A Novel Framework for Modeling and Mitigating Distributed Link Flooding Attacks

Christos Liaskos[1], Vasileios Kotronis[2] and Xenofontas Dimitropoulos[1]
[1]FORTH, Greece [2]ETH Zurich, Switzerland
Emails: {cliaskos, fontas}@ics.forth.gr, vkotroni@tik.ee.ethz.ch

***Abstract*—Distributed link-flooding attacks constitute a new class of attacks with the potential to segment large areas of the Internet. Their distributed nature makes detection and mitigation very hard. This work proposes a novel framework for the analytical modeling and optimal mitigation of such attacks. The detection is modeled as a problem of relational algebra, representing the association of potential attackers (bots) to potential targets. The analysis seeks to optimally dissolve all but the malevolent associations. The framework is implemented at the level of online Traffic Engineering (TE), which is naturally triggered on link-flooding events. The key idea is to continuously re-route traffic in a manner that makes persistent participation to link-flooding events highly improbable for any benign source. Thus, bots are forced to adopt a suspicious behavior to remain effective, revealing their presence. The load-balancing objective of TE is not affected at all. Extensive simulations on various topologies validate our analytical findings.**

***Index Terms*—DDoS, link-flooding, analysis.**

I. Introduction

DDoS attacks are well-known within the Internet community. For example, the attack against Spamhaus in 2013 was a powerful attack that inflicted more than 300 Gbit/s of malicious traffic upon the intended target [1]. However, researchers have recently shed light on new types of link-flooding attacks that can operate at Internet scales and are extremely difficult to detect and mitigate [2], [3]. The objective of these attacks is to deplete the bandwidth of certain network links, disconnecting entire domains—even countries—from the Internet. In particular, the Crossfire [2] attack is a stealthy and effective DDoS link-flooding attack that uses bots with non-spoofed IP addresses to send traffic to publicly accessible servers. While packets stemming from these attack sources are seemingly legitimate, their cumulative volume harms the intended victim in an indirect way by flooding links and cutting off connectivity towards its location. Each bot-to-server flow usually has very low bandwidth, and is thus very hard to detect and filter.

On the defender's side, Traffic Engineering (TE) is the network process that reacts to link-flooding events, regardless of their cause [4]. The TE module, hosting this process, is thus a natural point to incorporate attack detection and mitigation mechanisms. A TE process has two phases: i) the optimal load calculation for each network path, and ii) the mapping of specific traffic flows to paths in a manner upholding their calculated optimal load [5]. The first phase represents the first TE priority, which is to ensure that the network load is balanced, i.e., fairly distributed over all available paths. The flow mapping phase has received limited focus in existing TE solutions and is even random for all but elephant flows [6].

In the present paper the TE process is optimized for attack detection, without altering its load-balancing objective. The *methodology* consists in optimizing the flow mapping phase of TE. The new optimal mapping ensures that a benign flow will have the lowest probability of contributing to a future attack by chance. Therefore, bots are forced to behave improbably over time in order to remain effective. Internally, a novel analytical framework based on relational algebra is employed to relate bots to susceptible targets. In this aspect, the outlined TE flow mapping maximizes the support of bot-to-target relations, accentuating their presence over time.

To the best of our knowledge, this paper *contributes* the first analytical model targeting Crossfire-like link-flooding attacks, offering novel insights in this attack type. In addition, the presented analysis and algorithm have general applicability to multigraphs, multipath routing and generic bot behavior. Multiple time-varying and mixed malicious/benign connections per single bot are allowed. Furthermore, the analysis offers insights in the topological attributes that affect the attack and its mitigation. Moreover, the integration with existing TE modules is seamless. Finally, given that the analysis is built upon relational algebra principles, the proposed scheme lends itself to a straightforward implementation based on well-known, mature and scalable databases (e.g., SQL [7]).

The remainder of this paper is organized as follows. A description of the studied attack is given in Section II. The proposed analytical framework is described in Section III, while its algorithmic formulation is given in Section IV. The simulation results follow in Section V, while the related work is presented in Section VI. Finally, we conclude in Section VII.

II. Attack Model: Reactive Crossfire

The attack that we study as a use case is a reactive version of the Crossfire attack. In the classic Crossfire attack, the attacker has a swarm of bots (or botnet) at his disposal, and seeks to attack a certain area, called the *target area*. The goal is to cut off Internet connectivity to this area. To achieve this, he assigns his bots to send legitimate, low-rate traffic flows towards certain public servers, the *decoy servers*. These servers are reached over the same *target links* that connect the target area to the Internet. Thus the bots send traffic along paths that lead to both

This work was funded by the European Research Council via Grant Agreement no. 338402, project "NetVolution: Evolving Internet Routing".

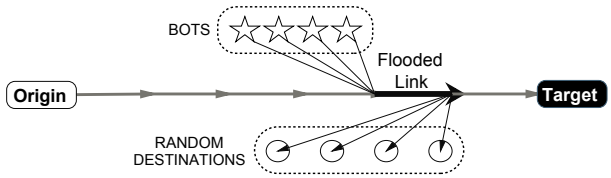


Figure 1: Overview of the Crossfire attack concept.

the decoys and the target, cumulatively flooding the shared target links. Traffic is sent only to the decoys, so that the target cannot directly observe the flood. The knowledge base of the attacker is mainly the *link-map*, i.e., the map of the links of the victim network. The most loaded links along the paths from the bots to the servers and from the bots to the target are flooded. The concept is illustrated in Fig. 1; the decoys can be random destinations that are on the appropriate paths to the target. As an extension, we also assume that the attacker monitors the network routes and reacts to any changes in the link-map, including shifts of load and routing changes. These changes may be possible defender's reactions; the attacker thus reiterates this process to harm the target anew.

While the attacker seeks to flood the target links, the defender aims at alleviating the load from the flooded connections and at finding and blocking any malicious traffic sources. While the first objective is the same as the one employed by most modern TE modules (i.e., balancing the load), the second one is much harder to implement. In our framework, we thus assume the following features for the defender's process. First, he monitors the network load and reacts to link-flooding events. Flood detection can be based on approaches such as the one from Xue et al. [8]. After the flood is known, the defender balances the load by re-routing traffic destined to different destinations, without though knowing the attacker's classification (target, decoys, benign servers). Moreover, he records sources that are consistently present in link-flooding events, even after re-routing. Sources that change their destination selection to adapt to re-routing are particularly suspicious; that means that re-routing has diverted their initial load away from the target link(s), while they want to return and inflict damage. The idea is that after such an interaction cycle between the attacker and the defender, attack sources may become more identifiable by exhibiting a behavior that is highly improbable for benign sources. For example, benign (e.g., flash-crowd) load would not re-adjust to routing changes, but it would use the same popular destination(s) as before.

The defender must gradually collect evidence to support the involvement of traffic sources in an attack towards a target, while dealing with the proper allocation of bandwidth during the attack via TE. The attacker's classification of the target and decoys, as well as the attacking bots should become more and more visible after a number of attack/re-route interactions. We substantiate this model analytically in Section III.

III. Analysis

Below we formulate the detection problem via relational algebra and study the effects of the attacker-defender interplay. Then, the defender's actions are analytically optimized.

Prerequisites. Assume a directed multigraph $G(\mathcal{N}, \mathcal{L})$ comprising a set of nodes $\mathcal{N}$ and a set of directed links $\mathcal{L}$. A single node is denoted as $n \in \mathcal{N}$ and a single link as $l \in \mathcal{L}$. Each node n logically hosts a set of network-wide unique entities, $\mathcal{E}^{(n)}$. An entity e represents an IP address or an IP prefix, depending on the required level of detection granularity. Each $\mathcal{E}^{(n)}$ set is connectable to any other entity $e \in \mathcal{E}^{(n^*)}$, $n \neq n^*$; inter-entity paths are based on the current set of routing tables $\mathcal{T}$ of G, i.e., the routing configuration.

The attack detection process will eventually relate a set of entities (e.g., botnet IP addresses) to a set of nodes (e.g., decoy servers). Therefore, we denote the associative relation, r, between two generic sets S_1 and S_2 as:

$$r : S_1 \xrightarrow{s} S_2 \tag{1}$$

where $s \in \mathbf{R}^+$ is the support of the relation. In general, the $\longrightarrow$ notation denotes ordered collocations of S_1 and S_2 elements within a stream of 2-tuple observations. The support s essentially counts these collocations within the stream.

Let $\|S\|$ denote the cardinality of a set S. We define the $L-specificity$ of relation (1) as $\|S_1\|$, and its $R-specificity$ as $\|S_2\|$. By definition, the following holds:

$$r_1 \supseteq r_2 \Leftrightarrow (S_1 \rightarrow S_2) \supseteq (S_3 \rightarrow S_4) \Rightarrow \begin{cases} S_1 \supseteq S_3 \\ S_2 \supseteq S_4 \end{cases} \tag{2}$$

A. Problem Formulation via Associative Relations

The analysis assumes a cycle of attacks followed by defense actions. We study only the time moments when links have been flooded; these moments are denoted as sequential time steps $t \in \mathbf{N}$. In addition, the described process may *optionally* assume the existence of heuristics that classify link floods as malevolent, filtering out natural causes like flash crowds [9].

An attack at time step t floods a set of links $\mathcal{L}_{\mathcal{A}}(t) \subset \mathcal{L}$, affecting the connectivity of a set of nodes $\mathcal{N}_{\mathcal{A}}(t) \subset \mathcal{N}$. Let $\mathcal{E}_{\mathcal{A}}(t)$ denote the set of all *suspicious* entities, defined as the ones that are origins of traffic flows present in $\mathcal{L}_{\mathcal{A}}(t)$ which did not exist at time $t-1$. In other words, $\mathcal{E}_{\mathcal{A}}(t)$ contains the entities that have launched new flows to be present in one or more flooded links at time step t. Thus, for all time steps up to t, we form the following associative relation:

$$r_{\mathcal{A}}(t) : \underset{\forall t}{\cup}\mathcal{E}_{\mathcal{A}}(t) \longrightarrow \underset{\forall t}{\cup}\mathcal{N}_{\mathcal{A}}(t) \tag{3}$$

The goal of the defender at time step t is to deploy a new set of routing tables effective until $t+1$, i.e., $\mathcal{T}(t+1)$, such that:

$$min\{s\}\ \forall r : e \xrightarrow{s} n,\ r \subset r_{\mathcal{A}}(t) \tag{4}$$

The defender assumes that all entities $e \in \underset{\forall t}{\cup}\mathcal{E}_{\mathcal{A}}(t)$ are benevolent, i.e., flows originating from an entity $e \in \mathcal{E}_{\mathcal{A}}(t)$ will not change their destination to affect n at time step $t+1$, to the extent justified statistically by normal traffic patterns. Thus he seeks to minimize the corresponding support to match this assumption. In essence, the formulation of relation (4) describes the construction of new routing tables, $\mathcal{T}(t+1)$, that disrelate entities from nodes affected by link-flooding events. The new $\mathcal{T}(t+1)$ assumes that all past $e \rightarrow n$ relations were

coincidental and, therefore, their support should decrease with high probability in the future. Relations that persist regardless of this effort are treated as indications of an attack.

Definition of Support. At any given time step t, the defender observes a set of $\mathcal{L}_{\mathcal{A}}(t)$ flooded links. With no loss of generality, let $\mathcal{L}_{\mathcal{A}}(t) = \{l_i,\, i = 1 \dots \|\mathcal{L}_{\mathcal{A}}\|\}$. Let $\mathcal{E}_i$ be the set of suspicious entities over link l_i. We form the relations:

$$\{r_i(t) : \mathcal{E}_i \rightarrow l_i\},\, i = 1 \dots \|\mathcal{L}_{\mathcal{A}}\| \tag{5}$$

Moreover, let $\mathcal{N}_i$ denote the set of nodes which may receive traffic via link l_i according to $\mathcal{T}(t)$, either as transit nodes or as flow destinations. We express these relations as:

$$\left\{r_i^{'}(t) : l_i \rightarrow \mathcal{N}_i\right\},\, i = 1 \dots \|\mathcal{L}_{\mathcal{A}}\| \tag{6}$$

Relations (5) and (6) can be combined by eliminating l_i:

$$\left\{r_i^{''}(t) : \mathcal{E}_i \rightarrow \mathcal{N}_i\right\},\, i = 1 \dots \|\mathcal{L}_{\mathcal{A}}\| \tag{7}$$

Remark 1. The set $r_i^{''}(t)$ is used for filtering out links from $\mathcal{L}_{\mathcal{A}}(t)$ that carry the same amount of information in terms of attack detection. An attack may flood several links that are sequential on a path, due to link capacity or link load variations. In these cases, the detection process should consider the most general of the $r_i^{''}(t)$ relations only, in order not to lose information that is important to the attack detection process. That is, links that are "shadowed" by other links on a path and do not offer new insight should be left out. Therefore, the remainder of the analysis will assume that all links l_i for which $\exists j : r_j^{''}(t) \supseteq r_i^{''}(t)$ have been filtered out of $\mathcal{L}_{\mathcal{A}}(t)$.

For each distinct $e \in \bigcup_{\forall i} \mathcal{E}_i$, (5) can be rewritten as:

$$e \xrightarrow{s(e)} \mathcal{L}_e(t) = \{l_i : e \in \mathcal{E}_i\},\, s(e) = \frac{\|\mathcal{L}_e(t)\|}{\|\mathcal{L}_{\mathcal{A}}(t)\|} \tag{8}$$

In a similar fashion, for each distinct $n \in \bigcup_{\forall i} \mathcal{N}_i$, (6) yields:

$$n \xrightarrow{s(n)} \mathcal{L}_n(t) = \{l_i : n \in \mathcal{N}_i\},\, s(n) = \frac{\|\mathcal{L}_n(t)\|}{\|\mathcal{L}_{\mathcal{A}}(t)\|} \tag{9}$$

Combining (8) and (9) produces the specific relation:

$$e \xrightarrow{s(e,n)} n,\, s(e,n) = \frac{\|\mathcal{L}_e(t) \cap \mathcal{L}_n(t)\|}{\|\mathcal{L}_e(t) \cup \mathcal{L}_n(t)\| - \|\mathcal{L}_e(t) \cap \mathcal{L}_n(t)\|}. \tag{10}$$

Notice that $s(e)$ expresses the probability of e acting as a bot at time t, $s(n)$ is the probability of n being an attack target at time t, and $s(e,n)$ is the probability of e attacking n at time t. The optimization criterion (4) thus seeks to minimize the *total* support of relations (8-10) *over all time steps*, i.e., the probabilities $\Pi_{\forall t} s_t(e)$, $\Pi_{\forall t} s_t(n)$ and $\Pi_{\forall t} s_t(e,n)$.

B. Effects of Attacker's Actions on the Detection Process

A smart attacker may consider that he may be tracked. Thus, he needs to take measures to obfuscate his presence. The existence of a bot entity may be obfuscated by:

1) Not participating to an attack at every step t.
2) Attacking targets beyond the intended ones.

These approaches affect the L and R specificity of the relations (10), as shown in the following Lemmas.

Lemma 2. *Limited entity participation to an attack reduces the L-specificity of a relation* $e \xrightarrow{s(e,n)} n$.

Proof: Assume an attack towards node n, consistently launched by a an entity e over time steps 0 to t, leading to the relation $e \xrightarrow{s(e,n)} n$. Assume that the same attack is now launched by a multiset of entities, $\mathcal{E}$, with $\|\mathcal{E}\| = t$, where an entity $e \in \mathcal{E}$ is allowed to attack at one time step only. Any cycle of size $\|\mathcal{E}\|$ can be used to deduce the exact order. The updated relation is now $\mathcal{E} \xrightarrow{s(e,n)} n$, yielding the same support, but a decreased L-specificity of $\|\mathcal{E}\| > 1$. ■

Lemma 3. *Sporadically attacking targets beyond the intended one reduces the R-specificity of a relation* $e \xrightarrow{s(e,n)} n$.

Proof: Similar to Lemma 2, using a $\mathcal{N}$ multiset. ■

Attacks employing Lemmas 2 and 3 can also be combined to yield relations with L and R specificity that is too low to be of any practical use. For example, such relations may produce only coarse indications of the form: "A network receives too much load originating from beyond its Internet-facing gateways". Notice that an attack that employs only Lemma 2 will still be R-specific via relation (9), potentially enabling TE-based defense measures despite the unspecificity of the attack sources. Similarly, an attack that only uses Lemma 3 for obfuscation will still be L-specific via relation (8), enabling direct mitigation, e.g., via entity filtering or blacklisting.

However, obfuscating an attack via Lemmas 2 and 3 also comes with a price, regardless of their usage combination. An attacker must have an increased number of entities at his disposal, in order to efficiently use Lemma 2. Likewise, attacking extraneous targets (see Lemma 3) requires the use of additional entities as well. Given that the bandwidth (and the number of entities) required to flood a link is closely related to the TE choices of the victim, we proceed to study the effects of classic TE objectives to the specificity of relations (10).

C. Effects of Classic TE Objectives on the Detection Process

A usual TE objective is to react to link congestion/flooding events, redistributing the network load as equally as possible among the available routes [4]. This goal is most commonly expressed as the minimization of the maximum link utilization throughout the network, given the present traffic flow between each node pair [5]. The process is split into two stages:

1) Optimal calculation of path load. This stage receives as inputs the current traffic matrix and a set of possible paths connecting each pair of networked entities. It then produces the optimal aggregate load that each route should carry. This problem can be formulated as a Linear Program (LP) as follows. Let $M_{e,e'}$ be the current traffic matrix containing the data rates between any two entities e and e'. Let the triplet $\langle e, e', k\rangle,\ k = 1 \dots K$ index each of the K available paths that can connect the entities $e,\ e'$ (K can be a function of $e,\ e'$). Furthermore, let $f_{\langle e,e',k\rangle}$ represent the fraction of $M_{e,e'}$ over path $\langle e, e', k\rangle$.

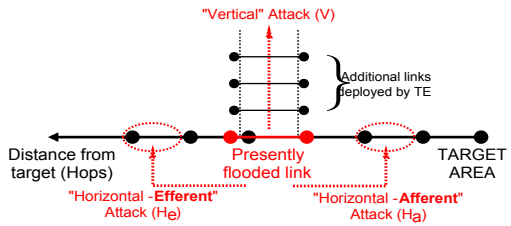


Figure 2: Types of attack responses to a link load-minimizing TE scheme. An attacker may move: i) "vertically", flooding the additional links deployed via TE, or ii) "horizontally", either nearing or distancing from the target area.

The objective of stage 1 is then achieved as follows [6]:

$$\begin{array}{ll} \text{minimize:} & \mathcal{U} \\ \text{subject to:} & \\ \forall_{e,e'}: & \sum_{\forall k} f_{\langle e,e',k\rangle} = 1 \\ \forall \langle e,e',k\rangle, \forall l \in \langle e,e',k\rangle: & \sum_{\forall e,e'} M_{e,e'} f_{\langle e,e',k\rangle} \leq \mathcal{U} \cdot C_l \end{array} \quad (11)$$

where C_l is the nominal capacity of link l and $\mathcal{U} \in [0,1]$ a helper variable. The first condition expresses the conservation of the traffic load, while the second one ensures that the load of each link is within its capacity constraint.

2) Mapping of entity pairs to paths. Given the current traffic matrix $M_{e,f}$, this stage maps pairs of entities $\langle e, e'\rangle$ to paths $\langle e, e', k\rangle$ in order to match the optimal $f_{\langle e,e',k\rangle}$ values produced in stage 1, after running the LP of formulation (11).

It is worth noting that there exist single-stage TE approaches based on metaheuristics (e.g., Genetic Algorithms), which attempt the same optimization [10], [11]. However, their performance is generally inferior to the LP-based approach.

The following remarks can be made on the relation between the TE module and the attack detection process. i) Since the TE is triggered on link congestion events, it will also be the first to respond to link flooding attacks as well. Therefore, the TE is a promising point for introducing detection-oriented mechanisms. ii) It is mandatory that the TE relieves the congested links, regardless of any actions pertaining to the attack detection process. iii) The second stage of TE, i.e., the entity pair to path mapping, relates entities to nodes and can potentially be tuned to aid detection, without changing the traffic distribution derived in stage 1. In light of these remarks, we proceed to study the distinct effects of minimizing the maximum link load (TE stage 1) on the detection process, regardless of any stage 2 TE approach. The tuning of TE stage 2 to detection purposes follows in Section III-D, based on relation extraction.

Assume that an attack floods a link l with capacity C_l affecting a target node, as shown in Fig. 2. At this point, let the aggregate attack traffic be $M \leq C_l$. A link load-minimizing TE scheme will reply by distributing the load of l over additional paths "parallel" to the congested. By definition of the min-max link utilization objective, no new path has 100% load after the TE step (assuming a non-saturated network). Therefore, flooding at least one of the parallel paths (including l) will require the increase of the aggregate attack traffic to $M' \geq M$.

In response to the TE run, the attacker may choose to attack "vertically", indeed flooding some or all of the deployed parallel links as required. However, the bandwidth emanating from each of the attacker' entities is upper bounded to avoid raising direct suspicions of flooding attempts [2]. Therefore, it is valid to assume that the number of entities participating to the attack, $n_{\mathcal{E}}$, increases with their total coerced load, i.e.,:

$$M' \geq M \Rightarrow n'_{\mathcal{E}} \geq n_{\mathcal{E}} \quad (12)$$

Assuming that the total number of entities available to the attacker is upper bounded, increasing $n_{\mathcal{E}}$ means that each entity must participate to attacks more frequently. Therefore, in accordance with Lemma 2, we conclude:

Proposition 4. *Vertical attacks favor the L-specificity of detected associative relations (L^+).*

The effect of vertical attacks to R-specificity may not be deduced independently of the 2^{nd} TE stage. The newly deployed parallel paths may simply implicate more nodes than those present at the right side of an existing $\mathcal{E} \rightarrow \mathcal{N}$ relation, decreasing R-specificity. However, other mappings may, e.g., divide $\mathcal{N}$ into node subsets that are routed via link-disjoint paths, increasing R-specificity in the face of a vertical attack.

Alternatively, an attacker may choose to reply "horizontally", towards the "efferent" direction (H_e), in an effort to deliberately decrease the R-specificity of $\mathcal{E} \rightarrow \mathcal{N}$ relations. Links further from the target area are likely to affect the routing of more nodes, increasing $\|\mathcal{N}\|$. However, such links will have greater nominal capacity for the same reason. This hypothesis is further reinforced by the over-provisioned nature of backbone links, capacity-wise. It is not uncommon for links to have a nominal capacity several times greater than the traffic demands of the served nodes [12]. Thus, attacking links towards H_e is expected to require increased entity participation in the general case. Therefore, similarly to Proposition 4 we conclude the following:

Proposition 5. *Horizontal-efferent attacks increase the L-specificity and decrease the R-specificity of detected associative relations (L^+, R^-).*

Working similarly, we deduce the duality of the "Horizontal-afferent" attacks, i.e., attacks to links closer to the target area:

Proposition 6. *Horizontal-afferent attacks decrease the L-specificity and increase the R-specificity of detected associative relations (L^-, R^+).*

Remark 7. We note that an attack must be horizontally bounded, i.e., not moving indefinitely towards H_a or H_e. Flooding links too near to the target area may give away the attack by minimizing the R-specificity of the detected relations. On the other hand, attacks too far from the target may actually miss their objective, affecting the connectivity of other nodes instead of the intended target, or demand too much bandwidth.

The effects of the attacker's responses are summarized in Table I. Notice that concurrent combinations of response types

Table I: Effects of attack responses on the L and R specificity of a detected entity→target relation.

Attacker's response types		
V	H_e	H_a
$L+$, $R\pm$	$L+$, $R-$	$L-$, $R+$

over a given path are also possible. In this case, however, due to Remark 1, the associative relations r observed after the attacker's response will overlap as $r_{H_e} \supseteq r_V \supseteq r_{H_a}$, yielding the specificity effects of the efferent-most attack. We observe that each of the attacker's responses yields a gain in either L or R-specificity, enforcing at least one of the relations (8), (9) or (10). Therefore:

Lemma 8. *The first stage of a min-max link utilization TE objective facilitates the detection of attacks at a given time step by increasing the L or R-specificity of the observed entity-to-target relations.*

Notice that Lemma 8 does not at first preclude attack *strategies*, i.e., series of attacker responses that may yield a solid loss in specificity over a time horizon. For instance, if H_e and H_a constantly yield $\langle L^{+1}, R^{-1} \rangle$ and $\langle L^{-10}, R^{+1} \rangle$ (using arbitrary units), then continuous alternations between the two responses would produce an unbounded loss in L-specificity with no effects on the R-specificity. However, the feasibility of such a strategy depends on the 2^{nd} TE stage, which quantifies the exact losses/gains in L/R-specificity. We therefore proceed next to directly optimize the 2^{nd} TE stage according to formulation (4), favoring the detection process.

D. Incorporation of Associative Relation Extraction to Classic TE Formulations

Let $r_t : e \xrightarrow{s} n$ be an observed relation with total support s at time step t. The goal of formulation (4) is to minimize s at time $t+1$, assuming that r_t represents a false positive.

Let $dest(e_f, t)$ return the destination of any *flow* e_f originating from entity e at time t, noticing that $dest(e_f, t) \neq n$ in general, by definition of the studied Crossfire attacks [2]. In addition, let H_t represent the history of the destinations of e_f:

$$H_t : \{dest(e_f, \tau),\, \tau \in [1, t]\} \tag{13}$$

Given H_t, a false-positive flow may retain its destination (affecting n or not) at $t+1$ with probability $P_{ret}(e_f) = P\left(dest\left(e_f, t+1\right) = dest\left(e_f, t\right) | \, H_t\right)$, or alter it to any other node $m \neq dest\left(e_f, t\right)$ (including null) with probability $P_m(e_f) = P\left(dest\left(e_f, t+1\right) = m | \, H_t\right)$. Let $P^a_{ret}(e_f)$ and $P^a_m(e_f)$ denote the probabilities of a flow e_f affecting n again at time $t+1$ due to this normal behavior justified by H_t. Minimizing the support s is then tantamount to minimizing the total probability of re-enforcing the relation $e \rightarrow n$ at $t+1$:

$$P^a_{t+1}(e) = \sum_{\forall e_f} \left(P_{ret}(e_f) \cdot P^a_{ret}(e_f) + \sum_{\forall m \neq dest(e,t)} P_m(e_f) \cdot P^a_m(e_f) \right) \tag{14}$$

Firstly, we proceed to minimize the term $P_{ret}(e_f) \cdot P^a_{ret}(e_f)$. Note that the term $P_{ret}(e_f)$ is invariant to any defender's actions, while $P_{ret}(e_f) \neq 0$ in the general case. We thus proceed to minimize the term $P^a_{ret}(e_f)$:

Remark 9. $P^a_{ret}(e_f) = 0$ when the pairs $\langle e_f, dest(e_f, t) \rangle$ and $\langle origin, n \rangle$ are mapped to link-disjoint paths in $\mathcal{T}(t+1)$.

The identity of the $origin$ (Fig. 1) may not be known, especially during the first attack rounds. Therefore, Remark 9 is practically implemented by mapping $\langle e_f, dest(e_f, t) \rangle$ to a path that is link-disjoint from $\langle s\left(l\right), n \rangle \; \forall flooded(l)$ at time t, $s\left(l\right)$ being the source node of l.

Secondly, we attempt to minimize the term $\sum_{\forall m} P_m(e_f) \cdot P^a_m(e_f)$ of equation (14). Let $\mathcal{M}(t+1)$ be the set of possible destinations m of e_f, $m \neq dest(e_f, t)$, affecting n in $\mathcal{T}(t+1)$ via one or more links (i.e., potential attack points). It holds that:

$$P_m(e_f) \cdot P^a_m(e_f) = \sum_{\forall m \in \mathcal{M}(t+1)} P\left(dest\left(e_f, t+1\right) = m | \, H_t\right) \tag{15}$$

In order to minimize equation (15), we employ virtual links [13] and proceed as follows:

Lemma 10. The configuration of the routing tables $\mathcal{T}\left(t+1\right)$ minimizes $\sum_{\forall m} P_m(e_f) \cdot P^a_m(e_f)$ when the following conditions hold: i) It connects $origin$ to n via a virtual link, ii) The virtual link comprises the smallest number of physical links (i.e., shortest path), iii) The nodes along the virtual link are the most improbable destinations of e_f at time $t+1$.

Proof: Condition (i) ensures that the set $\mathcal{M}\left(t+1\right)$ of equation (15) contains only the intermediate nodes encountered on the single, physical path represented by the virtual link. Without virtual linking, any physical link on the path from $origin$ to n may serve any number of additional nodes, increasing $\|\mathcal{M}\left(t+1\right)\|$. Condition (ii) ensures that $\|\mathcal{M}\left(t+1\right)\|$ has the minimal value supported by the network. Condition (iii) then minimizes $\sum_{\forall m \in \mathcal{M}(t+1)} P\left(dest\left(e_f, t+1\right) = m | \, H_t\right)$ by selecting the $\|\mathcal{M}\left(t+1\right)\|$ nodes with the lowest $P\left(dest\left(e_f, t+1\right) = m | \, H_t\right)$ values throughout the network. In order to circumvent the probably unknown identity of the $origin$, we work as in Remark 9, replacing $origin$ with $s(l)$ and repeating Lemma 10 for every $flooded(l)$ at time t. ■

Finally, the combination of Remark 9 and Lemma 10 leads to the following Theorem.

Theorem 11. *Let $r_t : e \xrightarrow{s} n$ be an associative relation detected at time t. Let e_f be a flow originating from e at time t. If the detection is falsely positive, the routing $\mathcal{T}(t+1)$ that minimizes the support s at time $t+1$:*

i) Routes $\langle e_f, dest(e_f, t) \rangle$ and $\langle origin, n \rangle$ via link-disjoint paths, $\forall e_f$. ii) Routes $\langle origin, n \rangle$ via a virtual link comprising the most improbable future destinations of e, $\forall e_f$. If all future destinations are equi-probable, $\langle origin, n \rangle$ is routed via the hop-wise shortest path between origin and n.

Qualitatively, Theorem 11 facilitates detection by forcing attacking entities to: i) constantly open new connections, and ii) use improbable decoys in the process. Notice that it also covers the total support of a relation up to $t+1$, which is expressed as the probability $\mathbb{P}^a_{t+1}(e) = \prod_{\tau=1}^{t+1} P^a_\tau(e)$. The theorem greedily reduces $\mathbb{P}^a_{t+1}(e)$ by minimizing each of the $P^a_\tau(e)$ terms.

Regarding its incorporation to a min-max link utilization TE scheme, the theorem can be implemented at the *entity pair to*

path mapping phase (TE stage 2). This phase maps each of the entity pairs to one of the physical paths connecting them, keeping the total load of each path near its optimal value. For each e_f, we first map $\langle e_f, dest(e_f, t)\rangle$ to an arbitrary path. Then, we proceed to list all available link-disjoint paths for $\langle origin, n\rangle$, following Theorem 11. If a pair of link-disjoint paths is not supported by the topology, we select paths with joint links as close as possible to $origin$ or n, employing Remark 7. Finally, we map $\langle origin, n\rangle$ to the returned path with the lowest $\sum_{\forall m \in path} P\left(dest\left(e_f, t+1\right) = m \middle| H_t\right)$ value. Notice that flows with common source and destination nodes are treated as an aggregated flow, via the same routing rules. Finally, all non-suspicious entity pairs are then mapped to links arbitrarily. The process has an average complexity of $O\left(\|\mathcal{R}_t\| \cdot K \cdot S \cdot F\right)$, where $\mathcal{R}_t$ is the set of observed relations r_t, K is the average number of physical paths per node pair, S is the average number of links comprising a path, and F is the average number of flows originating from any entity.

Theorem 11 also provides an insight on the topological attributes that affect the vulnerability of a network to Crossfire-like link-flooding attacks. Consider a full mesh topology, where all nodes are equi-probable flow destinations. According to part (ii) of the Theorem, the pair $\langle origin, n\rangle$ will always be routed via the shortest paths. However, in a mesh topology these paths always have a length of 1 hop and, therefore, contain no intermediate decoys. Thus, a Crossfire attack is not possible and the network is invulnerable. In essence, the longer the paths offered by a topology, the more the possible decoy nodes and the better the probability of launching Crossfire successfully.

E. On Reducing Complexity by Dissolving Infrequent Relations

While Theorem 11 can be implemented with linear complexity w.r.t. the number of observed relations, certain special cases may require additional attention. Firstly, an attacker may employ a very large number of entities for an attack. Secondly, sizable networks may yield too many probable attack targets. In addition, a single entity may be used for attacking multiple targets at once, increasing the number of observed relations further. Therefore, a defender may need to reduce $\|\mathcal{R}_t\|$ by dissolving some relations deterministically. The process is context-specific, depending on the capabilities and requirements of the defender. For instance, a top-x approach can be employed, keeping only the x relations with the greatest support. Another approach is to dissolve relations that have not been observed within a given timeout period. Another similar approach, advocated by well-known metaheuristics (e.g., ant colony optimization [14]) is to introduce a notion of "strength". The strength of a relation (which is not related to its support) is a number that increases every time the relation is observed, but otherwise decreases as time elapses. If the strength reaches a zero value, the relation is dissolved.

Regardless of the employed approach, the dissolution method must be carefully selected, since it may work in favor of the attacker. For instance, assume that an entity participates in each step of an attack with probability p. Let $g(t)$ be the strength gathered by the associated relations, which can be increased by $a > 0$ and decreased by $b > 0$ at each step. Consider that g is nullified for the first time at $t + \tau$. Then, it must hold that:

$$g(t)+a\cdot p\cdot\tau-b\cdot(1-p)\cdot\tau = 0 \overset{g(t)\geq 0}{\Rightarrow} [a\cdot p - b\cdot(1-p)]\cdot\tau < 0 \tag{16}$$

Therefore, since $\tau > 0$, we deduce that: $p < {}^{b}\!/_{(a+b)}$.

In other words, if an attacker uses a p value of less than ${}^{b}\!/_{(a+b)}$, then his bot relations and thus his attack will be stealthy.

Different dissolution approaches yield different behaviors. For instance, if the defender follows a $\tau-$timeout approach, a relation will be dissolved if not observed for τ time steps, i.e., with probability $P = (1-p)^{\tau}$. Therefore, an attacker may choose a p value which ensures that P is below a threshold. However, the attacker should have the necessary number of entities $\|\mathcal{E}\|$ at his disposal to take advantage of dissolution in any case, taking into account that his attacks should be successful with just $\|\mathcal{E}\| \cdot p$ entities. On the other hand, the defender should be aware of the trade-off resulting from his relation dissolution approach, and carefully balance computational complexity and detection potential.

Input: The flooded links $\mathcal{L}_{\mathcal{A}}$ at time t; The entities $\mathcal{E}_l$ with new connections present in each $l \in \mathcal{L}$; The nodes $\mathcal{N}_l$ affected by each $l \in \mathcal{L}$; The traffic matrix M.
Output: The relations (5), (6), (7) at time t; the new routing configuration at time $t+1$, i.e., $\mathcal{T}(t+1)$.

```
/* Initialization.                                                    */
Create persistent database tables R_EL, R_LN;
/* Remove shadowed links.                                             */
foreach l ∈ L_A do
    if ∃ l* ∈ L_A : E_l* → N_l* ⊇ E_l → N_l then
        L_A <-- L_A − l;
    end
end
/* Update Database.                                                   */
foreach l ∈ L_A do
    R_EL <-- R_EL ⋃ {⟨ϵ = e, λ = l, τ = t⟩ , ∀e ∈ E_l};
    R_LN <-- R_LN ⋃ {⟨λ = l, ν = n, τ = t⟩ , ∀n ∈ N_l};
end
/* Produce Relations.                                                 */
∀e ∈ R_EL : e -s-> ⋆, s = ‖σ_{ϵ=e} R_EL‖;
∀n ∈ R_LN : ⋆ -s-> n, s = ‖σ_{ν=n} R_LN‖;
∀⟨e, n⟩ ∈ (R : R_EL ⋈^λ R_LN) : e -s-> n, s = ‖σ_{⟨ϵ,ν⟩=⟨e,n⟩} R‖;
/* Dissolve Relations by TimeOut.                                     */
R_EL <-- R_EL − σ_{τ<t−TimeOut} R_EL;
R_LN <-- R_LN − σ_{τ<t−TimeOut} R_LN;
/* Perform TE.                                                        */
Calculate optimal load fractions f via formulation (11);
Produce T(t+1) via Theorem 11;
```

Algorithm 1: The proposed ARIEL scheme.

IV. Algorithmic formulation

The analytical findings of Section III lead to the formulation of Algorithm 1 (Associative RelatIon Extraction aLgorithm - ARIEL). The formulation allows for a straightforward implementation based on a relational database, exploiting the performance, stability and scalability benefits of this mature technology (e.g., SQL [7]). Therefore, we employ the additional

notation of $\sigma_c \mathbf{R}$ to express the selection of rows of a table $\mathbf{R}$ yielding true to the binary predicate condition c. In addition, $\mathbf{R}_1 \overset{col}{\bowtie} \mathbf{R}_2$ will denote the natural join of tables $\mathbf{R}_1$ and $\mathbf{R}_2$ on column col. ARIEL requires two persistent database tables, R_{EL} and R_{LN} with the columns detailed in lines $8-9$. The tables are created at the initialization phase. Subsequent runs of ARIEL utilize the same table instances as detailed next.

ARIEL is executed on a link-flooding event, after the execution of any extra heuristics for flow classification (e.g., [9]). The set containing the flooded links at the time of execution t is denoted as $\mathcal{L}_\mathcal{A}$. The entities present in each $l \in \mathcal{L}_\mathcal{A}$ are denoted as $\mathcal{E}_l$, while $\mathcal{N}_l$ contains the nodes whose traffic is routed via l. The defender may utilize common network logs to deduce which connections did not exist at $t-1$ [15]. ARIEL performs the task of attack detection at lines $2-15$ and min-max link utilization TE at lines $16-17$. Firstly, links that will lead to extraneous relations are filtered out of $\mathcal{L}_\mathcal{A}$, in accordance with Remark 1 (lines $2-6$). The process can be completed with an average of $O\left(\|\mathcal{L}_\mathcal{A}\| \cdot log\, \|\mathcal{L}_\mathcal{A}\| \cdot E_l\left[\|\mathcal{N}_l\| + \|\mathcal{E}_l\|\right]\right)$ calculations, given that the process can be treated as a partial sorting of the elements of $\mathcal{L}_\mathcal{A}$ by the $\supseteq$ operator. Each comparison operation then requires $O\left(E_l\left[\|\mathcal{N}_l\| + \|\mathcal{E}_l\|\right]\right)$ computations, $E_l\left[*\right]$ denoting the average of $*$ with regard to variable l. At steps $7-10$ ARIEL populates the tables R_{EL} and R_{LN} by processing $\mathcal{E}_l$, $\mathcal{N}_l$ as required by definition (3), while also adding timestamp information. The complexity is $O\left(\|\mathcal{L}_\mathcal{A}\| \cdot E_l\left[\|\mathcal{N}_l\| + \|\mathcal{E}_l\|\right]\right)$.

The detected associative relations are produced in lines $11-13$. As explained in Section III, an attacker may attempt to obfuscate the attacking entities or the attack target. At line 11, ARIEL attempts to detect all suspicious entities, regardless of target ($e \rightarrow \star$), in case the latter is obfuscated. For each distinct entity e, the support is proportional to the counting of all R_{EL} entries containing e. A similar approach is followed at line 12 to detect possible attack targets, regardless of the attacker's identity. Finally, the relations $e \rightarrow n$ are derived from the natural join of R_{EL} and R_{LN} over the links λ, at line 13. Notice that the support of the produced relations is *not normalized* in $[0, 1]$ as stated in equations (8-10). The normalization is skipped since it is trivial and reduces the clarity of the presentation of ARIEL. The complexity is $O\left(\|R_{EL}\| + \|R_{LN}\|\right)$, assuming that the database uses a hash approach to implement the join [7].

The algorithm then proceeds to dissolve "old" relations at lines $14-15$ by removing the corresponding entries from the tables R_{EL} and R_{LN}. The timeout approach is used as an example. Alternative dissolution approaches can be freely used at this point. The complexity is $O\left(\|R_{EL}\| + \|R_{LN}\|\right)$.

The TE task is accomplished at lines $16-17$. The complexity of the optimal link load calculation (line 16) depends on the employed LP solver [16]. Finally, the detection-oriented mapping of entity pairs to paths follows Theorem 11, whose process and complexity has been described in Section III-D.

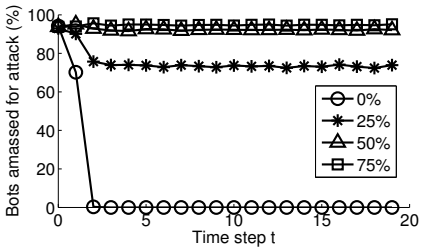
V. Simulations

We next perform simulations to study the effects of i) the TE phases (load spreading, entity mapping), and ii) the topological attributes on the specificity of relations extracted by ARIEL. The simulator, which we plan to release as a free open-source application, is implemented on the AnyLogic platform [17].

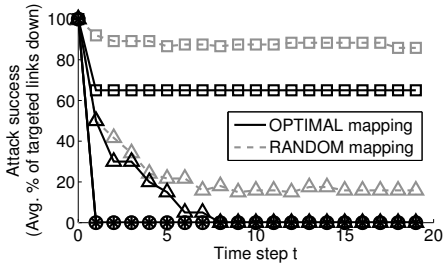
Setup. The simulations assume one synthetic and 50 real topologies (listed in the x-axis of Fig. 7, derived from the Internet Topology Zoo [18]). The synthetic topology comprises 25 nodes arranged in a 5×5 square grid. In every topology, all links are set to a capacity of $10GBps$. Furthermore, two alternative, link-disjoint paths are considered for each node-pair. In addition, each topology hosts a flat number of $10,000$ benign entities and $10,000$ bots, equally distributed to all nodes. This selection corresponds to a botnet of considerable size, i.e., the number of bots is equal to the total number of network users. Each entity (benign or bot) is allowed to have up to 5 connections opened at any given time. The origin and the attack target are selected as the most distant node pair (considering their hop-wise shortest path) in each topology. Time t is slotted, advancing at steps of 1 on link-flooding events (as in Section III), up to $t = 20$. The attacker operates as described in Section II. The bandwidth of each connection, $flow_{bw}$, is flat for bots or benign entities. Given a simulation configuration, $flow_{bw}$ needs to be calibrated to enable Crossfire. Too low $flow_{bw}$ is insufficient for link-flooding, while too high means that the whole network is flooded. We choose the lowest $flow_{bw}$ that enables the attack and provide this value at each Figure.

For the sake of experimentation, we define the input parameters $reuse\_ratio$ and $rehome\_ratio$, pertaining to bot and benign *connections*. A $reuse\_ratio = 10$ means that 10% of the active bot connections at time t, will re-adapt their destinations to participate in the attack at time $t+1$. The remaining 90% will remain unaltered. Similarly, the $rehome\_ratio$ defines how many benign connections will change their destination to a (uniformly) random node at time $t+1$. Finally, the metric Δs in introduced to quantify the specificity of detected relations. Δs is defined as the average (not normalized) support of relations involving bots/target nodes, minus the average support of relations involving benign entities/any other node. A higher Δs value means that bots/target nodes stand out more and, therefore, can be detected more efficiently. Finally, all runs are repeated for 95% confidence in the results.

Results. Figure 3 studies the general effects of TE on the efficiency of Crossfire attacks. Using the synthetic topology, we start with a bot connection $reuse\_ratio = 0\%$. In just two time steps, the attacker has run out of available bots (Fig. 3a) and is unable to flood the targeted links (Fig. 3b). A $reuse\_ratio$ of 25% increases the number of available bots to $\approx 75\%$, but their total connections are still insufficient for a successful attack. Thus, the attacker is forced to a $reuse\_ratio = 50\%$ which i) makes all bots visible (nearly 100% participation, Fig. 3a), while ii) achieving marginally successful attacks. It is at a $reuse\_ratio = 75\%$ when the attacks become consistently successful, should the defender use a random flow mapping at the 2^{nd} TE phase. The proposed optimal mapping makes even the $reuse\_ratio = 75\%$ insufficient, forcing the attacker to: i) use all his bots, and ii) use them almost exclusively for



(a) Effects of TE phase 1 (load balancing) on the bot availability.



(b) Effects of TE phase 2 (flow mapping to paths) on the attack success ratio.

Figure 3: The effects of two TE phases on the number of bots drawn to an attack and its success ratio. The $reuse\_ratio$ is varied in $0-75\%$. (Calibrated $flow_{bw} = 400KBps$, $rehome\_ratio = 10\%$).

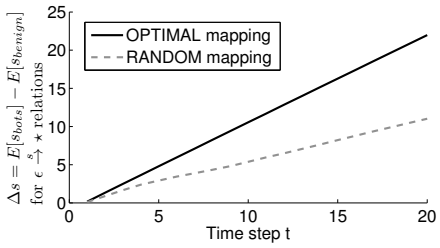


Figure 4: Effects of optimal/random mapping on the specificity of the $\epsilon \xrightarrow{s} \star$ relations. (Calibrated $flow_{bw} = 400KBps$, $rehome\_ratio = 10\%$).

attacks, accentuating their detection.

This becomes evident in Fig. 4, where Δs is doubled when the optimal mapping is used. We note that these findings are aligned to the theoretical hypotheses and conclusions of Sections III-C and III-D on the effects of TE on the detection process. Specifically, a load-balancing TE is shown to naturally force an attacker to use more bots to remain effective. In addition, the optimal mapping speeds up the detection. Notice that the random mapping yields a strictly increasing specificity as well, albeit at a slower rate.

We proceed to study specifically the effectiveness of an attacker's attempt to obfuscate his bots and his targets, corresponding to Lemmas 2 and 3. The $flow_{bw}$ is deliberately increased to yield a very congested network, in order to make attacks possible with less bots. Then, in Fig. 5, the $reuse\_ratio$ is varied from a high to a low value. As lower values are used, the bots participate to the attacks less frequently, which reduces the specificity of $\epsilon \rightarrow \star$ relations as expected by Lemma 2. At this point we also note that timing-out relations as described in Section III-E has the exact same effect.

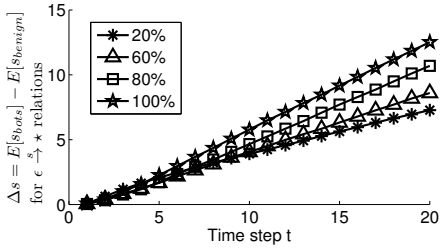


Figure 5: Effects of probabilistic bot participation to an attack on the specificity of the $\epsilon \xrightarrow{s} \star$ relations. The $reuse\_ratio$ ratio is varied between $20-100\%$. (Calibrated $flow_{bw} = 1400KBps$, $rehome\_ratio = 10\%$).

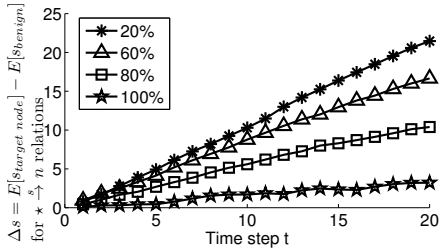


Figure 6: Effects of attacking random nodes (apart from the target) on the specificity of the $\star \xrightarrow{s} n$ relations. The $rehome\_ratio$ is varied between $20-100\%$. (Calibrated $flow_{bw} = 3500KBps$, $reuse\_ratio = 100\%$).

In Fig. 6 we set the $flow_{bw}$ to a value where the re-homing of benign connections causes link-flooding events by itself. The higher the $rehome\_ratio$, the more the naturally flooded links. In this manner, the attacker is expected to obfuscate his targeted nodes, as stated by Lemma 6. The results validate the theoretical claim, and the specificity of $\star \rightarrow n$ relations reduces when the $rehome\_ratio$ increases.

Finally, we proceed to test ARIEL in real topologies in Fig. 7. The topologies are random, selected alphabetically by name in the Topology Zoo database. Due to space restrictions, we illustrate the achieved specificity of the $\epsilon \rightarrow \star$ relations, given that the detection of the bots may be the first priority for the mitigation of the attack. ARIEL yields positive Δs in all cases, albeit with varying end-value at $t = 20$. To better understand the causes of this behavior, we tested the correlation between $\Delta s(t = 20)$ and several topology metrics (centrality, average number of neighbors, average shortest path length-AvgSPL, number of nodes, diameter, clustering coefficient, network density and network heterogeneity [19]). The AvgSPL metric exhibited the strongest covariance with Δs, as shown in Fig. 7. Specifically, the two plots yield a Pearson correlation coefficient of 0.7, with $P = 10^{-8}$ [20]. This outcome is aligned to Theorem 11, showing that topologies that offer longer paths between nodes are more vulnerable to attacks. However, the higher the vulnerability (higher AvgSPL), the better the detection result. We note though that this correlation is statistical and outliers exist (e.g., the "CrlNetServices" topology in Fig. 7). Nonetheless, the AvgSPL constitutes a good metric for an initial estimation of the vulnerability of the network, based on its topology.

Future work. We plan the following extensions. i) The formulation of the attacker's responses (Table I) paves the way

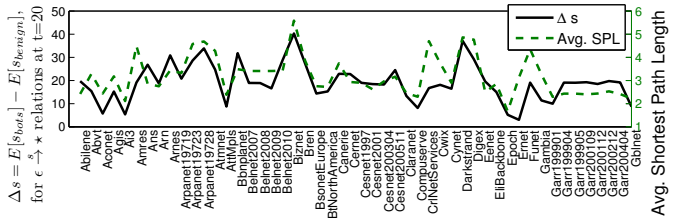


Figure 7: Effects of different topologies on the specificity of the $\epsilon \xrightarrow{s} \star$ relations. The effect is strongly correlated to the average-shortest-path-length (avg. SPL) topology metric.

for a game-theoretic approach, where the goal is to derive optimal attack and defense strategies. ii) Novel topological metrics should be defined to quantify the vulnerability of a network deterministically. iii) Noticing that the plots of Fig. 3a-6 are very well-formed (e.g., linear), we target the derivation of the exact formulas that govern the involved metrics.

VI. Related Work

Studer et al. introduce the Coremelt attack [3], where swarms of attack bots send traffic between each other in order to cause significant congestion within core network links, as collateral damage. The Crossfire attack [2]uses bots as sources and decoy servers as destinations, but falls within the same class of attacks. Research around such attacks has focused mostly on the system's side for detection. For example, Xue et al. propose the *LinkScope* system for detecting malicious link floods and for locating the target link or area whenever possible [8]. Their system uses end-to-end and hop-by-hop network measurement techniques to detect abrupt degradation of performance. The survey of Bhuyan et al. [21] gives an overview of the methods and tools used for detecting DDoS attacks; these range from statistical methods to machine-learning heuristic approaches. Zargar et al. [22] further present a comprehensive classification of various defense mechanisms against DDoS flooding attacks. In contrast to these studies, we propose a novel model and analysis of a joint detection and mitigation approach.

Several related DoS attack types have been studied in light of the SDN paradigm shift as well. Braga et al. capitalize on controller features for traffic analysis using Self Organizing Maps (SOMs) to classify flows and enable DDoS attack detection caused by *heavy hitters* [23]. Ashraf et al. provide a general survey of machine learning approaches for mitigating DDoS attacks in SDN environments [24]. Lim et al. propose a SDN-based scheme to block botnet-based DDoS attacks that do not exhibit detectable statistical anomalies [25]. The recent work of Lee et al. (CoDef) [26] is a first approach towards defeating new link-flooding attacks such as Coremelt and Crossfire. The authors propose a cooperative TE-based detection method for identifying low-rate attack traffic. The traffic sources and targets need to communicate directly via an extra protocol. Malicious traffic sources are identified by not complying with the instructed re-routing requests. Finally, Gkounis [27] studies the practical challenges of using SDN to implement a joint detection and mitigation scheme.

VII. Conclusion

This work introduced a novel framework for studying distributed link-flooding attacks. The goal of the framework is to facilitate the detection of susceptible bots and targeted network areas. This objective was formulated in terms of relational algebra and was seamlessly incorporated to standard TE modules. The analysis provided insights on optimizing the detection process, isolating the impact of the attacker and facilitating proper defender's reactions to the detection. Moreover, it shed light on the topological attributes that significantly influence the vulnerability of a network, attack-wise. The analytical insights were validated via extensive simulations on a variety of real and synthetic topologies.

References

[1] "The DDoS That Almost Broke The Internet," http://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet.

[2] M. S. Kang *et al.*, "The Crossfire Attack," in *IEEE SP 2013*.

[3] A. Studer *et al.*, "The Coremelt Attack," in *Proc. of ESORICS*, 2009.

[4] M. Pióro *et al.*, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Elsevier/Morgan Kaufmann, 2004.

[5] S. Balon *et al.*, "How Well Do Traffic Engineering Objective Functions Meet TE Requirements?" in *IFIP-TC6 2006*, pp. 75–86.

[6] A. R. Curtis *et al.*, "Mahout: Low-Overhead Datacenter Traffic Management Using End-host-based Elephant Detection," in *IEEE INFOCOM 2011*, pp. 1629–1637.

[7] D. Tow, *SQL Tuning*. O'Reilly Media, 2003.

[8] L. Xue *et al.*, "Towards Detecting Target Link Flooding Attack," in *ACM USENIX 2014*, ser. LISA'14, pp. 81–96.

[9] T. Thapngam *et al.*, "Discriminating DDoS Attack Traffic from Flash Crowd through Packet Arrival Patterns," in *Comp. Comm. Workshops - IEEE INFOCOM 2011*, pp. 952–957.

[10] M. Ericsson *et al.*, "A Genetic Algorithm for the Weight Setting Problem in OSPF Routing," *Jrn. of Comb. Optim.*, vol. 6(3), pp. 299–333, 2002.

[11] L. S. Buriol *et al.*, "A Hybrid Genetic Algorithm for the Weight Setting Problem in OSPF Routing," *Networks*, vol. 46, no. 1, pp. 36–56, 2005.

[12] R. Zhang-Shen *et al.*, "Designing a Predictable Internet Backbone Network." in ACM HotNets 2004.

[13] B. Koldehofe *et al.*, "The Power of Software-defined Networking: Line-rate Content-based Routing Using OpenFlow," in *ACM MW4NG 2012*.

[14] S. Luke, *Essentials of Metaheuristics*. Lulu Com, 2013.

[15] S. R. Chowdhury *et al.*, "PayLess: A Low Cost Netowrk Monitoring Framework for Software Defined Networks," in *IEEE/IFIP NOMS 2014*.

[16] N. Megiddo, *On the Complexity of Linear Programming*. IBM Thomas J. Watson Research Division, 1986.

[17] XJ-Technologies, "The AnyLogic Simulator," 2015. [Online]. Available: http://www.xjtek.com/anylogic/

[18] S. Knight *et al.*, "The Internet Topology Zoo," *IEEE Jrn. on Selected Areas in Comm.*, vol. 29, no. 9, pp. 1765–1775, 2011.

[19] S. Killcoyne *et al.*, "Cytoscape: A Community-based Framework for Network Modeling," in *Protein Networks and Pathway Analysis*. Springer, 2009, pp. 219–239.

[20] J. Benesty *et al.*, "Pearson Correlation Coefficient," in *Noise reduction in speech processing (chapter)*. Springer, 2009, pp. 37–40.

[21] M. H. Bhuyan *et al.*, "Detecting Distributed Denial of Service Attacks: Methods, Tools and Future Directions," *The Comp. Jrnl*, 2013.

[22] S. Zargar *et al.*, "A Survey of Defense Mechanisms against Distributed Denial of Service Flooding Attacks," *IEEE Comm. Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[23] R. Braga *et al.*, "Lightweight DDoS Flooding Attack Detection using NOX/OpenFlow," in *IEEE LCN 2010*, pp. 408–415.

[24] J. Ashraf *et al.*, "Handling Intrusion and DDoS Attacks in SDNs using Machine Learning," in *IEEE NSEC*, 2014, pp. 55–60.

[25] S. Lim *et al.*, "A SDN-oriented DDoS Blocking Scheme for Botnet-based Attacks," in *Proc. of IEEE ICUFN*, 2014, pp. 63–68.

[26] S. B. Lee *et al.*, "CoDef: Collaborative Defense Against Large-scale Link-flooding Attacks," in *ACM CoNEXT 2013*, pp. 417–428.

[27] D. Gkounis, "Cross-domain DoS Link-flooding Attack Detection and Mitigation Using SDN Principles," Master's thesis, ETH Zurich, 2014.