Sustainability of Service Provisioning Systems under Stealth DoS Attacks

Georgios S. Paschos* and Leandros Tassiulas†

* Mathematical and Algorithmic Laboratory, Huawei Technologies Co. Ltd., France
† Yale Institute for Network Science, USA

***Abstract*—We model a service provisioning system under attack by malicious intruders. The system consists of a bank of servers providing service to incoming requests. Malicious intruders generate fake requests attempting to degrade service provisioning–the fake traffic is assumed low-rate and thus it is practically undetectable. Legitimate traffic may be balanced using available mechanisms in order to mitigate the damage from the attack. We characterize the *guaranteed throughput region*, i.e. the legitimate traffic intensities that are guaranteed to be supported given specific intensities of the fake traffic. The result is first obtained under the assumption that fake traffic is routed using any static routing. Then, we relax this assumption allowing time-varying attacks. We show, that depending on the resources of the malicious attacker, and by the use of non-stationary attack policies, some of the servers are effectively neutralized and the guaranteed throughput is greatly compromized. We further examine the interaction between specific policies and encounter interesting phenomena such as the Join-the-Shortest-Queue is not maximally stable defense policy under specific time-varying attacks. The study offers defense insights, how to design the system and how to balance the traffic to sustain such attacks.**

I. Introduction

The *Denial of Service* (DoS) attack is an attempt to cause a server to stop functioning properly. One common method of attack involves saturating the target machine with communications requests, such that it cannot respond to legitimate traffic, or it responds so slowly as to be rendered effectively unavailable. DoS attacks result in service downtime for corporations and organizations that use Internet services, which in turn is translated to significant financial costs [1], [2], [23].

A DoS attack may be manifested as a multi-source attack, where multiple hosts coordinate to flood the victim with a barrage of requests. This attack is called *Distributed Denial of Service* (DDoS). In many examples, the malicious intruder(s) may manipulate a great number of computers (or network devices) in the ignorance of their users and use them to launch a DDoS attack. Botnet services are also available on the Internet for sale; it is possible to hire distributed computing power in order to launch DDoS attacks at a cost of a few dollars per hour. A recent article [24] uncovers the hidden economical aspects of this war, the *botconomics*.

Defending against DoS and DDoS attacks typically involves the use of a combination of attack detection, traffic classification and response tools, aiming to block traffic identified as illegitimate and allow traffic identified as legitimate [22]. Depending on the type of exploit, defending may be difficult. Common exploits include manipulated social media, Peer-to-Peer communications systems, TCP-SYN flood, HTTP GET, and many others. For a taxonomy of DoS attack exploits and defense mechanisms see [10], [20].

Since 2005, more elegant types of attack have appeared, for example the *pulsing* attack [18] or the Low-rate DoS (LDoS) attack. In these attacks, the participating computers (also termed *zombies*) alternate the intensity of the attack between low and high in order to avoid detection. Prior work [17] explains how this pulsing attack can be optimized from the scope of the attacker. Different names for this type of attack exist; Reduction of Quality (RoQ) attack is studied in [8], [9] and Denial of Quality (DoQ) attack is studied in [25]. We collectively refer to these types of attack with the term *stealth DoS*, also used in [6], [16], [28]. The difference from standard DoS lies on the fact that *the stealth DoS attack aims to reduce the system's resources and lead to service deterioration, rather than leading to complete service interruption*. This kind of attack is harder to detect since it results in a typical behavior of a critically loaded system, a situation which arises naturally in systems with fluctuating demand.

Most defense mechanisms against stealth DoS attacks focus on statistical processing of arriving requests [11], the success of which however is based on the legitimate traffic exhibiting a steady behavior with no significant peaks and bursts. In this work, we assume that fake traffic is indistinguishable from the legitimate one. Therefore, our work characterizes the performance that can be guaranteed for the legitimate system in spite of an attack of given intensity.

We consider a system described in Figure 1, where a distributed server installation consisting of N servers, is under a stealth DoS attack. The malicious intruders control a large number of computers, clustered in botnets, which are able to generate requests for fake traffic. Fake traffic is similar in structure to legitimate and considered impossible to discern. On the legitimate side of the system, we assume the existence of machines which aggregate traffic and route it to servers (dispatchers) as well as uncontrolled traffic which for example selects a server according proximity criteria. We model botnets, dispatchers, and users, by traffic streams that can reach only a subset of the available servers. A study of static DDoS attacks in more general topologies is found in [5].

We study the system sustainability. Given the capabilities of

An abstract version of this paper was presented as poster in the ACM Sigmetrics 2013 [21]. This work was supported partly through ONR grant N00014-14-1-2190. The views and opinions expressed in this article are those of the authors and do not necessarily reflect the official policy or position of any agency of the U.S. government or Huawei technologies.

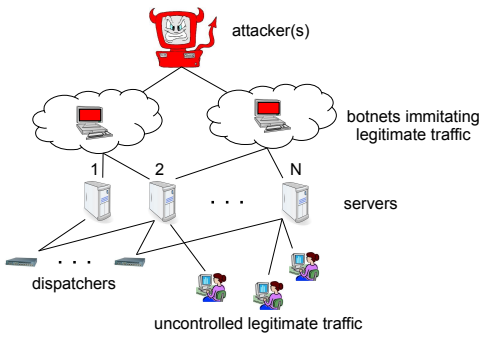

Fig. 1. A geographically distributed server installation under attack by botnets controlled by a malicious entity.

the malicious agents expressed in terms of the amount of fake traffic they could produce, a certain legitimate traffic intensity is sustainable if acceptable service can be provided to the legitimate traffic.

Initially, we assume that the fake traffic is directed to each server according to fixed routing coefficients. Under this model, we develop necessary and sufficient conditions for the *guaranteed throughput region*, i.e., the legitimate traffic intensities that are sustainable irrespective of the chosen malicious routing coefficients. Then, we proceed by allowing time-varying routing. In this extended model, we find that the guaranteed throughput region is reduced due to the fact that a non-stationary attack can effectively neutralize some of the servers. In the dynamic model, the problem is very rich and we observe unexpected phenomena from the interaction of the two scheduling controllers, the attacking and the defending. This is further demonstrated in section V, where the interaction between specific chosen policies is studied.

II. System Model and Definitions

Consider the set of parallel servers $\mathcal{N} \triangleq \{1, \ldots, N\}$, where server n has a constant service rate μ_n. Legitimate traffic arrives in the system at a set of input streams $\mathcal{L}$. At stream $l \in \mathcal{L}$ the arriving traffic has intensity a_l and can be routed to a subset of servers denoted with $\mathcal{S}_l \subseteq \mathcal{N}$.

Malicious intruders attempt a stealth DoS attack in order to disrupt the operation of the legitimate system. In particular, the attack is coordinated through a set of malicious input streams $\mathcal{M}$, where stream m generates fake traffic with intensity b_m and is capable of routing traffic to servers in the subset $\mathcal{Q}_m \subseteq \mathcal{N}$. See Figure 2 for an example of the studied system in terms of a bipartite graph. Throughout the paper we make the following modeling assumptions:

(a) Input streams (both legitimate and malicious) do not queue traffic and therefore they must route it to servers immediately.
(b) The attack is undetectable.
(c) The servers do not distinguish fake traffic from legitimate.

There exist two routing controllers with conflicting interests, the legitimate (defending) and the malicious (attacking). The legitimate is interested in stabilizing the system and the malicious in destabilizing it; stability will be explained shortly.

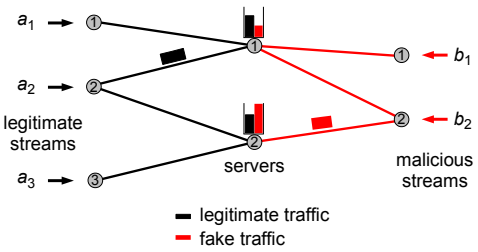

Fig. 2. An example of the system in terms of bipartite graph for $N = 2$ servers, $|\mathcal{L}| = 3$ legitimate streams and $|\mathcal{M}| = 2$ malicious streams. Also, in this example $\mathcal{S}_1 = \mathcal{Q}_1 = \{1\}$, $\mathcal{S}_2 = \mathcal{Q}_2 = \{1, 2\}$ and $\mathcal{S}_3 = \{2\}$.

A. Static Malicious Routing

To obtain intuition for the system operation we begin our analysis with the static routing model; this is relaxed in section IV. Additionally, the following static assumption is made:

S.1 The malicious controller chooses first the routing coefficients and then the legitimate controller learns this selection and chooses the best static routing response.

Assumption S.1 is equivalent to assuming a static attack and a dynamic defense, oblivious to the attack.[1] The studied scenario is of high practical interest since it captures cases where the intruder cannot make online routing decisions, e.g. due to constraints from the application or due to limited capabilities of the infrastructure with which the attack is launched. The case where both controllers make static decisions oblivious to each other is not the focus of this paper.

The legitimate controller splits traffic to allowable servers according to *routing coefficients* $f_{ln}, (l, n) \in \mathcal{L} \times \mathcal{N}$.

Definition 1. *(Legitimate policy* $\mathbf{f}$*)* A legitimate routing policy is a matrix (f_{ln}) with nonnegative elements having the following properties:

$$\sum_{n \in \mathcal{N}} f_{ln} = a_l, \quad \forall l \in \mathcal{L}$$
$$f_{ln} = 0, \quad \text{if } n \notin \mathcal{S}_l.$$

Let Π_1 be the set of all legitimate policies. The malicious controller operates in a similar manner.

Definition 2. *(Malicious policy* $\boldsymbol{\phi}$*)* A malicious routing policy is a matrix (ϕ_{mn}) with nonnegative elements having the following properties:

$$\sum_{n \in \mathcal{N}} \phi_{mn} = b_m, \quad \forall m \in \mathcal{M}$$
$$\phi_{mn} = 0, \quad \text{if } n \notin \mathcal{Q}_m.$$

Let Π_2 be the set of all malicious policies.

According to the classical stability condition, a server n is stable if the aggregate arrival traffic intensity is smaller or equal to its service rate. From the practical viewpoint, though, the stealth DoS attack is considered successful only if service to legitimate traffic fails. Thus, if some servers are unstable in the classical sense but they are avoided by the legitimate traffic then the system remains sustainable and the attack has failed. To capture this phenomenon, we use an unconventional stability definition as follows:

[1]The equivalence is related to the adaptivity of dynamic policies to time-varying service. A formal proof is omitted.

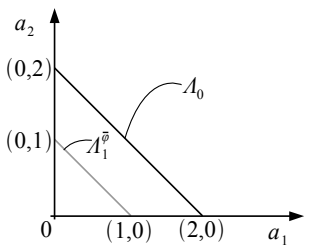

Fig. 3. Example 1: The stability region is reduced when the system is attacked by a malicious adversary using policy $\bar{\boldsymbol{\phi}}$ and traffic intensity $b = 1$.

Definition 3. *(System Stability)* A server $n \in \mathcal{N}$ is stable if *at least* one of the following conditions is satisfied

(i) $\sum_{l\in\mathcal{L}} f_{ln} + \sum_{m\in\mathcal{M}} \phi_{mn} \leq \mu_n$ (classical stability)
(ii) $\sum_{l\in\mathcal{L}} f_{ln} = 0$ (zero legitimate traffic).

The system is stable if all servers are stable in the above sense.

Let $\mathbf{a} \triangleq (a_1, \ldots, a_{|\mathcal{L}|})$ denote the vector of legitimate traffic intensities. Below we extend the standard notion of system stability region to include the impact of an attack with intensities $\mathbf{b} \triangleq (b_1, \ldots, b_{|\mathcal{M}|})$ and routing policy $\boldsymbol{\phi}$.

Definition 4. *(Stability Region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$)* The stability region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$ under an attack with intensities $\mathbf{b}$ and fixed malicious routing policy $\boldsymbol{\phi} \in \Pi_2$, is the set of all $\mathbf{a}$ for which there exists a legitimate policy $\mathbf{f} \in \Pi_1$ such that the system is stable.

Moreover, we define the notion of guaranteed throughput region as the set of legitimate traffic intensities $\mathbf{a}$ which are stabilizable regardless of the malicious policy used.

Definition 5. *(Guaranteed Throughput Region $\Lambda_{\mathbf{b}}$)* The guaranteed throughput region $\Lambda_{\mathbf{b}}$ under an attack with intensities $\mathbf{b}$, is the set of all $\mathbf{a}$ such that **for any** malicious policy $\boldsymbol{\phi} \in \Pi_2$ there exists a legitimate policy $\mathbf{f}(\mathbf{a}, \boldsymbol{\phi}) \in \Pi_1$ such that the system remains stable.

The guaranteed throughput region is parametrized by the fake traffic intensity, $\mathbf{b}$. For $\mathbf{b}$ large enough, $\Lambda_{\mathbf{b}}$ might contain only the zero vector $\mathbf{0} \triangleq (0, 0, \ldots, 0)$, which implies that even arbitrarily small legitimate traffic intensities cannot be guaranteed. In practical terms, we can think of such a situation as a regular DoS attack. *The stealth DoS attack, on the other hand, corresponds to cases where* $\mathbf{b}$ *is small and legitimate service can still be guaranteed despite the attack.* Under S.1 we have

$$\Lambda_{\mathbf{b}} = \cap_{\boldsymbol{\phi}\in\Pi_2} \Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}.$$

Example 1: Consider a system comprised of two legitimate streams, one malicious stream with $b \equiv b_1 = 1$ and two servers with $\mu_1 = \mu_2 = 1$. We have no routing constraints, i.e. $\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{Q}_1 = \{1, 2\}$. Note, that in the absence of the malicious intruder ($b = 0$), the guaranteed throughput region Λ_0 is the triangle defined by the origin and points $(0, 2), (2, 0)$ and it is equal to the stability region of the bipartite routing, see for example [26]. Then, for $b = 1$, pick a simple malicious policy $\bar{\boldsymbol{\phi}}$ with $(\bar{\phi}_{11}, \bar{\phi}_{12}) = (b, 0)$. $\Lambda_1^{\bar{\boldsymbol{\phi}}}$ is given by points satisfying $a_1 + a_2 \leq 1$ since a good legitimate policy will route all legitimate traffic to server 2, and completely avoid server 1. Figure 3 depicts the reduction of the stability region due to the attack with policy $\bar{\boldsymbol{\phi}}$. To derive the guaranteed throughput region we need to repeat this procedure for every malicious policy in set Π_2 and find the intersection of these regions, which in general is a very difficult task. For this reason we provide below a result that characterizes the guaranteed throughput region without examining all possible malicious policies. ■

III. Guaranteed Throughput - Static Attacks

In this section we present necessary and sufficient conditions that determine the guaranteed throughput region $\Lambda_{\mathbf{b}}$ for the case of static malicious routing.

A. Stability Region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$ in Deparametrized Form

First, we fix a malicious policy $\boldsymbol{\phi}$ and study the stability region under this policy. Define

$$r_n(\boldsymbol{\phi}) \triangleq \left(\mu_n - \sum_{m\in\mathcal{M}} \phi_{mn}\right)^+, \tag{1}$$

to be the *available resource* of server n after the traffic arriving from malicious streams under $\boldsymbol{\phi}$ is subtracted. We use $(.)^+ \equiv \max\{., 0\}$. Combining the definitions 1-3, we conclude that the system is stable *if and only if* there exists a legitimate policy $\mathbf{f}$ such that

$$\sum_{l\in\mathcal{L}} f_{ln} \leq r_n(\boldsymbol{\phi}), \text{ for all } n \in \mathcal{N}. \tag{2}$$

The above inequalities express the stability region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$ using flow variables $\mathbf{f}$. Next we develop a methodology for expressing the stability region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$ in terms of traffic intensities $\mathbf{a}, \mathbf{b}$ and service rates $\boldsymbol{\mu}$.

In the special case where $\mathbf{b} = \mathbf{0}$, our problem reduces to bipartite routing, whose stability region can be expressed in terms of arrivals and departures only, by means of the inequalities called *cut constraints* [12]–[14], [27]. In case the attack has $\mathbf{b} > \mathbf{0}$ but the attack policy $\boldsymbol{\phi}$ is fixed, we may obtain the corresponding cut constraints by replacing server n capacity with the remaining resource $r_n(\boldsymbol{\phi})$. Next, we perform this derivation.

First, let us define some useful notions. For an arbitrary non-empty subset of the servers $\hat{\mathcal{N}} \subseteq \mathcal{N}$ consider the induced subsets $\hat{\mathcal{L}}, \hat{\mathcal{M}}$,[2] where

- $\hat{\mathcal{L}} = \left\{l \in \mathcal{L} : \mathcal{S}_l \subseteq \hat{\mathcal{N}}\right\}$ is the set of legitimate traffic streams that **must** direct all traffic to some of the servers in $\hat{\mathcal{N}}$ and
- $\hat{\mathcal{M}} = \left\{m \in \mathcal{M} : \mathcal{Q}_m \cap \hat{\mathcal{N}} \neq \emptyset\right\}$ is the set of fake traffic streams that **can** direct fake traffic to some of the servers in $\hat{\mathcal{N}}$.

Sets $\hat{\mathcal{L}}, \hat{\mathcal{M}}$ are depicted in Figure 4 for a specific example of $\hat{\mathcal{N}}$. The following is a variation of Theorem 3.1 in [14].

Lemma 1 (Stability Region $\Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$ via cut constraints)**.** The traffic intensities $\mathbf{a}$ are sustainable under $\boldsymbol{\phi}$, and we write $\mathbf{a} \in \Lambda_{\mathbf{b}}^{\boldsymbol{\phi}}$, if and only if

$$\sum_{l\in\hat{\mathcal{L}}} a_l \leq \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}), \qquad \text{for all } \hat{\mathcal{N}} \subseteq \mathcal{N}.$$

B. Guaranteed Throughput Region $\Lambda_{\mathbf{b}}$

In this subsection we present necessary and sufficient conditions for the guaranteed throughput region. We will need an

[2]To avoid clutter we omit $\hat{\mathcal{N}}$ from $\hat{\mathcal{L}}(\hat{\mathcal{N}}), \hat{\mathcal{M}}(\hat{\mathcal{N}})$.

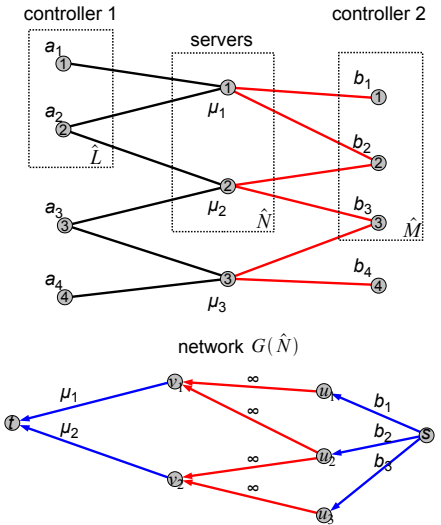

Fig. 4. An example system, a chosen subset $\hat{\mathcal{N}} = \{1, 2\}$ and the corresponding network where the maximum s-t flow $M_{\max}(\hat{\mathcal{N}})$ is calculated.

auxiliary network $\mathcal{G}(\hat{\mathcal{N}}) = (\mathcal{V}, \mathcal{E})$. Define the set of nodes as $\mathcal{V} \triangleq \{s, t, (u_i)_{i\in\hat{\mathcal{M}}}, (v_j)_{j\in\hat{\mathcal{N}}}\}$, where s is the source node, t is the sink, node u_i represents the element i of $\hat{\mathcal{M}}$, and node v_j represents the element j of $\hat{\mathcal{N}}$. The set of links consists of three disjoint subsets $\mathcal{E} = \mathcal{E}_b \cup \mathcal{E}_Q \cup \mathcal{E}_\mu$, where each subset contains the following *directional* links

$$\mathcal{E}_b \triangleq \{(s, u_i) : i \in \hat{\mathcal{M}}\},$$
$$\mathcal{E}_Q \triangleq \{(u_i, v_j) : i \in \hat{\mathcal{M}},\ j \in \mathcal{Q}_i\},$$
$$\mathcal{E}_\mu \triangleq \{(v_j, t) : j \in \hat{\mathcal{N}}\}.$$

A link of type (s, u_i) has capacity b_i, a link of type (v_i, t) has capacity μ_i, while all links in subset $\mathcal{E}_Q$ have infinite capacity. An illustrative example of the auxiliary network $\mathcal{G}(\hat{\mathcal{N}})$ is given in Figure 4. Let $M_{\max}(\hat{\mathcal{N}})$ denote the maximum s-t flow of network $\mathcal{G}(\hat{\mathcal{N}})$.

Using this auxiliary network we may now express the total remaining resource of a subset of servers after the worst attack is subtracted:

Lemma 2. Consider a subset of servers $\hat{\mathcal{N}} \subseteq \mathcal{N}$. The minimum remaining resource of this subset under any static attack is

$$\min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}) = \sum_{n\in\hat{\mathcal{N}}} \mu_n - M_{\max}(\hat{\mathcal{N}})$$

where $M_{\max}(\hat{\mathcal{N}})$ is the maximum s-t flow of network $\mathcal{G}(\hat{\mathcal{N}})$.

An attack that minimizes the remaining resource of a subset is called a *max-flow attack* for this subset. Intuitively, to guarantee sustainability of an arrival vector we must make sure that we can withstand any max-flow attack.

Theorem 1. *(Guaranteed Throughput Region)* The guaranteed throughput region is the set of all $\mathbf{a}$ satisfying

$$\sum_{l\in\hat{\mathcal{L}}} a_l \le \min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}), \qquad \text{for all } \hat{\mathcal{N}} \subseteq \mathcal{N}. \tag{3}$$

The proofs of lemmas and theorems are in the Appendix. We conclude the following for static malicious attacks:

- The impact of the worst possible attack on every subset $\hat{\mathcal{N}}$ is equal to $M_{\max}(\hat{\mathcal{N}}) \le \sum_{m\in\hat{\mathcal{M}}} b_m$.
- When $\mathcal{Q}_m = \mathcal{N}$ (no routing constraints for the malicious system), the total impact of the attack becomes equal to $\min\{\sum_{m\in\hat{\mathcal{M}}} b_m, \sum_{n\in\hat{\mathcal{N}}} \mu_n\}$.
- If a part of the system servers can be protected from the intruder, i.e. sever some links so that $\mathcal{Q}_m \subset \mathcal{N}$, then the benefit of this protection can be evaluated using (3).

IV. Guaranteed Throughput - Dynamic Attacks

In this section, we relax the assumption of static attack allowing the malicious controller to dynamically route the fake traffic. This has a severe impact on the legitimate system, which is characterized by the main result of subsection IV-D. In this section we (A) redefine the model to capture the dynamics, (B) introduce the notion of the dominated server, (C) define the optimal attacking policy DaR that periodically attacks the dominated servers and use it to (D) derive the guaranteed throughput region.

A. System Model for Dynamic Policies

The arriving legitimate traffic a_l at time t, is routed according to routing coefficients $f^\pi_{ln}(t)$ determined by the legitimate policy π. Similarly, the fake traffic b_m is routed according to the coefficients $\phi^\sigma_{mn}(t)$ determined by the malicious policy σ. We will denote with Π^t_1, Π^t_2 the corresponding set of policies to emphasize the time-varying aspect.

On server n there are two queues with backlogs denoted with $X^L_n(t)$ and $X^M_n(t)$, the first holding legitimate traffic and the second fake. In practice the servers only recognize the sum $X_n(t) = X^L_n(t) + X^M_n(t)$. For modeling purposes we assume that the servers employ the *Processor Sharing* discipline in the following sense: at time t, a server with total service rate μ_n serves legitimate traffic with rate $\mu^L_n(t) = \frac{X^L_n(t)}{X_n(t)}\mu_n$ and fake traffic with rate $\mu^M_n(t) = \frac{X^M_n(t)}{X_n(t)}\mu_n$, or zero if $X_n(t) = 0$. This model is known in the Queueing literature as Head-of-the-Line Proportional Processor Sharing (HLPPS) [4]. We choose this discipline because the service does not depend on the packet arrival order, which facilitates analysis. In section IV-E we experiment via simulations with the First-Come-First-Serve (FCFS) discipline and verify that the system behavior is qualitatively the same under the two models.

We also make the following assumptions regarding the available information to the two controllers:

D.1 The legitimate controller learns the total backlogs $X_n(t)$ at the end of the slot t.

D.2 The malicious controller learns the individual backlogs $X^L_n(t), X^M_n(t)$ at the end of the slot t.

Since legitimate systems are often not designed to keep track of sent packets, D.1 is a standard feedback assumption. D.2 models the common situation where the malicious intruder is a single powerful entity, with extra access to information of queued fake jobs $X^M_n(t)$, and by combining with available feedback $X_n(t)$ can obtain $X^L_n(t)$.

We redefine stability for this model as follows:

Definition 6. *(System Stability for the Dynamic Model)* Server n is stable if there exists a constant $D < \infty$ such that

$$\limsup_{t\to\infty} X^L_n(t) < D, \tag{4}$$

The system is stable if all servers are stable.

Note that in contrast to definition 3 we need not deal explicitly with the case where server n is not used by the legitimate system, since if this was the case we immediately have $\limsup_{t\to\infty} X_n^L(t) = 0$.

We naturally extend the definitions of stability region and guaranteed throughput region using the enriched sets of policies and the new stability criterion.

B. Dominated Servers

We introduce the notion of the *Dominated* server, which will play a key role in determining the guaranteed throughput region.

Definition 7. *(Dominated server)* A server $n \in \mathcal{N}$ is called dominated if there exists a static malicious routing $\phi \in \Pi_2$ such that

$$\sum_{m\in\mathcal{M}} \phi_{mn} - \mu_n > 0. \tag{5}$$

In words, a server is called dominated if it is possible for some static attack to overload it with fake jobs. Let $\mathcal{N}_d \subseteq \mathcal{N}$ be the set of all dominated servers and $\mathcal{N}_f \triangleq \mathcal{N} - \mathcal{N}_d$ the set of all *free* (non-dominated) servers. Note that the sets $\mathcal{N}_d, \mathcal{N}_f$ are determined once $\boldsymbol{\mu}, \mathbf{b}$ and $\mathcal{Q}_m$ are given and do not depend on the choice of policies from either controller.

From (5) it follows directly that the backlog of a dominated server n will grow at a positive rate $\rho_n \triangleq \sum_{m\in\mathcal{M}} \phi^*_{mn}(n) - \mu_n > 0$ if the malicious routing $\boldsymbol{\phi}(t)$ is assigned the value of the maximizer of the left term in (5), denoted here $\boldsymbol{\phi}^*(n)$. The actual rate of increase might be larger if legitimate traffic is also added. The key observation here is that: *for any dominated server n, the malicious controller can impose an instantaneous total backlog increase of rate at least ρ_n by targeting this server with all possible fake traffic*.

C. The DaR Malicious Policy

Next, we define the "Dominate and Release" (DaR) policy. The policy operates in periods, where period i has duration τ_i, which is a parameter chosen by the policy. Each period has two phases. *In the Dominate phase*, the policy targets one by one all dominated servers causing their backlogs to increase. The time spent on j^{th} dominated server t_i^j is designed so that *at the end of Dominate phase* all dominated servers have backlogs greater than a parameter B_i. The entire phase lasts for d_i:

$$d_i = \sum_{j=1}^{|\mathcal{N}_d|} t_i^j.$$

In the Release phase, the policy performs a static routing directed only to free servers for duration r_i.

Dominate and Release (DaR) Policy

Fix an order of the dominated servers $(1,\dots,J), J \triangleq |\mathcal{N}_d|$.

The **Dominate** phase is composed of J intervals, where in interval j the fake traffic is targeted to j^{th} dominated server. The duration of j^{th} interval t_i^j is chosen according to the recursive expression

$$t_i^j = \frac{t_i^{j-1}\rho_{j-1}}{\rho_{\max} + \mu_{\max}}, \tag{6}$$

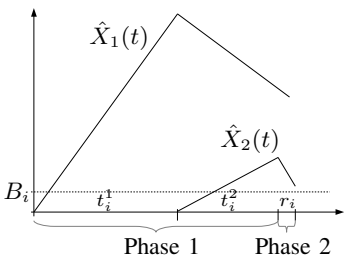

Fig. 5. Lower bounds for backlog evolution of the dominated servers $\hat{X}_1(t)$ and $\hat{X}_2(t)$ during a period τ_i of DaR policy operation. The bounds are tight if we assume that the dominated servers start empty and no legitimate traffic is sent to them. The first phase lasts for $t_i^1 + t_i^2$ time, whereby the two servers are targeted one-by-one. At the end of the first phase, both have a backlog larger than B_i. Then in the second phase of duration r_i the malicious controller attacks the free servers (their backlogs are not shown in the Figure).

where $\rho_{\max} = \max_{j\in\mathcal{N}_d} \rho_j$ and $\mu_{\max} = \max_{j\in\mathcal{N}_d} \mu_j$.

In the **Release** phase the fake traffic is directed to the free servers according to a static routing $\phi^f \in \Pi_2$, which is customizable. This phase lasts for $r_i = B_i/\mu_{\max}$, where $\mu_{\max}$ is the maximum server capacity and $B_i \triangleq t_i^J \rho_J$ is the draining time of the J^{th} dominated server.

The choice of (6) guarantees that at the end of the entire i^{th} dominate phase (denoted with t^i_{end}) all dominated servers satisfy

$$X_n(t^i_{\text{end}}) \geq B_i = t_i^J \rho_J,$$

where B_i corresponds to a backlog level, see Fig. 5. Intuitively, $t_i^{j-1}\rho_{j-1}$ is the "charge" of dominated server $j-1$, and it must be equal to the charge of the next server j plus a term $t_i^j \mu_{\max}$, which accounts for draining (discharging) of server $j-1$ at maximum rate while j is charging. Then both servers $j-1, j$ remain charged roughly at the same level. By repeating this trend we have that at the end of the dominate phase all dominated servers are charged with fake traffic.

Next we show, that due to the policy design, all parameters t_i^1, B_i, d_i, r_i in period i are linear functions of τ_i, and hence can be controlled. Our main result in subsection IV-D will be based on the fact that we can choose an increasing sequence $\tau_i, i = 1, 2, \dots$ so that B_i is also an increasing sequence.

Lemma 3. *(Parameter dependence on τ_i)* Consider a DaR attack policy and suppose $\mathcal{N}_d \neq \emptyset$, then the following quantities are proportional to τ_i:

(a) the first interval of the dominate phase t_i^1,
(b) the uniform lower bound of dominated server backlogs at the end of the dominate phase B_i, and
(c) the duration of the release phase r_i.

In particular

$$t_i^1 = C_1\tau_i, \quad B_i = C_2\tau_i, \quad r_i = C_3\tau_i.$$

where $C_1 = \frac{1}{c_1+c_2}, C_2 = \frac{\mu_{\max}c_2}{c_1+c_2}, C_3 = \frac{c_2}{c_1+c_2}$, and $c_1 \triangleq 1 + \sum_{j=2}^{J} \frac{\prod_{k=1}^{j-1}\rho_k}{(\rho_{\max}+\mu_{\max})^{j-1}}$, $c_2 \triangleq \frac{\prod_{k=1}^{J}\rho_k}{(\rho_{\max}+\mu_{\max})^{J-1}\mu_{\max}}$.

Since the dominated servers all become loaded at the end of the dominate phase with $B_i = C_2\tau_i$, the legitimate scheduler should avoid sending traffic to them during the release phase. The following Lemma provides a bound on the legitimate traffic that can be routed to the dominated servers under a DaR attack without causing a legitimate buffer overflow.

Lemma 4. *(Maximum legitimate traffic in dominated servers)* Consider a DaR attack policy with release phase routing ϕ^f and suppose $\mathcal{N}_d \neq \emptyset$. Then there exists an interval of duration $\xi(\tau_i)$ starting at the beginning of the release phase of the i^{th} period, such that if the total amount of legitimate traffic routed to the dominated servers during this interval is

$$A(\mathcal{N}_d) \geq 2|\mathcal{N}_d|D,$$

then the legitimate backlogs of the dominated servers will overflow D, where D is the stability threshold in (4). Moreover, $\xi(\tau_i)$ is linear to τ_i and independent of the order of dominated servers $1, \ldots, J$:

$$\xi(\tau_i) = \frac{C_2^*}{2\mu_{\max}}\tau_i,$$

where C_2^* is the minimum C_2 (defined in Lemma 3) over all possible orderings.

Lemma 4 implies that there exists an interval of length ξ in the release phase whereby the majority of legitimate traffic must be routed to the non-dominated servers. Combining this with an appropriate choice of ϕ^f, it can be shown that DaR is an optimal dynamic attack. In what follows, we use DaR to characterize the guaranteed region in the case of dynamic malicious policies.

D. Guaranteed Throughput Region - Dynamic Case

Theorem 2. *(Guaranteed throughput region under dynamic malicious policies)* The guaranteed throughput region $\Lambda_{\mathbf{b}}^{\text{dyn}}$ is given by all vectors $\mathbf{a}$ for which the following conditions are satisfied

$$\sum_{l\in\hat{\mathcal{L}}} a_l \leq \min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d} r_n(\boldsymbol{\phi}), \qquad \text{for all } \hat{\mathcal{N}} \subseteq \mathcal{N}. \tag{7}$$

Note the difference from Theorem 1, i.e. the service from dominated servers $\hat{\mathcal{N}} \cap \mathcal{N}_d$ is not counted as remaining resource when considering subset $\hat{\mathcal{N}}$. We reach the following concluding remarks:

- In a dynamic attack the servers with individual capacity less than the attack intensity become neutralized.
- Since an attacker can time-share between dynamic and static attacks, it may reach any intermediate performance degradation rate it wishes. If we make a crude assumption that an attack is detectable if more than a fraction of the system capacity is lost, then the attacker can incur the maximum possible damage subject to being undetectable, while using only a relatively small attack intensity (yet larger than individual server capacity).
- Consider the case of a system with a very large number of servers each with small capacity, i.e., each server can be a virtual machine. Then the attacker can destabilize this system with a very small attack intensity. In particular, it can control exactly the volume of the inflicted damage.
- A defense mechanism is to use load migration. In such a case the effect of dynamic attacks is mitigated since the resource becomes flexible, and equivalently unified in one virtual server with capacity equal to the sum capacities of individual servers.

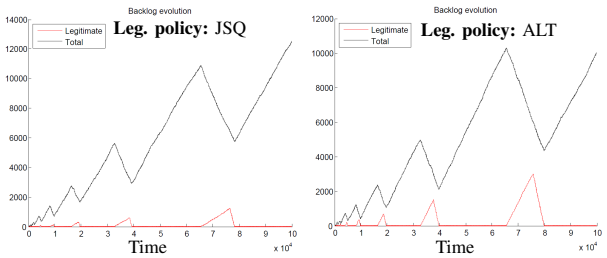

Fig. 6. **Malicious policy:** DaR with $\tau_i = 2^i$. Aggregate backlogs for legitimate (red) and total (black).

E. Model Validation via Simulations

In this section we present simulations for a system with random arrivals and service times, and First-Come-First-Serve (FCFS) discipline in order to demonstrate that the presented results are not an artifact of our traffic model, nor of our HLPPS assumption.

1) Setup

We study a simple example with two servers of unit rate, one legitimate stream with $a = 0.3$ and $\mathcal{S} = \{1, 2\}$ and one malicious stream with $b = 1.3$ and $\mathcal{Q} = \{1, 2\}$.

We use a custom built, event-driven simulator. Time is continuous. We use Poisson arrivals to generate jobs and FCFS queues at the servers. The duration of each job (legitimate and fake alike) is determined by an exponential random variable with unit mean.

Since all servers are dominated, the malicious controller is using DaR with no release phase, i.e., it simply alternates traffic between the two servers. For the interval lengths t_i^1, t_i^2 we use function 2^i. The legitimate controller uses either JSQ or a different policy which directs all traffic to the server that the malicious is not targeting. We call this policy Alternate (ALT).

Using intuition from Theorem 2 we expect the system to be unstable for any positive arrival rate. Additionally, it is of interest to compare the performance of the two legitimate policies.

2) Experiments

In Figure 6, we test DaR vs JSQ (left) and ALT (right). We verify that the system is unstable, since the legitimate backlogs (red) grow with time. This is an indication that our results also hold true for random arrivals and FCFS server scheduling. Then, observe that JSQ outperforms ALT, i.e. the legitimate backlogs are smaller under JSQ. Note that the maximum of legitimate backlogs on each interval increases linearly with time. We have that $\tau_i = \Theta(2^i)$ and $t \sim \sum_{k=1}^{i} 2^k = \Theta(2^i)$, which explains the linear relationship between B_i and t.

V. Performance of Specific Policies

In this section we study how specific legitimate and malicious policies interact.

A. Dynamic Legitimate - Static Malicious

A first order optimality criterion for a static malicious policy is for $\boldsymbol{\phi} \in \Pi_2$ to correspond to a max-flow achieving superflow on the network $\mathcal{G}(\hat{\mathcal{N}})$ (see proof of Lemma 4). A second order

criterion is to choose ϕ so that $\mathbf{a} \notin \Lambda_{\mathbf{b}}^{\phi}$. If $\mathbf{a}$ is not known to the attacker, then a plausible strategy is to solve

$$\min_{\phi \in \Pi_2} \text{Vol}(\Lambda_{\mathbf{b}}^{\phi})$$

where Vol is the volume of the set. This attempts to minimize the volume of the stability region $\Lambda_{\mathbf{b}}^{\phi}$.

For the legitimate policy, it is sufficient to statically balance the traffic on the remaining resource vector (r_n). If $\phi, \mathbf{a}$ are known, this can be done in an offline fashion using iterative water filling algorithms. Additionally, it can be done in an online fashion agnostically to $\phi, \mathbf{a}$ using a dynamic legitimate policy which we define below.

Definition 8. *(Join-the-Shortest Queue (JSQ) policy)* At any time t, let $n_l^\star = \arg\min_{n \in \mathcal{S}_l}\{X_n(t)\}$, $l \in \mathcal{L}$, ties broken arbitrarily. JSQ policy selects

$$f_{ln}^{\text{JSQ}}(t) = \begin{cases} a_l & \text{if } n = n_l^\star, \\ 0 & \text{otherwise.} \end{cases}$$

In words, the JSQ policy routes the traffic to the less loaded server. JSQ is known to achieve throughput optimality in the bipartite routing problem [7], in the absence of the malicious attacker. Also, it is known to have good load balancing properties [15], [19]. From [26] we have the following result.

Theorem 3. *(JSQ Optimality under static malicious policies)* For each static malicious policy ϕ, the JSQ policy stabilizes the system for all arrival rates $\mathbf{a}$ in the sustainable region $\Lambda_{\mathbf{b}}^{\phi}$.

The above implies that JSQ also achieves the guaranteed throughput region. However, the result is stronger in the following sense. If a suboptimal malicious policy ϕ is chosen so that $\Lambda_{\mathbf{b}}^{\phi} \supset \Lambda_{\mathbf{b}}$, then JSQ can improve the performance even further than the guaranteed throughput region. Formally, we say that JSQ is *maximally stable*, i.e., it is stable for all $\mathbf{a}, \phi$ which are stabilizable. *Thus, for a static attack, JSQ is always a desirable choice for the legitimate controller, since it is maximally stable and it operates agnostically to arrivals and malicious policy selection.* Next we show that JSQ is not maximally stable against dynamic attacks.

B. Dynamic Legitimate - Dynamic Malicious

In this subsection we consider a specific dynamic malicious policy called Join-the-Longest Legitimate Queue (JLLQ).

Definition 9. *(Join-the-Longest Legitimate Queue-JLLQ)* At any time t, let

$$n_m^\star = \arg\max_{n \in \mathcal{Q}_m}\{X_n^L(t)\}, m \in \mathcal{M},$$

ties broken arbitrarily. JLLQ policy selects

$$\phi_{mn}^{\text{JLLQ}}(t) = \begin{cases} b_m & \text{if } n = n_m^\star, \\ 0 & \text{otherwise.} \end{cases}$$

This policy is supported by the intuition that it targets the most loaded server considering only legitimate traffic. This way, it avoids overloading a server with no legitimate traffic, which from the point of view of the malicious controller is not beneficial. The use of JLLQ is motivated in cases where the optimal τ_i-DaR policy is either technologically infeasible or not desirable due to possible detection.

We show by examples and simulations that JLLQ has indeed a desirable property; *conditionally on JSQ being the legitimate policy, it destabilizes the system whenever the legitimate traffic intensity vector lies outside the guaranteed throughput region for static policies* Λ_b^{sta}. Arguably, this is a modest objective for the malicious controller, since the manipulation of dominated servers is conceded. However, even achieving this modest objective is not obvious when the arrival settings and the legitimate policy are not known and a τ_i-DaR-like policy is not available, or server capacities are larger than the attack resources.

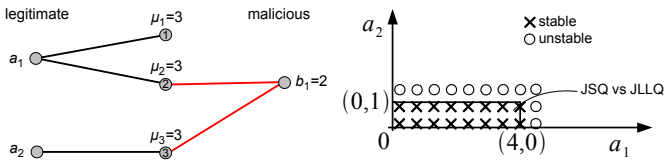

Fig. 7. (left) The system for Examples 3 and 4. (right) The guaranteed throughput region is depicted along with simulation experiments for chosen policies π=JSQ, ϕ=JLLQ. For the simulations we use $\circ$ for unstable points and $\times$ for stable points.

Conjecture 1. *(JLLQ vs JSQ)* If π = JSQ and σ = JLLQ, then the system is unstable iff

$$\mathbf{a} \notin \Lambda_{\mathbf{b}}^{\text{sta}}.$$

In words, if JSQ is in use, JLLQ destabilizes the system as long as the legitimate intensities lie outside the guaranteed throughput region for static malicious policies. To support the conjecture, we perform extensive simulations and present some in the following example.

Example 3 (JLLQ vs JSQ): Consider the example of Figure 7. Three servers with service rate 3 are fed by two legitimate and one malicious stream. The malicious stream can reach two servers ($\mathcal{Q} = \{2, 3\}$), while legitimate streams can reach $\mathcal{S}_1 = \{1, 2\}, \mathcal{S}_2 = \{3\}$ correspondingly. First, we use Theorem 1 to calculate the guaranteed throughput region under static policies Λ_b^{sta}. Note that due to $b < \mu_n$ for all n, we have $\Lambda_b^{\text{sta}} = \Lambda_b^{\text{dyn}}$. The region is characterized by $(a_1, a_2) \geq (0, 0)$, $a_1 \leq 4$ and $a_2 \leq 1$.

Next, we simulate the system for $t = 10000$, using JSQ for the legitimate controller and JLLQ for the malicious controller, examining different traffic intensity points. For each point, we consider the system to be stable if the average legitimate load of each server is less than 100 and use an x marker to denote it in the Figure. We use a $\circ$ marker if the system overcomes this threshold, in which case the system is considered unstable.

As observed by the comparison of simulations (markers) and the guaranteed regions (solid lines), whenever the legitimate traffic intensity vector lies inside the guaranteed throughput region, stability is achieved, i.e., JSQ achieves the guaranteed throughput region versus JLLQ. Additionally, whenever the vector lies outside the region, the system is unstable, which implies that JLLQ achieves the objective of destabilizing the system when outside the guaranteed throughput region for the static policies. Thus, the experimental observations support the conjecture. ■

Two questions remain: Is it true that $\Lambda_b^{\text{JLLQ}} = \Lambda_b^{\text{sta}}$? Is JSQ maximally stable? The answer is no to both questions given by the following example.

Example 4 (Tricking JLLQ): Consider again the setting of the previous example. Define the following legitimate policy:

Definition 10. *(Enhanced JSQ)* At time t, the Enhanced JSQ policy, assigns the job of stream l to a server according to the following algorithm:

1) If $l = 1$ compare $X_3(t) > \frac{1}{2}X_2(t)$
   - If true, then assign the job to server 2
   - else, assign the job to server $\arg\min_{n\in\mathcal{S}_1}\{X_n(t)\}$
2) If $l = 2$ assign the job to server 3.

The enhanced JSQ policy extends the classical JSQ by adding a condition; whenever $X_3(t)$ grows so that $X_3(t) > \frac{1}{2}X_2(t)$, the JSQ policy is overriden and arrivals from legitimate stream 1 are routed to server 2. This way, server 2 backlog grows faster than server's 3. Thus, JLLQ is lured to route more traffic to server 2.

By simulations, we find that the point (3,1.2), which lies outside the guaranteed throughput region Λ_b^{sta}, is stable when we use Enhanced JSQ vs JLLQ. ■

We conclude that JLLQ constrains the system to Λ_b^{sta} if operated vs JSQ, but cannot guarantee this performance vs any legitimate policy. *Moreover, JSQ is not maximally stable since it performs strictly worse in comparison to other policies.* Nevertheless, from simulations it appears that JSQ indeed achieves the guaranteed throughput region in all cases, i.e., it is the optimal defense versus an optimal attack.

Here, we should comment that the superiority of Enhanced JSQ to plain JSQ is symptomatic and related to fixing the JLLQ as the attacking policy. If a different attacking policy is used, the performance of Enhanced JSQ may be worse than JSQ. The interplay between the used policies gives rise to a game theoretic modeling of the problem, which is left for future work. We note that this direction is related to dynamic stochastic games. For example see [3] that analyzes dynamic games in the case of (i) adversary service, and (ii) individual user performance optimization.

VI. Conclusion

We derived necessary and sufficient conditions for the guaranteed throughput region for a system under a stealth DoS attack. This provides intuition as to how the network can be better designed to minimize the effect of these attacks. In case where the malicious controller uses a simple static routing policy, JSQ is proven to be a desirable defense policy. We show that the damage can be severe if the malicious controller performs non-stationary dynamic routing. Moreover, we exemplify the interaction between JSQ and JLLQ policies. It is found by simulations that JSQ is not a maximally stable policy in the sense that depending on the attacking policy it can be strictly outperformed by other legitimate policies.

The practical conclusions of this work are: (i) the impact of the attacks can be estimated using graph theoretic techniques (max-flow in the auxiliary network), (ii) pooling the resources allows load balancing and protects the system from attacks, (iii) in the dynamic setting, resource pooling can be effectively achieved by migrating load dynamically in order to prevent the servers from becoming dominated.

References

[1] The Risk vs. Cost of Enterprise DDoS Protection. Arbor Networks, White paper.

[2] Akamai. The state of the Internet. Technical report, 2013.

[3] E. Altman and N. Shimkin. Worst-case and nash routing policies in parallel queues with uncertain service allocations. *University of Minnesota*, IMA Preprint No. 1120, 1993.

[4] M. Bramson. Convergence to equilibria for fluid models of head-of-the-line proportional processor sharing queueing networks. *Queueing Systems*, 23(1-4):1–26, 1996.

[5] Q. Duan, H. Jafarian, E. Al-Shaer, and J. Xu. Modeling DDoS attacks by generalized minimum cut problems. Technical report, arXiv:1412.3359, 2014.

[6] M. Ficco and M. Rak. Intrusion Tolerance of Stealth DoS Attacks to Web Services. In *Information Security and Privacy Research*, volume 376 of *IFIP*, pages 579–584. Springer, 2012.

[7] R. D. Foley and D. R. McDonald. Join the Shortest Queue: stability and exact asymptotics. *Ann. Appl. Prob.*, 11(3):569–607, 2001.

[8] M. Guirguis, A. Bestavros, and I. Matta. Reduction of Quality (RoQ) Attacks on Dynamic Load Balancers: Vulnerability Assessment and Design Tradeoffs. In *INFOCOM*, pages 857–865, 2007.

[9] M. Guirguis, A. Bestavros, I. Matta, and Y. Zhang. Reduction of Quality (RoQ) Attacks on Internet End-Systems. In *INFOCOM*, pages 1362–1372, Mar. 2005.

[10] A. Hussain, J. Heidemann, and C. Papadopoulos. A Framework for Classifying Denial of Service Attacks. In *ACM Sigcomm*, pages 99–110, Aug. 2003.

[11] S. Kandula, D. Katabi, M. Jacob, and A. W. Berger. Botz-4-Sale: Surviving Organized DDoS Attacks That Mimic Flash Crowds. In *NSDI*, Boston, MA, May 2005.

[12] F. P. Kelly. Loss Networks. *Ann. Appl. Prob.*, 1(3):319–378, Aug. 1991.

[13] L. R. Ford Jr. and D. R. Fulkerson. Maximal flow through a network. In *Canadian Journal of Mathematics*, volume 8, pages 399–404, 1956.

[14] C. N. Laws. Resource Pooling in Queueing Networks with Dynamic Routing. *Advances in Applied Probability*, 24(3):699–726, Sept. 1992.

[15] C.-P. Li, G. S. Paschos, E. Modiano, and L. Tassiulas. Dynamic overload balancing in multi-server systems. *IFIP Networking*, 2014.

[16] H. Liu and M. S. Kim. Real-Time Detection of Stealthy DDoS Attacks Using Time-Series Decomposition. In *IEEE ICC*, pages 1–6, 2010.

[17] X. Luo and R. Chang. Optimizing the Pulsing Denial-of-Service Attacks. In *Proceedings of International Conference on Dependable Systems and Networks (DSN 2005)*, pages 582–591, June 2005.

[18] X. Luo and R. K. C. Chang. On a new class of Pulsing Denial-of-Service Attacks and the Defense. In *In Network and Distributed System Security Symposium (NDSS)*, pages 61–79, 2005.

[19] D. McDonald. Overloading Parallel Servers when Arrivals Join the Shortest Queue. *Lecture Notes in Statistics 117, Springer Verlag*, pages 169–196, 1996.

[20] J. Mirkovic and P. Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *SIGCOMM Comput. Commun. Rev.*, 34(2):39–53, Apr. 2004.

[21] G. S. Paschos and L. Tassiulas. Sustainability of service provisioning systems under attack. In *Proceedings of the ACM SIGMETRICS*, SIGMETRICS '13, pages 371–372, New York, NY, USA, 2013. ACM.

[22] T. Peng, C. Leckie, and K. Ramamohanarao. Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems. *ACM Comput. Surv.*, 39(1), Apr. 2007.

[23] Prolexic. Ddos attack report. Technical report, Prolexic Quarterly Global Report, Q4, 2013.

[24] C. A. Tais. General Analysis of the economy behind DDoS attacks. *Hyperion International Journal of Econophysics & New Economy*, 4(2):392–403, 2011.

[25] Y. Tang, X. Luo, Q. Hui, and R. Chang. On Generalized Low-rate Denial-of-Quality Attack against Internet Services. In *IWQoS, 17th International Workshop on Quality of Service*, pages 1–5, July 2009.

[26] L. Tassiulas. Adaptive Back-pressure Congestion Control Based on Local Information. *IEEE Transactions on Automatic Control*, 40(2):236–250, Feb. 1995.

[27] L. Tassiulas and A. Ephremides. Throughput properties of a Queueing Network with distributed dynamic routing and flow control. *Ann. Appl. Prob.*, 28:285–307, 1996.

[28] M. Zhang, Z. Zheng, and N. B. Shroff. A game theoretic model for defending against stealthy attacks with limited resources. *CoRR*, abs/1508.01950, 2015.

APPENDIX A
PROOF OF THEOREM 1

First we provide the proof of Lemma 2.

Proof of Lemma 2: Fix a malicious policy $\boldsymbol{\phi} \in \Pi_2$ and consider the corresponding induced *superflow* on network $\mathcal{G}(\hat{\mathcal{N}})$; on a link (u_m, v_n) set the flow to ϕ_{mn} and assume that flow conservation is satisfied at all nodes of $\mathcal{G}(\hat{\mathcal{N}})$. The capacity constraints are clearly satisfied on infinite capacity links (u_m, v_n), and additionally satisfied on links (s, u_m), since $\sum_{n\in\hat{\mathcal{N}}} \phi_{mn} \leq b_m$. We call it a superflow since the capacity constraint might be violated on links (v_n, t), i.e. we might have $\sum_m \phi_{mn} > \mu_n$ for some n. The throughput of the superflow induced by policy $\boldsymbol{\phi}$ is $\sum_{n\in\hat{\mathcal{N}}} \min\{\mu_n, \sum_m \phi_{mn}\}$ and the maximum achievable throughput by all superflows is equal to the maximum flow:

$$\begin{aligned} M_{\max}(\hat{\mathcal{N}}) &= \max_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} \min\{\mu_n, \sum_m \phi_{mn}\} \\ &= \max_{\boldsymbol{\phi}\in\Pi_2} \left[\sum_{n\in\hat{\mathcal{N}}} \mu_n - \left(\mu_n - \sum_m \phi_{mn}\right)^+\right] \\ &= \max_{\boldsymbol{\phi}\in\Pi_2} \left[\sum_{n\in\hat{\mathcal{N}}} \mu_n - r_n(\boldsymbol{\phi})\right] \\ &= \sum_{n\in\hat{\mathcal{N}}} \mu_n - \min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}). \end{aligned}$$

■

Proof of Theorem 1 (Necessity): In particular, we will show that if one of the conditions does not hold, then the system is unstable. Note that if for some subset $\hat{\mathcal{N}}$ we have $\hat{\mathcal{L}} = \emptyset$, then the LHS of (3) is zero and the condition is automatically satisfied for this subset. This is because $\sum_{n\in\hat{\mathcal{N}}} \mu_n$ is an s-t cut of network $\mathcal{G}(\hat{\mathcal{N}})$ and thus larger or equal to the maximum s-t flow $M_{\max}(\hat{\mathcal{N}})$. Hence by Lemma 2, the RHS is nonnegative.

Next, pick a subset $\hat{\mathcal{N}}$ with $|\hat{\mathcal{L}}| > 0$ and let

$$\sum_{l\in\hat{\mathcal{L}}} a_l > \min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}). \tag{8}$$

Let $\boldsymbol{\phi}^\star$ be the minimizer of the RHS of (8), it follows

$$\sum_{l\in\hat{\mathcal{L}}} a_l > \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}^\star).$$

and thus by Lemma 1 the system is unstable. ■

Proof of Theorem 1 (Sufficiency): Fix any malicious policy $\bar{\boldsymbol{\phi}} \in \Pi_2$. We have for any $\hat{\mathcal{N}}$

$$\min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}} r_n(\boldsymbol{\phi}) \leq \sum_{n\in\hat{\mathcal{N}}} r_n(\bar{\boldsymbol{\phi}}).$$

Thus, our conditions imply

$$\sum_{l\in\hat{\mathcal{L}}} a_l \leq \sum_{n\in\hat{\mathcal{N}}} r_n(\bar{\boldsymbol{\phi}}), \qquad \text{for all } \hat{\mathcal{N}}$$

which by Lemma 1 is sufficient for stability. ■

APPENDIX B
PROOF OF LEMMAS 3 AND 4

Proof of Lemma 3: We express the parameters t_i^j, d_i, r_i in terms of of t_i^1 first, and then yield the expression of t_i^1 in terms of τ_i. First, we can express the duration of the interval j as a function of the duration of the first interval using (6):

$$t_i^j = \frac{\prod_{k=1}^{j-1} \rho_k}{(\rho_{\max} + \mu_{\max})^{j-1}} t_i^1. \tag{9}$$

It follows that the duration of the total i^{th} dominate phase is given by

$$d_i = c_1 t_i^1,$$

where $c_1 \triangleq 1 + \sum_{j=2}^{J} \frac{\prod_{k=1}^{j-1} \rho_k}{(\rho_{\max}+\mu_{\max})^{j-1}}$. The duration of the i^{th} release phase is given by

$$r_i = \frac{B_i}{\mu_{\max}} = \frac{t_i^J \rho_J}{\mu_{\max}} = c_2 t_i^1, \tag{10}$$

where $c_2 \triangleq \frac{\prod_{k=1}^{J} \rho_k}{(\rho_{\max}+\mu_{\max})^{J-1}\mu_{\max}}$.

Since it must be $\tau_i = d_i + r_i$, we have

$$t_i^1 = \tau_i \left(\frac{1}{c_1 + c_2}\right), \tag{11}$$

where c_1 and c_2 are positive constants given above. Note that these constants depend on the chosen order of dominated servers. Concluding, (a) is obtained directly from (11), (b) is obtained by noticing that from (10) we have $B_i = c_2 \mu_{\max}$, and (c) is obtained from $r_i = B_i/\mu_{max}$. ■

Proof of Lemma 4: Pick an interval of duration $\xi(\tau_i)$ within the period i, such that its starting time instance coincides with the end of the i^{th} dominate phase/beginning of the i^{th} release phase, denoted with t_{end}^i. Since $\xi(\tau_i)$ overlaps with the release phase, a constraint we must always satisfy is that $\xi(\tau_i) \leq r_i$ which is equivalent to

$$\tau_i \geq \frac{\xi(\tau_i)}{C_3} = \frac{\mu_{\max}\xi(\tau_i)}{C_2}. \tag{12}$$

Also, we want to impose an additional constraint, such that at the end of the interval $\xi(\tau_i)$ all servers have total backlog at least as large as $B_i/2$. Since at the beginning of the interval $\xi(\tau_i)$ all backlogs are at least B_i, it is sufficient to choose $\mu_{\max}\xi(\tau_i) \leq B_i/2$, where $\mu_{\max}$ is an upper bound on the faster draining rate across all servers. This can be achieved by ensuring

$$\tau_i = \frac{B_i}{C_2} \geq \frac{2\mu_{\max}\xi(\tau_i)}{C_2}. \tag{13}$$

It is enough to consider only constraint (13) since if (13) is satisfied then so is (12).

Now, assume that during the whole $\xi(\tau_i)$ interval the legitimate backlogs are always smaller than D, i.e. $X_n^L(t) < D$ for all $n \in \mathcal{N}_d$ and t in the interval $[t_{\text{end}}^i, t_{\text{end}}^i + \xi(\tau_i)]$. We deduce that the maximum service rate for the legitimate traffic in all dominated servers can be bounded within the interval $\xi(\tau_i)$ as follows

$$\begin{aligned} \sum_{n\in\mathcal{N}_d} \mu_n^L(t) &= \sum_{n\in\mathcal{N}_d} \frac{X_n^L(t)}{X_n(t)} \mu_n \overset{(13)}{\leq} \sum_{n\in\mathcal{N}_d} \frac{X_n^L(t)}{B_i/2} \mu_n \\ &< \sum_{n\in\mathcal{N}_d} \frac{D}{B_i/2} \mu_{\max} = \frac{|\mathcal{N}_d| D}{B_i/2} \mu_{\max} \\ &= \frac{2|\mathcal{N}_d|\mu_{\max}}{C_2\tau_i} D, \quad \forall\, t \in [t_{\text{end}}^i, t_{\text{end}}^i + \xi(\tau_i)]. \end{aligned}$$

Let $A(\mathcal{N}_d)$ be the total legitimate traffic routed to the dominated servers within the interval $\xi(\tau_i)$. We conclude that the legitimate backlogs at the end of the interval are

$$\sum_{n\in\mathcal{N}_d} X_n^L(t_{\text{end}}^i+\xi(\tau_i)) > A(\mathcal{N}_d) - \xi(\tau_i)\frac{2|\mathcal{N}_d|\mu_{\max}}{C_2\tau_i}D \overset{(13)}{\geq} A(\mathcal{N}_d) - |\mathcal{N}_d|D. \quad (14)$$

Combining (14) with $X_n^L(t) < D$, we conclude $A(\mathcal{N}_d) < 2|\mathcal{N}_d|D$.

To prove independence from ordering, observe that to satisfy (12)-(13) it is sufficient to choose $\xi(\tau_i) = \frac{C_2}{2\mu_{\max}}\tau_i$, where C_2 depends on the order of dominated servers considered. Let $C_2^* > 0$ be the minimum C_2 over all the possible orderings (they are finitely many). Then a global selection $\xi(\tau_i) = \frac{C_2^*}{2\mu_{\max}}\tau_i$ satisfies (13)-(12) for all orderings while at the same time $\xi(\tau_i)$ is proportional to τ_i. ∎

Appendix C
Proof of Theorem 2

Proof of Theorem 2 (Necessity): We will show the necessity of the conditions by showing that for any arrival vector such that for some subset $\hat{\mathcal{N}}$ the inequality in (7) is violated, the malicious controller can use a DaR policy with a $\boldsymbol{\phi}^f$ that depends on the subset $\hat{\mathcal{N}}$ to cause the legitimate backlog to overflow D on some server.

First, observe that for $\mathcal{N}_d = \emptyset$, the conditions reduce to Theorem 1 and the DaR policy reduces to a static attack. We can follow the same methodology with the proof of Theorem 1 to show the necessity. Thus, below we deal with the case $\mathcal{N}_d \neq \emptyset$.

Fix a subset of server $\hat{\mathcal{N}}$, such that inequality (7) is not satisfied, i.e. it is

$$\sum_{l\in\hat{\mathcal{L}}} a_l > \min_{\boldsymbol{\phi}\in\Pi_2} \sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d} r_n(\boldsymbol{\phi}). \quad (15)$$

It follows that there exists a static malicious routing $\boldsymbol{\phi}^\star$ and a positive ϵ such that

$$\sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d} r_n(\boldsymbol{\phi}^\star) = \sum_{l\in\hat{\mathcal{L}}} a_l - \epsilon. \quad (16)$$

Next we choose a DaR with $\boldsymbol{\phi}^f = \boldsymbol{\phi}^\star$. Since the free (non-dominated) servers have strictly positive remaining resource for any static attack policy, it is also

$$\sum_m \phi_{mn}^\star \leq \mu_n, \quad \text{for all } n \in \hat{\mathcal{N}}\setminus\mathcal{N}_d, \quad (17)$$

thus combining with (16) we get

$$\sum_{l\in\hat{\mathcal{L}}} a_l + \sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d}\left[\sum_m \phi_{mn}^\star - \mu_n\right] = \epsilon. \quad (18)$$

Pick any arbitrary finite D, focus on the time interval $I_i \triangleq [t_{\text{end}}^i, t_{\text{end}}^i + \xi(\tau_i)]$ where $\xi(\tau_i)$ is chosen to be a linear function of τ_i as in Lemma 4, and we will choose τ_i later. By definition of $\hat{\mathcal{L}}$, the total legitimate traffic generated during I_i that must be routed to $\hat{\mathcal{N}}$ is exactly $A(\hat{\mathcal{N}}) = \xi(\tau_i)\sum_{l\in\hat{\mathcal{L}}} a_l$. If there are no free servers, then the instability is readily obtained by picking a large $\xi(\tau_i)$ that exceeds the condition of Lemma 4. Thus in the remaining we consider only the case $\mathcal{N}_f \supset \emptyset$. We assume $\sup_{t\in I_i} X_n^L(t) < D$ for all n, t and provide the proof by contradiction.

By Lemma 4, some of the dominated servers $\hat{\mathcal{N}}\cap\mathcal{N}_d$ will overflow D if within I_i we have $A(\mathcal{N}_d) \geq 2|\mathcal{N}_d|D$, in which case we reach a contradiction by the overflow of a dominated server. Therefore, we continue assuming otherwise, and the total legitimate traffic that is routed to free servers $\hat{\mathcal{N}}\setminus\mathcal{N}_d$ during the same interval is

$$A(\hat{\mathcal{N}}\setminus\mathcal{N}_d) \geq \xi(\tau_i)\sum_{l\in\hat{\mathcal{L}}} a_l - 2|\mathcal{N}_d|D. \quad (19)$$

Considering the total backlog evolution in I_i, we get a lower bound on the sum of total backlog of free servers at the end of the interval using (18)-(19)

$$\begin{aligned}
&\sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d} X_n(t_{\text{end}}^i+\xi(\tau_i)) \\
&\geq A(\hat{\mathcal{N}}\setminus\mathcal{N}_d) + \xi(\tau_i)\sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d}\sum_m \phi_{mn}^f - \xi(\tau_i)\sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d}\mu_n \\
&\overset{(19)}{\geq} \xi(\tau_i)\left(\sum_{l\in\hat{\mathcal{L}}} a_l + \sum_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d}[\sum_m \phi_{mn}^f - \mu_n]\right) - 2|\mathcal{N}_d|D \\
&\overset{(18)}{=} \xi(\tau_i)\epsilon - 2|\mathcal{N}_d|D.
\end{aligned}$$

By the pigeonhole principle this implies a lower bound on the most loaded free server's total backlog at the end of the interval I_i

$$\max_{n\in\hat{\mathcal{N}}\setminus\mathcal{N}_d} X_n(t_{\text{end}}^i+\xi(\tau_i)) \geq \frac{\epsilon\xi(\tau_i)}{|\hat{\mathcal{N}}\setminus\mathcal{N}_d|} - 2|\mathcal{N}_d|D, \quad (20)$$

which can be summarized in the asymptotic statement

$$X_{\bar{n}}^M(t_{\text{end}}^i+\xi(\tau_i)) \in \Omega(\tau_i), \quad \text{for } \bar{n} \text{ maximizer of (20).} \quad (21)$$

By the assumption $X_n^L(t) < D$, we also obtain a lower bound on the malicious service rate for all servers n

$$\mu_n^M(t) = \frac{X_n^M(t)}{X_n^L(t)+X_n^M(t)}\mu_n \geq \mu_n - \frac{D\mu_n}{X_n(t)}, \quad \forall n, t \in I_i \quad (22)$$

In the remaining of the proof focus on a free server that maximizes (20), denoted with $\bar{n}$. Choose

$$t_0 = \inf\{\tau \in I_i : X_{\bar{n}}^M(t) > 0,\ \forall t > \tau, t \in I_i\},$$

where by convention we set $t_0 = t_{\text{end}}^i + \xi(\tau_i)$ if $X_{\bar{n}}^M(t_{\text{end}}^i + \xi(\tau_i)) = 0$. If $t_0 = t_{\text{end}}^i + \xi(\tau_i)$, then (20) is contradicted for any $\xi(\tau_i)$ large enough to make the RHS positive. Else for $t \in [t_0, t_{\text{end}}^i + \xi(\tau_i))$, since $X_n^M(t) > 0$ the malicious backlog of server n is governed by the following differential equation

$$\dot{X}_n^M(t) = \sum_m \phi_{mn}^f - \mu_n^M(t), \quad t \in [t_0, t_{\text{end}}^i + \xi(\tau_i)]$$

Combining with (22), taking into account $X_n(t) \geq X_n^M(t) > 0$ we have

$$\dot{X}_n^M(t) \leq \sum_m \phi_{mn}^f - \mu_n + \frac{D\mu_n}{X_n(t)} \leq \frac{D\mu_n}{X_n^M(t)}, \quad t \in [t_0, t_{\text{end}}^i + \xi(\tau_i)] \quad (23)$$

where in the last inequality we used (17) and $X_n(t) \geq X_n^M(t)$. Using separation of variables and positivity of $X_n^M(t)$, we solve the differential inequality $\dot{X}_n^M(t)X_n^M(t) \leq D\mu_n$ in the interval $[t_0, t_{\text{end}}^i + \xi(\tau_i)]$ and obtain an upper bound for the malicious backlog

$$X_n^M(t_{\text{end}}^i+\xi(\tau_i)) \in O(\sqrt{\tau_i})$$

which contradicts the law $X_n^M(t_{\text{end}}^i+\xi(\tau_i)) \in \Omega(\tau_i)$ obtained above in (21). This in turn completes the contradiction since we can choose τ_i arbitrarily large. ∎

Proof of Theorem 2 (Sufficiency): To show the sufficiency of the conditions, we will provide a legitimate policy which uses only the free servers and stabilizes the system whenever the arrival vector $\mathbf{a}$ satisfies the necessary conditions.

Assume a vector $\mathbf{a}$ that satisfies the conditions and consider a legitimate policy which at time t inspects the malicious decisions $\phi^{\sigma}_{mn}(t)$ and calculates the legitimate decisions $f^{\pi}_{ln}(t)$ so the following holds for all t

$$\sum_{l\in\mathcal{L}} f^{\pi}_{ln}(t) + \sum_{m\in\mathcal{M}} \phi^{\sigma}_{mn}(t) \leq \mu_n, \quad \text{for all } n\in\mathcal{N}_f \qquad (24)$$

and $f^{\pi}_{ln}(t) = 0,$ for all $n \in \mathcal{N}_d$.

To see why π exists, note first that by definition all free servers satisfy

$$\sum_{m\in\mathcal{M}} \phi^{\sigma}_{mn}(t) \leq \mu_n, \quad \forall \sigma, t, n \in \mathcal{N}_f$$

and thus at each time instance

$$r_n(\boldsymbol{\phi}^{\sigma}) = \left(\mu_n - \sum_{m\in\mathcal{M}} \phi^{\sigma}_{mn}(t)\right)^{+}$$
$$= \mu_n - \sum_{m\in\mathcal{M}} \phi^{\sigma}_{mn}(t) \geq 0, \quad \forall n \in \mathcal{N}_f.$$

Then, fix t and use the sufficiency of Theorem 1, eq. (3) to conclude that given $\boldsymbol{\phi}^{\sigma}$, there exist $f^{\pi}_{ln}(t)$ such that (24) holds. The same is true for any t. Thus, the service rate can always be made larger or equal to the total arrival rate. Then, stability of free servers follows by picking a large enough D. Dominated servers are stable by the definition of the legitimate policy, which does not route any traffic to them. ■

Georgios S. Paschos (S'01-M'06-SM'15) received his dipl. in Electrical and Computer Engineering '02 from Aristotle University of Thessaloniki, and Ph.D. degree '06 from ECE dept. University of Patras, both in Greece. He was an ERCIM postdoc fellow for one year in VTT, Finland. From '08 to '14, he was affiliated with The Center of Research and Technology Hellas–Informatics & Telematics Institute, CERTH-ITI, Greece. He was also teaching as an adjunct lecturer in the Department of Computer and Communication Engineering of the University of Thessaly, for the period '09-'11. From '12 to '14, he was a postdoctoral associate at LIDS, Massachusetts Institute of Technology, USA. Since '14, he is a Principal Researcher in the French Research Center of Huawei Technologies, Paris, and the head of the Network Control and Resource Allocation team. His main interests are in the area of wireless communications, networks and stochastic modeling. He has served as a Technical Program Committee member in INFOCOM, Mobihoc, and WiOPT conferences. Since Sep '15 He serves as an Associate Editor for the IEEE/ACM Trans. on Networking.

Leandros Tassiulas (S'89-M'91-SM'05-F'07) is the John C. Malone Professor of Electrical Engineering at Yale University. His research interests are in the field of computer and communication networks with emphasis on fundamental mathematical models and algorithms of complex networks, architectures and protocols of wireless systems, sensor networks, novel internet architectures and experimental platforms for network research. His most notable contributions include the max-weight scheduling algorithm and the back-pressure network control policy, opportunistic scheduling in wireless, the maximum lifetime approach for wireless network energy management, and the consideration of joint access control and antenna transmission management in multiple antenna wireless systems. Dr. Tassiulas is a Fellow of IEEE (2007). His research has been recognized by several awards including the IEEE Koji Kobayashi computer and communications award, the inaugural INFOCOM 2007 Achievement Award "for fundamental contributions to resource allocation in communication networks," the INFOCOM 1994 best paper award, a National Science Foundation (NSF) Research Initiation Award (1992), an NSF CAREER Award (1995), an Office of Naval Research Young Investigator Award (1997) and a Bodossaki Foundation award (1999). He holds a Ph.D. in Electrical Engineering from the University of Maryland, College Park (1991). He has held faculty positions at Polytechnic University, New York, University of Maryland, College Park, and University of Thessaly, Greece.