Botnet Identification in Multi-Clustered DDoS Attacks

Vincenzo Matta, Mario Di Mauro, Maurizio Longo
Department of Information & Electrical Engineering and Applied Mathematics
University of Salerno, Fisciano (SA), Italy
{vmatta, mdimauro, longo}@unisa.it

***Abstract*—In a randomized DDoS attack with increasing emulation dictionary, the bots try to hide their malicious activity by disguising their traffic patterns as "normal" traffic patterns. In this work, we extend the DDoS class introduced in [1], [2] to the case of a *multi-clustered* botnet, whose main feature is that the emulation dictionary is split over the botnet, giving rise to *multiple* botnet clusters. We propose two strategies to identify the botnet in such challenging scenario, one based on cluster expurgation, the other one on a union rule. Consistency of both algorithms under ideal conditions is ascertained, while their performance is examined over real network traces.**

***Index Terms*—Distributed Denial-of-Service, DDoS, Cyber-Security, Signal Processing for Network Security.**

I. Introduction and Motivation

More and more often, Distributed Denial-of-Service (DDoS) attacks hit the headlines for their dangerous impact on several real-world affairs. A DoS attack is realized through a bulky volume of requests sent to a target destination site, which is overwhelmed until its resources saturate, and the service to legitimate users is denied. The qualification of being "distributed" comes from the fact that such requests are sent by a net of dispersed machines (the *bots*), which can be malicious users acting consciously, or legitimate users that have been infected, e.g., by worms and/or Trojans. The bots can be coordinated by one or more *botmasters*, and the ensemble of bots is globally referred to as the *botnet*. The goal of the defender is identifying the members of the botnet, in order to ban the bots, without denying the service to normal users.

The simplest, centralized DoS attacks (e.g., TCP SYN flooding) exploited vulnerabilities in the protocol stack, relying essentially on repeated, high-rate transmissions of the same request from a single user. In such circumstances, the anomalous transmission rate was sufficient to identify the source of the attack. In contrast, in a *D*DoS attack the individual bot's rate is kept moderate, while the global attacking rate must be large. Nevertheless, without further sophistication, the compromised machines can be still identified at a single-user level. In fact, traffic patterns of normal users are usually characterized by a certain degree of innovation (for instance, as time elapses, distinct web-pages are likely to be visited), while the repetition scheme implicitly shows the anomalous bot character.

This work focuses on a more challenging variant of DDoS attack, namely, on the recent class of *application-layer* DDoS attacks. This peculiar form of attacks goes beyond the simplest repetition-based attacks, by exploiting the ample range of possibilities available at the application layer [3], [4]. In such novel attacks, the bots choose randomly their requests from a set of admissible messages (an *emulation dictionary*), trying so to disguise their traffic patterns as normal ones. The enhanced degree of variability in the message selection (e.g., the relatively large number of pages accessible in surfing through a website), makes the individual bot's patterns so reach to prevent from single-user inspection. As far as we know, the first formal characterization of the aforementioned class of randomized DDoS attacks has been provided in [1], [2], for the case where the botnet is composed by a single cluster using one and the same emulation dictionary.

In many practical situations, however, it is expected that the emulation dictionary is disseminated through the botnet, in such a way that distinct groups of bots have access to different portions of the overall emulation dictionary. This could happen for different reasons. One case is that, due to various constraints (e.g., bandwidth, energy), the botmaster sends to the bots only portions of the learned dictionary. Another case is a genuinely decentralized DDoS, where the botnet is clusterized in separate groups (perhaps coordinated by different botmasters, giving rise to a *hierarchical* DDoS) acting independently, and, in particular, performing the dictionary learning task separately.

A. Related Work

With no pretence of exhaustiveness, we now describe shortly some recent inferential strategies for DDoS identification. We redirect the Reader to [5] for a more comprehensive summary. Statistical methods for DDoS attack identification are proposed in [6], with focus on the detection of anomalies in the characteristics (e.g., entropy, relative frequencies) of selected packet attributes. In [7] a hierarchical method is presented, where shifts in spatial/temporal traffic patterns are exploited by the detection system to see where and when a flooding attack occurs. A method to detect low-rate DDoS attacks is proposed in [8], relying on the generalized entropy and on the information distance. In [9], shrew DDoS attacks (where flows are constrained to a small fraction of their rate) are examined, and the relationships between attack patterns and network environment are obtained by capturing the adjustment behaviors of the TCP congestion window at the target's side. Unfortunately, the aforementioned strategies are not suited to face the novel class of *randomized DDoS attacks*. To overcome this issue, in [1], [2] a botnet identification algorithm is proposed, which lies somehow between the two extremes of fully-parametric [10]–[12], and fully-data-driven [13] approaches. Following emerging trends in signal processing for network cyber-security [14]–[17], in [1], [2] consistent botnet identification is achieved through descriptive indicators (i.e., the message innovation rate and the emulation dictionary rate) that arise from a minimal set of physical assumptions.

However, the algorithms proposed in [1], [2] cannot deal with more general forms of attacks, such as the *multi-clustered* attack object of this work, whose goal is accordingly filling this gap. We propose two novel algorithms aimed at revealing the presence of multiple botnet clusters hidden in the network, and prove that such algorithms enable consistent botnet identification.

ISBN 978-0-9928626-7-1 © EURASIP 2017

II. The Multi-Clustered DDoS Attack

Let $N_{\mathcal{S}}(t)$ denote the overall number of transmissions occurred, up to time t, in a given subnet $\mathcal{S}$. The *transmission* activity of $\mathcal{S}$ is quantified in terms of the *empirical* transmission rate:

$$\boxed{\hat{\lambda}_{\mathcal{S}}(t) \triangleq \frac{N_{\mathcal{S}}(t)}{t}} \tag{1}$$

When a limiting (as $t \to \infty$) rate exists, it is denoted by $\lambda_{\mathcal{S}}$.

A second indicator relates instead to the message *content*. In order to characterize the variability in the activity of network users, we focus on the *new* messages that these latter produce as time elapses. Such variability can be quantified in terms of a Message Innovation Rate (MIR), which has been originally introduced in [1]. Let us collect into an *empirical dictionary*, $\mathscr{D}_{\mathcal{S}}(t)$, all the *distinct* messages sent, up to time t, by the users belonging to a given subnet $\mathcal{S}$. The *empirical* MIR can be accordingly defined as:

$$\boxed{\hat{\rho}_{\mathcal{S}}(t) \triangleq \frac{|\mathscr{D}_{\mathcal{S}}(t)|}{t}} \tag{2}$$

The limiting MIR, when it exists, is denoted by $\rho_{\mathcal{S}}$.

Our model for *multi-clustered* DDoS is inspired to recent kinds of application-layer DDoS [3], [4], and is a generalization of the DDoS class originally proposed in [1]. We assume that the botnet is made of C non-overlapping clusters, each of which has access to an emulation dictionary (at time t) denoted by $\mathscr{E}_c(t)$, for $c = 1, 2, \ldots, C$. A bot of the c-th cluster performs normal traffic emulation by picking admissible messages from $\mathscr{E}_c(t)$. In order to guarantee a non-suspicious innovation rate, the dictionary is learned in a *continuous* fashion, namely, its cardinality increases with t. To quantify richness of the emulation dictionary, we introduce the Emulation Dictionary Rate (EDR) if the c-th cluster:

$$\boxed{\alpha_c \triangleq \lim_{t\to\infty} \frac{|\mathscr{E}_c(t)|}{t}} \tag{3}$$

When a bot of the c-th cluster transmits, it picks (uniformly at random) a message from the available emulation dictionary $\mathscr{E}_c(t)$. As a result of the transmission activity, to any subnet $\mathcal{B}$ of the botnet we can associate a certain *empirical* dictionary, $\mathscr{D}_{\mathcal{B}}(t)$. At time $t + s$, such an empirical dictionary is possibly increased by embodying the *distinct* messages (which were not initially contained in $\mathscr{D}_{\mathcal{B}}(t)$) picked during interval s by the bots in $\mathcal{B}$.

A. Botnet MIR

The implications regarding the aforementioned network indicators have been examined in detail in [1], [2]. For the sake of completeness, the pertinent results are collected in the forthcoming theorem, which basically rephrases Theorem 1 in [1], [2] to handle the multi-clustered setting.

Theorem 1 (MIR of a multi-clustered botnet). *Let $\mathcal{B}_{\text{tot}}$ be a multi-clustered botnet, and let the transmission policies be either synchronous with constant transmission rate, or independent Poisson processes, with rates λ_u, for $u \in \mathcal{B}_{\text{tot}}$. Let $\mathcal{B} = \bigcup_{c=1}^{C} \mathcal{B}_c$, where $\mathcal{B}_c$ is a subnet of the c-th botnet cluster, and let α_c the EDR of the c-th cluster. If $\mathcal{B}_c \neq \emptyset$, the (limiting) MIR of $\mathcal{B}_c$ is:*

$$\boxed{\rho_{\mathcal{B}_c} = \frac{\alpha_c \lambda_{\mathcal{B}_c}}{\alpha_c + \lambda_{\mathcal{B}_c}}} \tag{4}$$

where $\lambda_{\mathcal{B}_c} = \sum_{u\in\mathcal{B}_c} \lambda_u$ is the aggregate transmission rate of $\mathcal{B}_c$. Moreover, the overall MIR of $\mathcal{B}$ fulfills the inequality:

$$\boxed{\rho_{\mathcal{B}} \leq \sum_{c:\mathcal{B}_c\neq\emptyset} \frac{\alpha_c \lambda_{\mathcal{B}_c}}{\alpha_c + \lambda_{\mathcal{B}_c}}} \tag{5}$$

which is satisfied with equality when the emulation dictionaries of the different clusters are mutually disjoint. ■

As regards the individual-cluster MIR in (4), the result comes directly from Theorem 1 in [1], [2]. As regards the overall MIR in (5), the result comes from the fact that the MIR is sub-additive, while the equality follows because disjointeness of the emulation dictionaries implies disjointness of the corresponding empirical dictionaries and, hence, additivity of the corresponding MIRs.

III. Botnet Identification Algorithms

The possibility of a successful botnet identification relies on the fact that bots and normal users are expected to behave quite differently as regards their degree of innovation. In fact, the members of a botnet cluster produce their transmission activity by picking messages from one and the same emulation dictionary. The implied commonalities between two members of the same botnet cluster are expected to emerge in terms of a MIR that is lower than the MIR that would be obtained, e.g., if the two users were normal. This is because the mutual independence of the activities of two normal users, or of a normal user and a bot, implies typically a low degree of correlation (some partial overlap could arise due to, e.g., common interests, popular web-pages, peculiar website structure), which is reflected in a small intersection between the corresponding (individual) empirical dictionaries. Such heuristic argument has been made precise in [1], [2]. Specifically, given two disjoint subnets, $\mathcal{S}_1$ and $\mathcal{S}_2$, two MIRs are introduced, namely, the sum of MIRs: $\hat{\rho}_{\text{sum}}(\mathcal{S}_1, \mathcal{S}_2) \triangleq \hat{\rho}_{\mathcal{S}_1} + \hat{\rho}_{\mathcal{S}_2}$, and the MIR of a reference botnet: $\hat{\rho}_{\text{bot}}(\mathcal{S}_1, \mathcal{S}_2) \triangleq \frac{\hat{\alpha}'(\mathcal{S}_1,\mathcal{S}_2)(\hat{\lambda}_{\mathcal{S}_1}+\hat{\lambda}_{\mathcal{S}_2})}{\hat{\alpha}'(\mathcal{S}_1,\mathcal{S}_2)+\hat{\lambda}_{\mathcal{S}_1}+\hat{\lambda}_{\mathcal{S}_2}}$, with explicit dependence upon t being suppressed for ease of notation. The value $\hat{\alpha}'(\mathcal{S}_1, \mathcal{S}_2)$ in the latter formula is a reference EDR estimated from the data. The detailed procedure to compute it is available in [2], and is not reported here for space constraints. Then, for $\epsilon \in (0, 1)$, an intermediate threshold is defined as: $\gamma(\mathcal{S}_1, \mathcal{S}_2) = \hat{\rho}_{\text{bot}}(\mathcal{S}_1, \mathcal{S}_2) + \epsilon\,[\hat{\rho}_{\text{sum}}(\mathcal{S}_1, \mathcal{S}_2) - \hat{\rho}_{\text{bot}}(\mathcal{S}_1, \mathcal{S}_2)]$. The heuristic reasoning about identifiability translates into the following conditions. When the two subnets belong to the *same* botnet cluster (below referred to as "joint case"), the empirical MIR, $\hat{\rho}_{\mathcal{S}_1\cup\mathcal{S}_2}$, converges toward $\hat{\rho}_{\text{bot}}$ as time elapses, as predicted by Theorem 1. Next, consider the case that one subnet contains normal users and/or bots belonging to clusters not contained in the other subnet. In such case (below referred to as "nearly-disjoint case") it is realistic to assume that the degree of dependence between the two subnets is lower than the degree of dependence observed when both subnets belong to the *same* botnet cluster. The above arguments lead to:

$$\text{Joint case} \quad \Rightarrow \quad \hat{\rho}_{\mathcal{S}_1\cup\mathcal{S}_2} < \gamma(\mathcal{S}_1, \mathcal{S}_1), \tag{6}$$

$$\text{Nearly-Disjoint case} \quad \Rightarrow \quad \hat{\rho}_{\mathcal{S}_1\cup\mathcal{S}_2} \geq \gamma(\mathcal{S}_1, \mathcal{S}_1). \tag{7}$$

Actually, when (7) is exactly verified (the verification of (6) being guaranteed, for t large enough, by Theorem 1), we shall say that the Botnet Identification Condition (BIC) is fulfilled.

Building upon the recipe summarized by (6) and (7), in [1], [2] an algorithm is proposed (named BotBuster), which exhibits

ISBN 978-0-9928626-7-1 © EURASIP 2017

Algorithm: $\hat{\mathcal{B}}$=BotClusterBuster(traffic patterns, ϵ, κ, ξ)

$\mathcal{N} = \{1, 2, \ldots, N\}$; $\hat{\mathcal{B}} = \emptyset$
for $i \in \mathcal{N}$ **do**
  $\hat{\mathcal{B}}_i = \{i\}$
  for $j \in \mathcal{N} \setminus \{i\}$ **do**
    if $\hat{\rho}(\hat{\mathcal{B}}_i \cup \{j\}) < \gamma(\hat{\mathcal{B}}_i, \{j\})$ **then** $\hat{\mathcal{B}}_i = \hat{\mathcal{B}}_i \bigcup \{j\}$
  end
  if $|\hat{\mathcal{B}}_i| = 1$ **then** $\hat{\mathcal{B}}_i = \emptyset$
  if $\hat{\lambda}_{\hat{\mathcal{B}}_i} \leq \frac{\kappa}{1+\kappa} \xi \, \hat{\lambda}_{\mathcal{N}}$ **then** $\hat{\mathcal{B}}_i = \emptyset$ (cluster expurgation)
end
$\hat{\mathcal{B}} = \bigcup_{i=1}^{N} \hat{\mathcal{B}}_i$

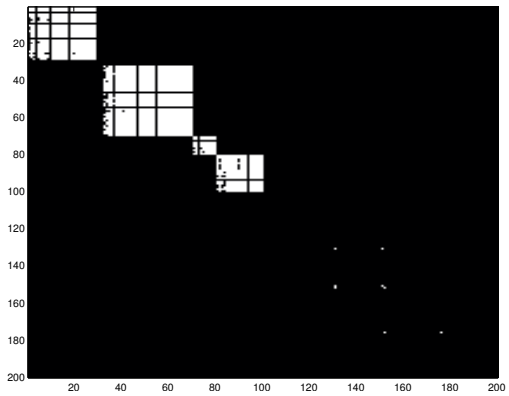

Fig. 1. Candidate botnet clusters: screenshot of the algorithm's output.

the following basic features *for the case that a single botnet (i.e., $C = 1$) is hidden in the network*: i) under the BIC, the true botnet is estimated consistently; ii) the algorithm has complexity $\mathcal{O}(N^2)$, and is further open to parallelization.

However, there is an issue that forbids successful applicability of the BotBuster algorithm to the multi-clustered case addressed in this work. Such issue relates to the fact that (as experimental verification reveals) the BIC is *not* always verified in practice. As a result, during its flow, the algorithm occasionally produces, along with the (nearly-)right botnet, spurious groups of users that are not the right botnet. In the single-cluster case, such pathology is remediated by choosing, at the end of the procedure that scans all the nodes as pivots, the estimated botnet with the highest cardinality [1], [2]. Such choice is based on the observation that the cardinality of groups erroneously marked as botnet is typically much smaller than the cardinality of a real botnet.

In the multi-clustered case, opting for the same maximum-cardinality rule is clearly detrimental, since it would select only the largest botnet cluster, which might be a largely insufficient measure of protection to face the DDoS attack. Therefore, different strategies are necessary. In the forthcoming sections we design two strategies suited to face a multi-clustered DDoS attack.

A. Main Routine for Multi-Clustered Botnet Identification

We start by examining the algorithm BotClusterBuster, whose pseudo-code is reported at the top of this page. We consider the operation of the algorithm at a given time epoch. For the sake of simplicity, dependence upon time is suppressed. Initially, the algorithm selects the first user as pivot (this operation will be repeated for all N nodes). User 1 is initially declared as a bot ($\hat{\mathcal{B}}_1 = \{1\}$). Then, by means of (6) and (7), it is declared whether users 1 and 2 form a botnet. If so, $\hat{\mathcal{B}}_1 = \{1, 2\}$, otherwise $\hat{\mathcal{B}}_1 = \{1\}$. Then, it is declared whether the currently estimated botnet $\hat{\mathcal{B}}_1$ forms a botnet with user 3, and so on. At the end of this loop, a candidate botnet cluster $\hat{\mathcal{B}}_1$ is obtained (if the candidate cluster has cardinality equal to one, it is automatically discarded). After iterating such inner loop over the entire set of pivots, the algorithm ends up with a sequence of candidate clusters, namely, $\hat{\mathcal{B}}_1, \hat{\mathcal{B}}_2, \ldots, \hat{\mathcal{B}}_N$. We remark that, differently from the BotBuster algorithm of [1], [2], *all* the candidate botnet clusters produced in the intermediate algorithm steps should be retained, in order to take into account the possible presence of *multiple* botnet clusters. The situation is pictorially illustrated in Fig. 1, where we display the candidate botnet clusters estimated by the algorithm at a certain time, with reference to a network composed by 100 normal users and 100 bots, with 4 true botnet clusters, with sizes $10, 20, 30, 40$. The i-th "row" of the image represents the output of the algorithm when user i is chosen as a pivot. A white pixel means "estimated bot presence", a black pixel means "estimated bot absence". Accordingly, if the (i, j)-th pixel is white, the algorithm is estimating that user j is a bot when user i is chosen as pivot. From Fig. 1, we can appreciate the emergence of 4 clusters, corresponding to the true botnet clusters (bots are ordered so as to appear well-clusterized in the image, a choice made only for clarity, since the algorithm is clearly invariant to permutations). On the other hand, we also see that a couple of small spurious clusters is wrongly identified by the algorithm.

Now, were the BIC verified, all checks performed by the algorithm would give the right answer (with probability tending to 1 as $t \to \infty$), and the sequence of candidate clusters would contain exclusively (repetitions of) the true botnet clusters. Therefore, a sufficient criterion to produce the global botnet estimates would consist of applying the union operator: $\hat{\mathcal{B}} = \bigcup_{i=1}^{N} \hat{\mathcal{B}}_i$. The corresponding algorithm will be referred to as UnionBotBuster. The pseudo-code for UnionBotBuster can be retrieved from the pseudo-code of BotClusterBuster, by simply skipping the instruction referred to cluster expurgation. However, since in practice the BIC is only approximately verified, the union rule would favor inclusion of spurious clusters. Therefore, some refined criterion to select the best clusters is desirable.

B. Cluster Expurgation

The DDoS power can be measured in terms of the global botnet transmission rate, $\lambda_{\mathcal{B}}$. For a DDoS attack to be effective, we expect that, for $\kappa \geq 1$: $\lambda_{\mathcal{B}} = \kappa \lambda_{\mathcal{N} \setminus \mathcal{B}}$, with $\lambda_{\mathcal{N} \setminus \mathcal{B}}$ being the global transmission rate of normal users. We stress that the lower bound $\kappa = 1$ corresponds to the optimistic (at the botnet's side) assumption that a low DDoS rate is sufficient to impair the target site. In terms of the overall network transmission rate, we get: $\lambda_{\mathcal{B}} = \frac{\kappa}{1+\kappa} \lambda_{\mathcal{N}}$. It is reasonable to assume that a true botnet cluster concurs to the attack with a significant fraction of the attacking rate, which leads to the following botnet membership condition: for $i = 1, 2, \ldots, N$, and for $\xi \in (0, 1)$, the candidate cluster i is retained if:

$$\hat{\lambda}_{\hat{\mathcal{B}}_i} > \frac{\kappa}{1+\kappa} \xi \, \hat{\lambda}_{\mathcal{N}} \triangleq \tau, \tag{8}$$

and is discarded otherwise. The final estimate is then produced by applying the union operator to the survived clusters, namely, $\hat{\mathcal{B}} = \bigcup_{i: \hat{\lambda}_{\hat{\mathcal{B}}_i} > \tau} \hat{\mathcal{B}}_i$. Let us now introduce two performance indices:

ISBN 978-0-9928626-7-1 © EURASIP 2017

for a network $\mathcal{N} = \{1, 2, \ldots, N\}$, containing a botnet $\mathcal{B}$ ($\hat{\mathcal{B}}(t)$ is the final botnet estimated at time t):

$$\eta_{\text{bot}}(t) = \frac{\mathbb{E}[|\hat{\mathcal{B}}(t) \cap \mathcal{B}|]}{|\mathcal{B}|}, \quad \eta_{\text{nor}}(t) = \frac{\mathbb{E}[|\hat{\mathcal{B}}(t) \cap (\mathcal{N} \setminus \mathcal{B})|]}{|\mathcal{N} \setminus \mathcal{B}|}, \quad (9)$$

which are the expected fraction of correctly identified bots and the expected fraction of normal users declared as bots, respectively. We shall say that an identification algorithm is consistent if:

$$\boxed{\lim_{t\to\infty} \eta_{\text{bot}}(t) = 1, \qquad \lim_{t\to\infty} \eta_{\text{nor}}(t) = 0} \quad (10)$$

THEOREM 2 (Consistency of the algorithms BotClusterBuster and UnionBotBuster). *Let $\mathcal{N} = \{1, 2, \ldots, N\}$ be a network containing a multi-clustered botnet $\mathcal{B}$, with asymptotically disjoint emulation dictionaries. The bots' transmission policies are either synchronous with constant transmission rate, or independent Poisson processes. The normal users' transmission policies are arbitrary. If the BIC in (7) holds, then the algorithm UnionBotBuster is consistent. Moreover, let $\lambda_{\min}$ be the smallest (limiting) transmission rate of a botnet cluster. If:*

$$\lambda_{\min} > \frac{\kappa}{1+\kappa}\,\xi\lambda_{\mathcal{N}}, \quad (11)$$

then the algorithm BotClusterBuster is consistent. ■

Consistency of UnionBotBuster holds because: i) by Theorem 1, as $t \to \infty$, a candidate cluster obtained starting with a *bot* pivot converges to the *true* botnet cluster containing the pivot; ii) since the BIC is assumed to be perfectly verified, a candidate cluster obtained starting with a normal-user pivot converges to the empty set. As regards BotClusterBuster, convergence of $\eta_{\text{nor}}(t)$ to zero is still implied by ii). On the other hand, successful inclusion of all candidate clusters coming from a bot pivot requires that the botnet-membership condition in (8) is verified for all clusters, at least for t large enough. This amounts to assume that, for the least favorable case (i.e., the cluster with the lowest transmission activity) the threshold is crossed. Since such condition is required asymptotically, the empirical values of the transmission rates in (8), are replaced, in (11), by their limiting counterparts. We remark that the theorem focuses on the case of (at least asymptotically) disjoint emulation dictionaries. Another relevant case could be that of partially overlapped emulation dictionaries, which is left for future work.

IV. EXPERIMENTAL RESULTS

An e-commerce portal has been selected as target destination of the attack. We employed a standard software for packet capturing, and performed a preliminary filtering: i) only the application-layer traffic directed to the target destination is retained; ii) the survived packets are divided on the basis of source IP address, before being fed to the identification algorithm. Following the above recipe, we collected more than 20 minutes of traffic, from 10 users (students/researchers in our laboratory). The obtained traffic streams have been then partitioned into 1-minute chunks. Then, each distinct chunk is treated as representative of a distinct user, for a total number of equivalent normal users ≈ 200. In order to check that the activity of the users is enough sustained, as well as compatible with typical traffic rates, we have performed a careful trace-by-trace inspection, and we have computed the average number of per-user TCP flows (2680), and the average packet size (776 bytes). As regards the multi-clustered DDoS attack, we need an emulation dictionary. To do so, we have first

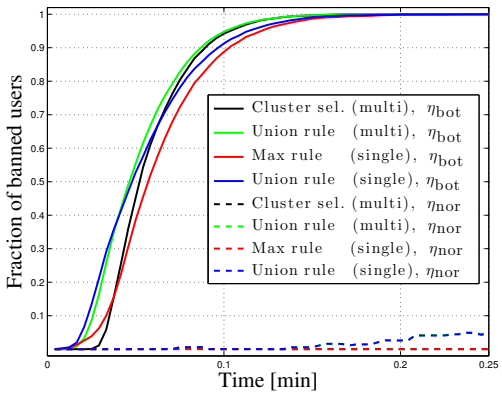

Fig. 2. Expected fraction of banned users (estimated over 100 Monte Carlo runs) as a function of time, for different types of attack and identification algorithm, as detailed in the legend. The network is made of 100 normal users and 100 bots.

taken all the *distinct* messages present in the *overall* dataset. Then, we have split such ensemble into C disjoint sets, with C being the number of clusters. Finally, at epoch t, the c-th emulation dictionary $\mathscr{E}_c(t)$ is constituted by the first $\lfloor e_0 + \alpha_c t \rfloor$ messages of such ensemble (e_0 is the initial dictionary size, set to 100 messages in the simulations).

In a first set of experiments, the individual bot transmission rate has been chosen as twice the average rate of normal users, and the EDR has been chosen as compatible with the innovation rates estimated over the normal users' traces. Such choices are made to let the bots well concealed in the midst of legitimate users. We considered an equal number (100) of normal users and bots, which is again a favorable choice for the attacker. With reference to such setting, we first launched a single-cluster DDoS. The botnet identification algorithms for this case are: the BotBuster algorithm introduced in [1], [2] (which selects the maximum-cardinality cluster only), and the UnionBotBuster algorithm. Then, we launched a *multi-clustered* DDoS, where the 100 bots have been split over $C = 4$ clusters, with sizes $10, 20, 30, 40$ (cluster ordering is immaterial, since, in our simulations, the bots are randomly spread over the network). The EDR of each cluster has been correspondingly reduced by a factor 4. For the multi-clustered scenario, the identification algorithms are BotClusterBuster and UnionBotBuster. As regards the threshold parameters of BotClusterBuster, we set: $\kappa = 1, \quad \xi = 1/10$. Both choices are not "too selective", in the sense that they tend to favor cluster inclusion, and, hence, they tend to increase η_{nor}. In fact, $\kappa = 1$ corresponds to a low reference attacking rate. Likewise, $\xi = 1/10$ corresponds to assume that the smallest cluster must contribute to the total attacking rate for at least one tenth of the total, which seems a rather conservative choice.

In Fig. 2, we display η_{bot} and η_{nor} as functions of time. Points of each curve correspond to the algorithms' output sampled each 0.25 seconds, with the average performance indices estimated over 100 Monte Carlo runs. As a general comment, we see that all algorithms do their job properly, since: i) the fraction of correctly banned users increases up to unity as time elapses, matching the theoretical results about algorithms' consistency; ii) the fraction of erroneously banned users is kept small. It is not zero, as a consequence of the forewarned imperfect verification of the BIC. Let us first compare the single-cluster case to the multi-clustered case. In particular, we focus on the comparison between the

ISBN 978-0-9928626-7-1 © EURASIP 2017

maximal-cluster rule (single) and BotClusterBuster (multi). We see that η_{nor} is in practice zero for both cases (dashed red curve almost perfectly superimposed to dashed black curve). Switching to η_{bot}, we see that the performance in the multi-clustered case outperforms the performance corresponding to the single-cluster case, but for uninteresting small values of η_{bot}. Such behavior can be explained as follows. The power of a botnet cluster is ruled by the EDR, which, in the multi-clustered case, is smaller than the EDR of the single-cluster case. Now, provided that the cluster selection operates properly, the main factor ruling the algorithm performance is the EDR. Accordingly, the botnet identification is quicker in the multi-clustered case (lower EDR $\Leftrightarrow$ lower attack strength) than in the single-cluster case (higher EDR $\Leftrightarrow$ higher attack strength). We now move on examining the connections between the cluster-expurgation rule and the union rule. The fraction of banned users for the latter is always higher than that corresponding to the former. This behavior is expected, since the union rule does *not* attempt to select the best cluster(s), but merges all candidate bots. Clearly, the increase in the percentage of correctly banned users is paid in the coin of an undesired increase of the wrongly banned ones.

In a second scenario (see Fig. 3), we consider two different implementations of the multi-clustered DDoS. In a first setting (case 1 in Fig. 3), we make the identification problem more challenging by i) reducing the individual bots transmission rate (by a factor 2); ii) reducing the size of the smallest botnet cluster, with the cluster sizes being $5, 20, 30, 45$. In the second setting (case 2 in Fig. 3), we still work with the reduced transmission rate, but the cluster sizes are restored to the values $10, 20, 30, 40$, while we reduce the value of the cluster-selection parameter ($\xi = 1/20$). With this choice, we are in a sense over-estimating the maximum number of clusters, and, hence, we are favoring the inclusion of *spurious* clusters in the final estimate. The results of the experiments are displayed in Fig. 3. We start by focusing on the BotClusterBuster algorithm. As a general trend (applying both to case 1 and case 2) we see that the new challenges (reduced transmission rate, reduced cluster size, reduced threshold parameter) correspond to an increase of η_{nor}. This is expected, since all the modifications w.r.t. the setting of Fig. 2 goes in the direction of favoring the inclusion of spurious clusters.

In contrast, as regards η_{bot}, the situation changes in the two cases. In case 1, the low value of the smallest cluster size (equal to 5), leads to a violation of hypothesis (11), which forces BotClusterBuster to exclude the smallest cluster, and in turn explains why η_{bot} saturates to 0.95 as time elapses. Instead, in case 2, the variations of the sensible parameters are not sufficient to impair consistency. As regards the UnionBotBuster algorithm, considerations similar to the cases in Fig. 2 can be drawn. However, since now the observation window is increased, we notice an increasing trend in η_{nor}, which is probably due to the fact that, even if the spurious clusters are typically few and small, the union rule tend to enhance their contribution.

As a result of the conducted analysis, some useful insights can be gained. While no superiority of one class of algorithms over the other class can be generally claimed, we feel that algorithms performing a cluster selection could be more appropriate in typical DDoS scenarios, for the following reason. In a DDoS attack, the main goals of the network defender are: i) avoiding that the destination site crashes; ii) guaranteeing proper service to the legitimate users. Thus, it seems preferable to ban few normal users (lower η_{nor}), at the price of losing some bots (lower η_{bot}). Indeed, banning, e.g., 95% of the bots should not impair the destination site, and keeping η_{nor} as low as possible means minimizing the number of normal users with denied service.

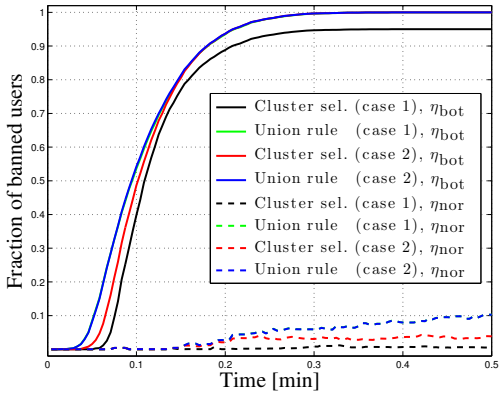

Fig. 3. Expected fraction of banned users (estimated over 100 Monte Carlo runs) as a function of time, for different parameters of the multi-clustered DDoS and of the algorithms. The network is made of 100 normal users and 100 bots.

REFERENCES

[1] V. Matta, M. Di Mauro, and M. Longo, "Botnet identification in randomized DDoS attacks," in *Proc. EUSIPCO*, Budapest, Hungary, Aug./Sep. 2016, pp. 2260–2264.

[2] V. Matta, M. Di Mauro, and M. Longo, "DDoS Attacks with Randomized Traffic Innovation: Botnet Identification Challenges and Strategies," *IEEE Trans. Inf. Forensics and Security*, vol. 12, no. 8, pp. 1844–1859, Aug. 2017.

[3] "Taxonomy of DDoS attacks." http://www.riorey.com/types-of-ddos-attacks/#attack-15.

[4] S. Ferretti and V. Ghini, "Mitigation of random query string DoS via gossip," *Communications in Computer and Information Science*, vol. 285 CCIS, pp. 124–134, 2012.

[5] N. Hoque, D. Bhattacharyya, and J. Kalita, "Botnet in DDoS attacks: trends and challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2242–2270, fourth quarter 2015.

[6] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," in *Proc. DISCEX*, Washington, DC, USA, Apr. 2003, pp. 303–314.

[7] J. Yuan and K. Mills, "Monitoring the macroscopic effect of DDoS flooding attacks," *IEEE Trans. Depend. Secure Comput.*, vol. 2, no. 4, pp. 324–335, Oct. 2005.

[8] Y. Xiang, K. Li, and W. Zhou, "Low-rate DDoS attacks detection and traceback by using new information metrics," *IEEE Trans. Inf. Forensics and Security*, vol. 6, no. 2, pp. 426–437, Jun. 2011.

[9] J. Luo, X. Yang, J. Wang, J. Xu, J. Sun, and K. Long, "On a mathematical model for low-rate shrew DDoS," *IEEE Trans. Inf. Forensics and Security*, vol. 9, no. 7, pp. 1069–1083, Jul. 2014.

[10] M. Barni and B. Tondi, "The source identification game: an information theoretic perspective," *IEEE Trans. Inf. Forensics and Security*, vol. 8, no. 3, pp. 450–463, Mar. 2013.

[11] B. Kailkhura, S. Brahma, B. Dulek, Y. S Han, and P. Varshney, "Distributed detection in tree networks: Byzantines and mitigation techniques," *IEEE Trans. Inf. Forensics and Security*, vol. 10, no. 7, pp. 1499–1512, Jul. 2015.

[12] S. Marano, V. Matta, and L. Tong, "Distributed detection in the presence of Byzantine attacks," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 16–29, Jan. 2009.

[13] L. Györfi, M. Kohler, A. Krzyżak, H. Walk, *A Distribution-Free Theory of Nonparametric Regression*, 2nd ed. New York: Springer-Verlag, 2002.

[14] M. Mardani, G. Mateos, and G. B. Giannakis, "Dynamic anomalography: tracking network anomalies via sparsity and low rank," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 1, pp. 50–66, Feb. 2013.

[15] P. Venkitasubramaniam, T. He, and L. Tong, "Anonymous networking amidst eavesdroppers," *IEEE Trans. Inform. Theory*, vol. 54, no. 6, Jun. 2008.

[16] J. Kim and L. Tong, "Unsupervised and nonparametric detection of information flows," *Signal Processing*, vol. 92, no. 11, pp. 2577–2593, Nov. 2012.

[17] S. Marano, V. Matta, T. He, and L. Tong, "The embedding capacity of information flows under renewal traffic," *IEEE Trans. Inform. Theory*, vol. 59, no. 3, pp. 1724–1739, Mar. 2013.

ISBN 978-0-9928626-7-1