach can substantially mitigate DoS by decreasing bandwidth lost to 25% and reducing critical targets (single point of failure) to 1%. These benefits demand a low cost in acceptance rate (at most 14% less in our experiments).

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 defines the attack model, the main concepts related to virtual network embedding, and the three factors considered in our optimization model. Section 4 presents the MIP/LP optimization models for the two proposed strategies. Numerical results are discussed in Section 5, and Section 6 concludes this paper.

2. RELATED WORK

We focus on mitigating attacks that compromise physical resources to cause DoS on virtual networks. Since this problem is closely related to network survivability, we discuss related work in both conventional and virtualized networks. Most scenarios on network survivability are considered at the network design, that is, with a given demand matrix. This is not the case for network virtualization since requests are received in an on-demand basis. The most relevant work in this area is that of Huang et al. [10], which considers on-demand connection requests on SONET/SDH networks with pre-defined source/sink nodes. Their work however is unsuitable for network virtualization as node embedding (i.e., placement and requirements) should be considered in addition to link embedding.

Generally, resilience in virtual networks is provided by reserving backup resources. Rahman et al. [12], Chen et al. [3], and Guo et al. [8] try to deal with disruptions on substrate links. While [12] set aside a percentage of bandwidth on each substrate link, [3] and [8] allocate disjoint backup paths with sufficient additional resources to restore each primary path. Yeow et al. [1] attempt to protect virtual networks by allocating physically disjoint backup nodes and backup paths between this nodes. Yu et al. [14] consider regionalized disruption events on the substrate network (i.e., when a single event compromises multiple devices at the same location). In order to survive such events, each virtual node is replicated on a separate location, with backup paths passing through different locations. All these solutions may become too expensive as backup resources remain idle when no disruption occurs.

In contrast, some work attempt to recompute virtual link embeddings when disruptions occur [9]. Although these schemes do not rely on backup resources, reassigning paths may demand a long convergence time, leaving virtual networks inoperable during such periods.

Our approach differs from previous work for the following reasons: i) resilience is improved without the use of backup resources; ii) recoverability is achieved without the need to recalculate and reallocate end-to-end paths; and iii) constraints are relaxed, which leads to a less expensive strategy for the Infrastructure Provider.

3. PRELIMINARIES

We begin this section with the attack model and related assumptions. Then, we describe a network virtualization environment and the virtual network embedding problem. Finally, we present the objectives used by our multipath embedding formulation.

3.1 Attack Model

The main threat considered in this article is the disruption of the communication between virtual nodes caused by attacks (or failures) compromising parts of the underlying substrate network.

Attacks to physical routers and links. A DoS attack to a physical device may occur by obtaining physical access, or by exploiting vulnerabilities. The former happens when an attacker identifies the location of an optical fiber or a router and interrupts these devices (e.g., strategically planed fiber cuts have been known to affect thousands customers1). The later happens when an attacker exploits some vulnerability on the control software or protocol to compromise a device (recent news suggest that several networking equipment on the Internet operate with insecure firmware2).

Assumptions. The scope of the paper is defined by the following set of assumptions:

- The disruption of a node can be reduced to the disruption of multiple links, that is, those directly connected to the disrupted node.
- The attacker might infer the topology or at least part of it, identifying the most critical elements. The worst case scenario corresponds to an attack compromising the node with most aggregated bandwidth embedded on its incident links.
- Virtual networks are isolated among each other. Therefore, a virtual network cannot prejudice another by consuming more resources than it was granted.

3.2 Virtual Network Embedding

Virtual network embedding consists of allocating virtual nodes and links, respectively, on substrate nodes and paths. Each virtual node is embedded in a single physical node, and each virtual link is embedded into one or more paths of the substrate network. Moreover, virtual nodes of the same virtual network should be embedded on distinct physical nodes. The embedding of virtual networks is performed on-demand, as requests arrive. Hence, it is not possible to determine beforehand which virtual networks should be embedded. These definitions are similar to [6, 4].

Substrate. The substrate network is given by a weighted directed graph $G^S(N^S, E^S, A^S)$, where $N^S$ is the set of nodes, $E^S$ is the set of links and $A^S$ is the set of node and link attributes. We focus on the following attributes: CPU capacity of nodes, and bandwidth capacity and delay of links.

Virtual network request. Each request is modeled as a weighted directed graph $G^V(N^V, E^V, A^V)$, where sets $N^V$, $E^V$ and $A^V$ are analogous to the aforementioned ones.


2http://goo/gl/w3yNB, http://goo.gl/DSgtN
3.3 Optimization Objectives

As previously stated, our approach aims at improving the resilience of virtual networks in a way that can be cost-effective to the substrate network. This design goal requires dealing with three main factors: path disjointness, load-balancing, and differential delay minimization. Thus, in this section, we describe each factor and what we do to tackle them. Section 4.1 describes how we implement these factors in our formulations.

Prioritizing disjointness. Embedding a virtual link into multiple substrate paths tends to increase resilience to disruptions in the substrate network. However, unless we select paths which are sufficiently different, a virtual link will still be vulnerable when a single substrate link embeds most of the virtual link capacity. In fact, in a worst-case scenario a single link disruption could lead multiple virtual links to lose their entire capacity. Therefore, to address this problem, we define $\Xi = \sum_{v \in V} \xi_{S,v}$ as a function which penalizes path similarity. For now, assume that $\xi_{S,v}$ grows proportionally to the number of paths of virtual link $e^v$ that share a given substrate link $e^S$.

Load balancing. As mentioned earlier, virtual network requests are received on-demand. This implies that, on the long-run, a set of physical substrate links may become saturated. When this happens, virtual network requests that need to use a given substrate link (or a set of substrate links) may be rejected. Thus, to avoid this problem we define function $\Phi = \sum_{e^S} \phi_{e^S}$, which penalizes saturation of the substrate links. By avoiding saturation, this function tends to balance the overall load of the substrate network and avoid the creation of bottlenecks.

Minimizing differential delay. When splitting the capacity of a virtual link over multiple paths, it may happen that the selected paths present different propagation delays. This aggravates out-of-order packet delivery and hurts application performance. Tackling such problem may require an additional cost in memory or processing capacity for border routers or end-hosts. This additional cost is proportional to the difference of end-to-end delays among paths (i.e., the differential delay). Thus, similarly to [15], we define differential delay minimization as an optimization objective.

Consider $P_{e^v}$ to be the set of substrate paths selected to embed a given virtual link $e^v$. Also consider $D_p$ to be the delay of a given path $p$ and $\delta_{e^v} = \max_{p \in P_{e^v}} (|D_p - D_q|)$ to be the maximum difference of propagation delays among the set of paths $P_{e^v}$. Therefore, $\Delta = \sum_{e^v \in E^V} \delta_{e^v}$ penalizes the differential delay of each virtual link.

4. MITIGATING DISRUPTIONS

In this section, we present an approach for improving the resilience of virtual networks against denial of service attacks. Our focus is to protect against attacks that compromise physical devices in an attempt to disrupt the embedded virtual networks. Toward this end, we use two complementary strategies, one preventive (§ 4.1) and the other reactive (§ 4.2), as follows.

4.1 Resilient Virtual Network Embedding

This strategy aims at providing resilience by embedding each virtual link into a set of substrate paths such that:

(i) all capacity of those links are distributed along the paths; and (ii) substrate paths of the same virtual link have little or no similarity. Our formulation is given as follows.

Variables:

- $X_{n^S,n^V} \in \mathbb{B}$: indicates if substrate node $n^S \in N^S$ is being used to embed virtual node $n^V \in N^V$;
- $\mathcal{P}_{p,e^S,e^V} \in \mathbb{B}$: indicates if path $p$ uses substrate link $e^S \in E^S$ to embed the virtual link $e^V \in E^V$;
- $\mathcal{I}_{p,e^S,e^V} \in \mathbb{R}$ within $[0,1]$; indicates the amount of capacity of the virtual link $e^V \in E^V$ to be allocated along substrate link $e^S$ of path $p$.

Input:

- $c(.) \in A^*$: CPU capacity of a (physical/virtual) node;
- $b(.) \in A^*$: bandwidth of a (physical/virtual) link;
- $d(.) \in A^S$: propagation delay of a physical link;
- $P, |P|$: represent, respectively, the set of path indices and the cardinality of this set (i.e., maximum number of paths per virtual link).

Objective: minimize the impact of the three factors of the problem (§3.3),

$$\min Z = w_\Phi \cdot \Xi + w_\Phi \cdot \Phi + w_\Delta \cdot \Delta,$$

where $w_\Phi$, $w_\Xi$, and $w_\Delta$ indicate the relative importance of each part of the objective function. The definition of $\Xi$, $\Phi$, and $\Delta$ will be given after the constraints.

Constraints:

- $\sum_{n^S \in N^S} X_{n^S,n^V} = 1, \forall n^V \in N^V$; (2)
- $\sum_{n^V \in N^V} X_{n^S,n^V} \leq 1, \forall n^S \in N^S$; (3)
- $X_{n^S,n^V} \cdot c(n^V) \leq c(n^S), \forall n^S \in N^S$; (4)
- $\sum_{p \in P} \sum_{e^V \in E^V} \mathcal{P}_{p,e^S,e^V} \cdot b(e^V) \leq b(e^S), \forall e^S \in E^S$; (5)
- $\sum_{(a^S, b^S) \in E^S} \mathcal{P}_{p,(a^S, b^S),e^V} - \sum_{(b^S, a^S) \in E^S} \mathcal{P}_{p,(b^S, a^S),e^V} = X_{a^S,a^V} - X_{b^S,b^V}, \forall p \in P, \forall a^S \in N^S, \forall e^V = (s^V, t^V) \in E^V$; (6)
- $\sum_{p \in P} \sum_{b^S \in N^S} \mathcal{P}_{p,(a^S, b^S),e^V} - \sum_{p \in P} \sum_{a^S \in N^S} \mathcal{P}_{p,(b^S, a^S),e^V} = X_{a^S,a^V} - X_{b^S,b^V}, \forall a^S \in N^S, \forall e^V = (s^V, t^V) \in E^V$; (7)
- $\mathcal{I}_{p,e^S,e^V} \leq \mathcal{P}_{p,e^S,e^V}, \forall p \in P, \forall e^S \in E^S, \forall e^V \in E^V.$ (8)

Constraints (2) and (3) guarantee, respectively, that each virtual node is embedded in exactly one physical node and that they are embedded on different physical nodes. Constraints (4) and (5) ensure that physical node and link capacities are not exceeded. The composition of a valid end-to-end path is ensured by Constraint (6). Finally, the last two constraints define the amount of bandwidth capacity allocated on each path. They ensure flow conservation on each path [Constr. (7)] and that traffic only flows on links that are selected by paths [Constr. (8)].
Penalty functions: we prioritize disjointness by penalizing path similarity. This constraint is defined by approximating a piecewise-linear function which grows with the number of paths that share a given physical link. The following expression generalizes the piecewise function:

$$\xi_{e,v} \geq w_K \cdot \sum_{p \in P} p_{e,v} - c_k, \quad \forall e \in E, \forall v \in e.$$  

(9)

where constant $K (\leq |P|)$ indicates how many paths of virtual link $e^v$ overlap on physical link $e$, and $w_K = k^{s-1}$ ($s \geq 2$) weights this overlap. Constant $c_k$ is used to select which function dominates the value of $\xi_{e,v}$ for each sharing scenario. For example, suppose the maximum number of paths is set to four ($|P| = 4$) and a given virtual link $x^v$ has three of its paths sharing a substrate link $y^S$, then $\xi_{e,v}$ would equal $w_3 \cdot 3 - c_3$. Constant $c_k$ is calculated by the following expression: $c_k = w_K \cdot (K - 1) - [w_{K-1} \cdot (K - 1) - c_{K-1}]$, $\forall e \in E^S, \forall v \in e^v, c_0 = 0, w_0 = 0$. Since all values are given beforehand, we use Equation (9) to create the set of constraints for each sharing scenario. For example, with 2 paths the following two constraints would be created:

$$\xi_{e,v} \geq \sum_{p \in P} p_{e,v} - c_1, \quad \forall e \in E^S, \forall v \in e^v;$$

$$\xi_{e,v} \geq 1 \cdot \sum_{p \in P} p_{e,v} - c_2, \quad \forall e \in E^S, \forall v \in e^v.$$  

We penalize link saturation ($\Phi$) by employing a classic traffic engineering function proposed by Forster and Thorup [7]. This piecewise-linear function defines an exponential cost which is proportional to the substrate link capacity and utilization. We use this function by adding the same constraints and values defined by the authors.

Finally, to calculate differential delay we need to find the maximum and minimum delays of a given set of paths. As the order of paths is not important, we add the following constraint without altering the result of the optimization:

$$\sum_{\forall e \in E^S} \sum_{\forall v \in P} p_{e,v} \cdot d(e^S) \leq 0, \quad \forall v \in P, \forall e \in E^S.$$  

This constraint sorts paths in non-decreasing order of end-to-end propagation delay. Therefore, the differential delay of a virtual link $d_{e}^v$ can be calculated as follows:

$$d_{e}^v = \sum_{\forall e \in E^S} \sum_{\forall v \in P} p_{e,v} \cdot d(e^S) - \sum_{\forall e \in E^S} \sum_{\forall v \in P} p_{e,v} \cdot d(e^S), \quad \forall v \in e^v : P = \{1, \ldots, k\}.$$  

Avoiding cycles: the previous set of constraints does not prevent cycles in the network. To deal with this problem, we create the auxiliary variable $\mathcal{y}_{p,m,s,v}^e$ and the following constraint to count the number of hops:

$$\mathcal{y}_{p,n,m,s,v}^e - |E^S| + |E^S| \cdot p_{e,v} \leq \mathcal{y}_{p,m,s,v}^e + 1 \quad \forall p \in P, \forall e \in E^S, \forall v \in e^v.$$  

(10)

This constraint can be read as follows: "for each node connected to links selected by $\mathcal{y}_{p,n,m,s,v}^e$ and $\mathcal{y}_{p,m,s,v}^e$ would try to mutually decrement, counting towards infinity.

### 4.2 Opportunistic Capacity Recovery

A key contribution of this paper is what we call opportunistic recovery. Instead of allocating backup resources, the capacity compromised by a successful DoS attack is reallocated over the available bandwidth along the set of unaffected paths. These paths can be any path that was previously selected to allocate a virtual link and remained operational after an attack. From this point onwards, we will refer to such paths as active paths.

Whenever a physical link becomes inaccessible, one or more paths of a virtual link may disappear. If the virtual link has no active path left, it means it was fully compromised. Otherwise, one or more active paths remain available, and any spare bandwidth remaining on these paths can be used to restore the capacity of the virtual link. The formulation of this strategy is given as follows:

**Variable:**

- $Q_{p,e,v} \in \mathbb{R}$ within $[0, 1]$: the fraction of capacity each virtual link $e^v$ reallocates to path $p$.

**Input:**

- $E^S \subset E^S$: substrate links that remain available after a disruption event;
- $E^V \subset E^V$: virtual links affected by the disruption;
- $P_{e,v}^v$: active paths of virtual link $e^v$ after disruption;
- $F_{e,v}^v$: compromised capacity of virtual link $e^v$;
- $P_{p,e,v}^p \in \mathbb{R}$: indicates if physical link $e^S$ is being used by active path $p$ on virtual link $e^v$.

**Objective:** maximize bandwidth capacity recovered after disruption events. Therefore, function

$$\max T = \sum_{\forall v \in e^v} \sum_{\forall p \in P_{e,v}^v} Q_{p,e,v}^v,$$  

(11)

attempts to allocate all spare bandwidth available over the set of active paths.

**Constraints:**

$$\sum_{\forall p \in P_{e,v}^v} Q_{p,e,v}^v \leq F_{e,v}^v, \quad \forall v \in e^v;$$

$$\sum_{\forall v \in e^v} \sum_{\forall p \in P_{e,v}^v} P_{p,e,v}^p \cdot Q_{p,e,v}^v \cdot b(e^v) \leq b(e^S), \quad \forall v \in e^v.$$  

(12)

(13)

The first constraint [Constr. (12)] guarantees that the amount of bandwidth reallocated cannot be greater than the amount compromised by an attack. The second constraint [Constr. (13)] ensures that the allocated bandwidth remains within the available capacity of physical links.

### 5. EVALUATION

The strategies described in the previous section have two main roles: (i) to prevent against the impact of attacks by allocating, efficiently, virtual links into multiple paths; and (ii) to react against these attacks, whenever possible, by reallocating compromised capacity over a set of active paths. This section evaluates the proposed approach by quantifying the impact of attacks over the affected virtual links - with and without our strategies - as well as the amount of accepted requests. Further, we evaluate the time needed to optimally allocate virtual network requests.

The mathematical formulation described in this paper was implemented on CPLEX 12.3. All experiments were executed on an Intel i7 with 8 cores of 2.93 GHz and 8 GB of RAM.
5.1 Evaluation Settings

The time of our experiments is organized in rounds. Each round can receive one or more virtual network requests and execute the allocation algorithm. Furthermore, the recovery algorithm is executed whenever an attack occurs. The round only finishes after concluding all steps of the allocation and/or recovery algorithms.

The network topologies were synthetically generated using BRITE with the Barabási-Albert (BA-2) network model. Next, we present the network and workload configurations used in this paper, which are in line with those in [6, 4].

**Substrate.** The physical network is composed of 30 nodes and 114 links on a 60x60 grid. The CPU and bandwidth capacity of nodes and links are uniformly distributed within $[50,100]$. The propagation delay of links is directly proportional to the distance between nodes, as generated by BRITE, normalized to the greatest value (i.e., within $(0,1]$).

**Virtual networks.** The capacity of nodes and links is uniformly distributed within $[5,20]$ and $[50,100]$, respectively. These values were chosen so that a single attribute would not be the main factor of impact on the experiments. The number of nodes on a virtual network request is uniformly distributed within 2 and 4. The use of such low values was necessary to perform the set of experiments as the allocation phase consumes too much CPU time. This limitation is discussed at the end of Section 5.2.

**Workload.** Our workload is composed of virtual network requests and denial of service attacks. The arrival rate of each request is given by a Poisson process with an average of 7 requests per 100 rounds. The duration of each request is assumed to be geometrically distributed with an average of 1000 rounds. The attacks are modeled by a Poisson process with mean of 1 attack per 100 rounds and duration of 10 rounds. As described earlier (§3.1), each attack is launched against the physical node that has the greatest bandwidth allocation on its links.

Experiments where executed during 5000 rounds. Thus, each experiment generated, approximately, 350 requests and 50 attacks. Finally, CPLEX gap to optimally was set to 1%, so solutions could be found in feasible time.

5.2 Evaluation Results

The proposed strategies help mitigating DoS attacks by preventing bandwidth loss. Fig. 1 depicts the effect of physical network disruptions with and without the use of our strategies. In all scenarios, $w_E$, $w_F$, $w_D$ were fixed in 1/3. Axis x and y represent, respectively, the number of paths to be used when embedding a virtual link and the average bandwidth loss after the attacks; therefore, the lower the bar, the better it is. Intuitively, increasing the number of paths should make the network more resilient to attacks.

![Figure 1: Bandwidth loss and bandwidth recovery on scenarios with 1, 2, 4 and 6 paths per virtual link, and $w_E = w_F = w_D = 1/3$.](image)

Our baseline have only one path per virtual link. In this case, when an attack occurs, all allocated bandwidth is lost and the virtual links become inoperable. The increase in the number of paths per virtual link allows a higher level of protection, since the proposed strategies work together against attacks. In Fig. 1, the preventive strategy is presented by dotted-lines (red bars), while the behavior of both strategies is presented by full lines (blue bars). Observe that, in the scenario with 6 paths per virtual link, the preventive strategy can protect approximately 50% of the capacity. As expected, results are improved when combining both strategies. Fig. 1 shows that the second strategy is able to recover part of the compromised capacity, so that we can reduce this loss to approximately 25%. Moreover, although the efficiency of our strategies increases with the number of paths, this behavior is nonlinear. This happens because the substrate network has a limited amount of disjoint paths available to embed each virtual link. We provide further evidence of this behavior on our next experiment.

![Figure 2: Amount of paths that use the same substrate link.](image)

**Multipath allocation reduces the severity of attacks to critical targets.** We define as “critical target” a substrate resource, namely a node or link, that concentrates the full capacity of a virtual link by a circumstance of the physical network and/or the allocation procedure. According to this definition, if an attack on a critical target succeeds, one or more virtual links would be fully compromised. The CDF presented in Fig. 2 shows the percentage of virtual links vulnerable to such attacks. The X axis indicates how many paths of the same virtual link utilize a given physical link (i.e., 1 means that only one path uses the substrate link and there is no sharing). When considering 2 paths per virtual link ($|P| = 2$), approximately 11% of virtual links have paths that share the same physical link. By increasing the number of paths, the percentage of vulnerable virtual links decreases to 4% ($|P| = 4$), and then to 1% ($|P| = 6$). On the other hand, it is also possible to see that the percentage of virtual links with disjoint paths also decreases. The importance of disjoint paths is that if one or more paths share the same underlying physical link, the efficiency of the opportunistic recovery strategy is limited by the capacity of the shared link. When virtual links are embedded on 2 paths, 89% of these virtual links have disjoint paths (i.e., do not share any substrate link). This decreases to 7% when the number of paths per virtual link increases to 6. This evidence sustains our earlier observation that the amount of disjoint paths on the substrate affects the performance of our approach.

The allocation cost is proportional to the level of resilience against attacks. Fig. 3 illustrates the acceptance ratio through time in scenarios using different parameters. These graphs should be read as follows. At first (0 ~ 900), none of the arriving virtual network requests are rejected since the substrate has sufficient resources to embed all of them. As time passes and more requests arrive, resources become scarce.
Figure 3: Impact on acceptance ratio when varying (a) parameters $w_{\Xi}$, $w_{\Phi}$, and $w_{\Delta}$ and (b) the number of paths per virtual link.

(900 ~ 2000) which results in some requests being rejected. Finally, the allocation process stabilizes (2000 ~ $\infty$) when the amount resources being freed is approximately the amount demanded by new requests, and acceptance ratio stays between 40 and 60%. Fig. 3(a) shows how prioritizing one factor over the others can affect the acceptance ratio of the substrate network. Our baseline curve has $w_{\Xi} = w_{\Phi} = w_{\Delta} = 1/3$. The other three curves have the main factor set to 0.998 and the other two to 0.001. In practice, this is the same as replacing our 3-factor objective function by one that has a single factor. When we prioritize path disjointness ($w_{\Xi}$), the amount of allocated substrate resources is increased and the acceptance ratio decreases. This effect happens because the allocation algorithm tends to choose longer paths to avoid sharing substrate links. Prioritizing less saturated links ($w_{\Phi}$) or differential delay minimization ($w_{\Delta}$) results in the best and worst acceptance ratios, respectively (with a reduction of 11% in acceptance ratio). On the one hand, increasing $w_{\Phi}$ helps to avoid bottlenecks, which tends to leave free access to substrate nodes. On the other, increasing $w_{\Delta}$ prioritizes paths with similar delays, which means longer paths with similar delays may be chosen over shorter paths with more distinct delays.

Fig. 3(b) shows the efficiency of our allocation algorithm through time when varying the number of paths per virtual link. Although increasing the number of paths provides better granularity, it tends to result in worse allocations. In the worst case, when considering 6 paths, acceptance ratio drops 14%. This effect happens because prioritizing disjointness ($\Xi$) and differential delay minimization ($\Delta$) does not take any effect when employing a single path. However, the relative importance of both metrics, and their collateral effect on allocation, increases with the number of paths.

The difficulty of this problem increases exponentially, thus justifying the study of heuristics. Previous work proposes heuristics to deal with the virtual network embedding problem [6, 4]. Its similarity to the multiway separator problem [1] is a strong evidence that this problem is NP-Hard. Our virtual network embedding approach requires few milliseconds to optimally solve a 2-node Virtual Network Request (VNR), yet it demands approximately 250s to optimally solve 4-node VNR. Therefore, the natural evolution of this work consists in defining heuristics (e.g., Greedy algorithms) and meta-heuristics (e.g., Simulated Annealing) to improve the scalability of our approach.

6. CONCLUSIONS

One of the potential advantages in network virtualization is the use of isolation to withstand network attacks. Nonetheless, in this paper we argued that virtual networks are still vulnerable to DoS attacks performed by compromising physical substrate resources. Moreover, we presented a novel approach to protect against such attacks. Among our contributions, we highlight that, unlike previous work, our allocation approach does not need to set aside backup resources or recalculate end-to-end paths to reallocate virtual links. Rather, we divide our approach in two complementary strategies. The first protects against attacks by embedding virtual links into multiple, preferably disjoint paths. The second attempts to recover any compromised capacity over the set of active paths (i.e., those unaffected by the attack).

Our solution improves the protection against DoS attacks at the cost of a lower acceptance ratio. Additionally, this improvement gets proportionally smaller as resilience is prioritized over allocation. Therefore, it is possible to adapt the solution to the requirements of the environment. For instance, one can choose to get the best possible resilience by sacrificing the efficiency of allocation, or else adopt a more conservative approach and still mitigate disruptions.

Regarding future work, we envision heuristics that allow our approach to be used on larger scenarios. Moreover, we are investigating the incorporation of migration to the recovery process, making networks more robust against attacks.

Acknowledgment

This work has been supported by European Commission’s FP7 and CNPq Research Agency (Project SecFuNet, FP7-ICT-2011-EU-Brazil), RNP’s CTIC program (Project ReVir), as well as PRONEM/FAPERGS/CNPq (Project NPRV).

7. REFERENCES