A Collaborative Approach to Facilitate Intrusion Detection and Response against DDoS Attacks

Saman Taghavi Zargar
Networking and Telecommunications Department
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA 15260, USA
stzargar@sis.pitt.edu

James B.D. Joshi
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA 15260, USA
jjoshi@sis.pitt.edu

Abstract—**Intrusion detection and response systems (IPSs) for protecting against distributed denial-of-service (DDoS) attacks will benefit significantly if all the routers within each autonomous system (AS) are capable of detection and response in addition to sampling. However, DDoS detection and response will incur high storage and processing overhead if each router does redundant detection and response tasks. Many overlay communication protocols have been introduced in the literature to achieve coordination among the routers but they generally have high communication overheads. Furthermore, DDoS detection and response requires that all the flows intended to the same destination be analyzed together in order to efficiently capture the correlation between them. In order to do that, current approaches centrally collect all the sampled data and analyze them, which also increases the communication overhead. In this paper, we present a collaborative approach to distribute the sampling, detection, and response responsibilities among all the routers within the AS in such a way that each router can detect and respond to DDoS attacks. Our proposed approach achieves coordination among all the routers in the network to eliminate redundant sampling, detection, and response tasks without exploiting any specific communication protocol. We propose an optimal assignment of disjoint flows to each of the routers within the ASs in such a way that all the flows destined for the same host will be sampled, analyzed, and properly responded at the same router. Each router can thus capture the correlation between flows destined for a specific destination.**

Keywords-Network security; Intrusion detection systems; DDoS attacks;distributed IDS ; collaborative IDS

I. INTRODUCTION

DDoS attacks often originate from a group of organized and widely scattered zombies simultaneously and continuously sending a large amount of traffic to a host (Destination flooding) or a link (Link flooding). A DDoS attack aims to disturb the host's communication or congest a link in order to disrupt the legitimate communications through that link. Attackers usually forge sources to hide zombies' real locations. Most of the time, by the time DDoS attack is detected, there is nothing that can be done except to disconnect the victim from the network and manually fix the problem [1] [2]. DDoS attacks waste a lot of resources (e.g. processing time, space, etc.) on paths that lead to the targeted machine; hence, it is desirable to detect and stop such attacks as soon as possible. Intrusion detection and response systems or IPSs, aim to detect and respond to attacks as early as possible. There are different approaches to address DDoS attacks already proposed in the literature based on two intrusion prevention systems, mostly host-based IPS (HIPS) and network-based IPS (NIPS) [1] [2] [3]. HIPSs are deployed on end-hosts, either source (Source-side) or destination (Destination-side) of the attack. Source-side detection and response approaches such as ingress filtering [4], D-WARD [5], and MULTOPS [6] can take place at either edge routers of the local network or access routers of an AS that connect to the subscribers' edge routers [1]. Source-side HIPSs aim to detect and filter the attack traffic but they are not practical against DDoS attacks. There are two reasons which make the generation of filtering rules against DDoS not practical at source-side HIPSs. First, the sources of attacks can be distributed in different domains making it difficult for each of the sources to detect and respond accurately. Hence, collaborative attack detection and response approaches are required to capture all the traffic from all the distributed sources to the victim. Second, it is difficult to differentiate legitimate and DDoS attack traffic at the sources, since the volume of the traffic is not big enough and traffic only aggregates at the points close to destinations. Although, D-WARD can generate filtering rules at the source, it consumes more memory space and CPU cycles than some NIPSs [7]. Hence, source-side HIPSs are not effective against DDoS attacks.

In the destination-side HIPSs, detection will be done mostly at the destination and the response will be initiated and distributed to other nodes by the victim. There exist various destination-side approaches where detection and response are placed either at the edge routers or access routers of the destinations' AS. In the first kind of destination-side detection and response approaches, called trace-back [8] [9] [4], routers in the path to the victim, mark packets (i.e. add routers' identification to each packet) so that the victim can identify the path of attack traffic and distinguish it from legitimate traffic after the detection. However, storing the entire path in the IP identification field of each packet needs certain coding schemes and these schemes sometimes are not able to assign each mark to a unique path; hence, false positive rates of these approaches are still high. In other words, legitimate packets would be treated as attack packets. Another approach is to limit the rate of attack traffic which causes congestion to the destination link or host (e.g. Pushback [10], and ACC [10]) but it does not completely stop the attack traffic. Packet filtering at the edge routers (e.g. Pi [11], Preferential filtering [12], [10], [13], [14]) is the third kind of destination-side approaches. This approach lets the receivers install dynamic network filters to block the

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.46

undesirable traffic. Packet filtering approaches are completely dependent on attackers' power, and when it increases, filters become ineffective and they cannot properly be installed. The forth kind of destination-side approaches is capability approach [15] that lets the destination explicitly authorize the traffic it desires to receive (e.g. Portcullis [16], TVA [17], and SIFF [18]). Senders obtain the capabilities, which are short-term authorizations, from the receivers and put a stamp on their packets. However, granting capabilities is an important challenge which is addressed by the source authentication system on recent proposals (e.g. Passport [19] TVA+ [13]). Nevertheless, when attackers can get capabilities from colluders, the capability approaches are ineffective [20].

Most of the HIPSs are not capable of detecting and responding to the attack traffic properly. They cannot accurately detect and respond to the attack at the source or stop the attack before it reaches the victims. Therefore, NIPSs propose to address this problem and to help HIPSs to do their job accurately. NIPSs are deployed inside the networks, e.g., on the routers [7]. Detecting attack traffic and creating proper response to stop it at the routers is an ideal goal for network-based approaches. However, it incurs high storage and processing overhead at the routers if each router does redundant detection and response through the path to the destination, which can present a significant burden. Various researchers have proposed different approaches to reduce the amount of storage and consumption of CPU cycles for detection and response at the routers (e.g. Bloom filters [7] [21], Packet sampling [22], etc.) but these approaches are not sufficient when routers still do redundant jobs. Moreover, reducing the amount of redundant detection and response between the routers requires coordination among them. Various communication protocols [1] have been proposed to coordinate attack detection and response. However, NIPSs that have been proposed thus far are not very effective and efficient because they incur huge communications overhead. The lack of adequate bandwidth during DDoS attacks may limit the protocol for communications and cause NIPSs to fail. Furthermore, one of the most important ways to detect DDoS attack is to find the correlation between different flows intended to the same destination. Current NIPSs are capable of finding this correlation either at the destination or centrally at each of the ASs on the path. The former cannot detect and respond early enough to prevent resource consumption along the path, and the latter needs centralized collection of all the sampled/monitored data (mostly redundant unless coordinated between all the routers) from all the routers in each AS to be analyzed which increases the communication overhead.

In this paper, we propose a collaborative approach to distribute the sampling, detection, and response responsibilities among all the routers within the AS in such a way that, DDoS attack detection and response approaches will be facilitated with our approach.

Our contributions in this paper are as follows:

- We propose a coordination scheme to aid collaboration among all the nodes in the network without exploiting any specific communication protocol, which significantly reduces communication overhead.
- Our approach assigns disjoint flows to each of the routers within the AS in such a way that all the flows intended for the same destination host (Destination flooding) will be sampled and further analyzed at the same router. Therefore, the correlation between those flows destined to the same machine can be captured by that specific router.
- Consequently, each router within the AS can sample, analyze, and respond to DDoS attacks, as near as possible to the source and before attack flows get near the destination; hence, reducing the amount of ineffective resource consumption.

There are approaches in the literature in which victims can effectively stop DDoS attacks after perfect detection (e.g. AITF [14], Stop-it [13]). Our contribution will facilitate those approaches in such a way that, responsible routers on a path can sample, detect and stop the attacks in addition to the victims' system and more importantly earlier than that.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes our proposed collaborative intrusion detection and response design. Section 4 shows our simulation experiments and results. Section 5 concludes the paper and presents possible future directions.

II. Related Work

In this section we discuss work closely related to our work.

Currently, flow monitoring, which provides data for detection and response, is typically done completely independently in each router. However, it may lead to redundant flow monitoring and inefficient use of router resources. Recently, Sekar *et al.* [23] have proposed a centralized system that coordinates monitoring responsibilities among all the routers in each AS and they show that their approach significantly increases the flow monitoring capabilities of the network. The Coordinated sampling (*CSamp*) approach [23] that they propose aims to:

1) *Provide high flow coverage*
2) *Minimize redundant reports*
3) *Consider routers' resource constraints*
4) *Support all the flow monitoring applications*
5) *Satisfy flow monitoring objective (e.g. ensure fairness)*

CSamp uses a hash-based sampling that leads to coordination among all the routers within the AS, without any explicit communication. Disjoint hash ranges are assigned to each router so that any flow's hash can at most match one router's hash range. CSamp also considers router constraints in each sampling period to maximize flow coverage.

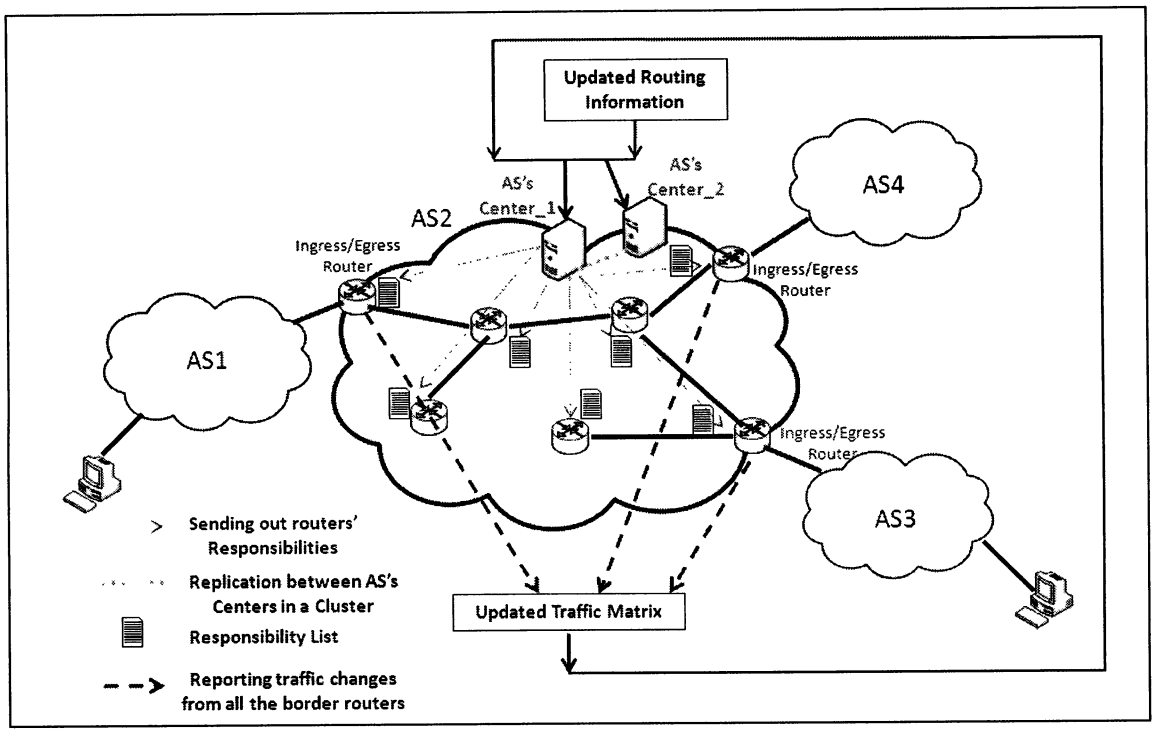

Fig. 1. Collaborative Intrusion Detection and Response Approach.

The optimization engine of *CSamp* uses the traffic matrix and the routing information as an input in each sampling period to compute the optimized distribution of sampling responsibilities among all the routers within the AS. The output of the optimization engine is then translated into sampling manifests or a list of hash ranges for each flow and then manifests are sent to all the routers within the AS.

The idea of reducing duplicate measurements in the network has been introduced earlier in [21]. CSamp adds two features to the idea of [21]. First, CSamp considers resource constraints on the routers. Second, CSamp uses hash-based sampling to obtain coordination among all the routers within the AS without any explicit communication.

CSamp has some limitations for an ideal NIPS:

1) CSamp assumes that *the detection and response will be done centrally after each measurement period.* Hence, in each AS, all the sampled data should be collected centrally and then analyzed to further detect and respond to possible attacks. This assumption *increases the communication overhead.*

2) The key effort in CSamp is to maximize the coverage of all the flows among all the routers but they *do not guarantee covering all the flows due to router constraints*; the CSamp approach is *not applicable for NIPSs and may lead to high false negative rates due to less number of flows that will be covered.*

Our proposed collaborative approach eliminates the communication overhead caused by central detection and response in *CSamp*. In order to do this, we introduce a centralized system that coordinates the responsibilities of detecting the attack flows, and responding to them in addition to sampling the traffic among all the routers in a path (within the AS). Similar to *CSamp*, we do not exploit any specific communication protocol to achieve coordination among all the routers toward eliminating redundant sampling, detection, and response tasks. Furthermore, to address the second limitation of *CSamp*, we cover all the flows within the AS in our approach by relaxing the router constraints. We believe that with the development of fast memory, better processing capacity, and with the current state of the art routers' features, relaxing these constraints of the routers is reasonable. However, we plan to extend our approach in future to consider and explore these constraints and their effects on our design in depth.

III. Collaborative Approach to Facilitate Intrusion Detection and Response

In this section, we present our proposed scheme; Figure 1. illustrates the overall idea of the scheme.

Each AS has a number of Ingress/Egress routers and some interior routers. Ingress/Egress routers are label switch routers that are Sources/Destinations for a given label switch path in a Multiprotocol Label Switching (MPLS) network. There are multiple Origin-Destination (OD) pairs or pairs of Ingress/Egress routers within each AS. OD pairs are characterized by their router-level paths. There is at least one center in each AS (or a cluster of centers to guarantee availability) for running the assignment/optimization problem we discuss later based on the traffic matrix and routing information they receive as inputs, and distributing routers' responsibility lists. *A responsibility list is a list of destination IP (DestIP) addresses that should be sampled, analyzed, and properly responded to in each router.* All the Ingress/Egress routers are responsible for reporting traffic dynamics to the

center/s so that it/they can update the responsibility lists promptly for future sampling periods.

Routing information will also be updated through flow-based monitors such as OSPF monitor [24]. For clarity of our design, we will first describe the assumptions we make, then we present our setup process to provide data needed to feed to our assignment/optimization problem we formulate below and finally the network-wide assignment/optimization formulation itself.

A. Assumptions

We make the following assumptions in our design:

- All the routers within each AS can detect the destination flooding attacks, depending on the number of correlated flows they can sample. There are several approaches proposed in the literature to detect DDoS attacks at the router level [7]. Although ASs closer to the victim will catch more attack flows to have more accurate detection, we believe it is possible and more efficient to detect some of those flows and stop them earlier at ASs closer to the source.
- The traffic matrix and the routing information of the network are available to the ASs.
- Flow sizes are the same for all the flows in our approach. As we mentioned earlier, we also relaxed the routers memory constraints.
- Our approach facilitates detection approaches to cover destination flooding attacks. We do not cover link flooding in our current version.

B. Setup Process

Table I lists the notations used to describe the design of our approach. We will also define them when we refer to them for the first time. There are five sets that should be initialized as the inputs for the assignment/optimization formulation in the centers of the AS. Set of all the routers (R), all the paths (P), and the entire destination IPs (*DestIP*s) (J) are directly available through the traffic matrix and the routing information. As we mentioned earlier, each OD-pair is characterized by its router level path i and the number n_i of IP flows of that path in each measurement duration (e.g. five minutes). Each path i, has a set of routers R^i ($R^i \subseteq R$), that lie on it. Each flow has a *DestIP* from a set of destination IP addresses and there can be more than one flow associated with the same *DestIP* within the same path or among all the paths within the AS. Our goal is to assign all the flows with the same *DestIP* and with at least one common router on their paths, to only one of the common routers in such a way that:

A. All the flows will be assigned to all the routers within the AS, i.e., all the flows are covered.

B. Balance the load, as much as possible, among the routers within the AS.

Table II shows the available data through the routing information and the traffic matrix in an abstract level.

Symbol	Meaning
n_i^j	*Total number of flows in path i with the DestIP equal to j*
n_i	*Total number of flows in paht i*
$\sum_{i \in P} n_i$	*Total number of flows in all the paths i*
R	*Set of all the routers*
R^i	*Set of all the routers lie on path i*
J	*The entire DestIPs within the AS*
$\|J\|$	*Total number of DestIPs within the AS*
P	*Set of all the paths within the AS*
$\|P\|$	*Total number of paths within the AS*
Des	*An array of the entire destination IPs*
Des_j	*The j^{th} element of Des presents the number of paths in which DestIP j appears*
J^{Common}	*Set of all the DestIPs which occur in at least two paths*
L	$L = \{L_1, \ldots, L_{\|R\|}\}$ *Where, $L_k \in L$ and represents the number of paths in which router k appears*
R^{Common}	*Set of all the routers that occur in at least two paths*
Total load	*Total number of flows assigned to the router k*
Balanced load	*Total number of flows / Total number of routers (Within the AS)*

TABLE I. NOTATIONS USED

The set of all the routers which occur in at least two paths (R^{Common}) can be found through a pseudo code given in Figure 2.

TABLE II. AVAILABLE DATA THROUGH THE ROUTING INFORMATION AND THE TRAFFIC MATRIX

Path i	***Routers lie on the path i (R^i)***	***DestIP (J)***	***Number of Flows (n_i^j)***
1	R_1, R_2, R_3	192.168.0.1	4
1	R_1, R_2, R_3	192.168.0.15	1
2	R_1, R_4, R_6, R_2	192.168.0.1	8
2	R_1, R_4, R_6, R_2	10.0.0.2	5
⋮	⋮	⋮	⋮

In Figure 2, L is an array of all the routers. L_k, the k^{th} element of L presents the number of paths in which router k appears. R^{Common} is then calculated as the final set of all the routers that occur in at least two paths.

Finally, the set of all the *DestIP*s which occur in at least two paths (J^{Common}) calculated as shown in Figure 3. In Figure 3, *Des* is an array of the entire destination IPs. Des_j, the j^{th}

element of *Des* presents the number of paths in which *DestIP j* appears.

Algorithm for Generating set of R^{Common}

1. **for** each router $k \in R$ **do**
2. $L_k = 0$
3. **end for**
4. **for** each path $p \in P$ **do**
5. **for** each router k in path p ***do***
6. L_k ++
7. **end for**
8. **end for**
9. **for** each router $k \in R$ **do**
10. **if** $L_k > 1$ **then**
11. $R^{Common} \leftarrow R^{Common} \cup \{k\}$
12. **end if**
13. **end for**

Fig. 2. Set of all the routers which occur in at least two paths (R^{Common}).

Algorithm for Generating set of j^{common}

1. **for** each *DestIP* $j \in J$ **do**
2. $Des_j = 0$
3. **end for**
4. **for** each path $p \in P$ **do**
5. **for** each *DestIP j* in path p ***do***
6. Des_j++
7. **end for**
8. **end for**
9. **for** each *DestIP* $j \in J$ **do**
10. **if** $Des_j > 1$ **then**
11. $J^{Common} \leftarrow J^{Common} \cup \{j\}$
12. **end if**
13. **end for**

Fig. 3. Set of all the DestIPs which occur in at least two paths (J^{Common}).

Generating the sets J^{Common} and R^{Common} have the time complexity O(|*P*|*|*R*|) and O(|*P*|*|*J*|) respectively, in which P is the total number of paths, R is the total number of routers, and *DestIP* is the total number of different destination IPs.

Table III shows the worst case values of *R*, *P*, and *DestIP* for ten real-world topologies (ISPs). The number of routers is from a study in 2002 [25] by Rocketfuel. We also added maximum number of paths by considering all possible pairs of routers (using shortest path routing) as OD pairs (e.g. VSNL ISP at most has 11*11=121 paths). The maximum number of *DestIP*s is equal to maximum number of flows with different destination IPs in each five-minute interval. Hence, like CSamp [23], we also use a baseline traffic volume of 8 million IP flows for Internet2 with 11 points of presence (PoPs) (per five-minutes) to scale the total number of flows by the number of routers in each of the topologies. For instance, Telstra with 345 routers has $\frac{345}{11} \times 8 = 250$ million flows; hence, it has at most 250 million different *DestIP*s.

C. Assignment/Optimization Formulation

Our objective is to minimize the gap between the loads of all the routers within the AS and the balanced ideal load. We define the balanced ideal load as follows:

TABLE III. THE PARAMETER VALUES FOR 10 REAL-WORLD TOPOLOGIES

AS #	*Name*	*Number of Routers (R)*	*Max Number of Paths (P)*	*Max Number of DestIPs*
1221	Telstra (Australia)	345	120K	250×10^6
1239	Sprintlink (US)	471	220K	342×10^6
1755	Ebone (Europe)	133	17K	96×10^6
7018	AT&T (US)	487	237K	354×10^6
3356	Level3 (US)	624	389K	453×10^6
2914	Verio (US)	869	755K	632×10^6
3257	Tiscali (Europe)	247	61K	179×10^6
3967	Exodus (US)	157	24K	114×10^6
4755	VSNL (India)	11	121	8×10^6
6461	Abovenet (US)	357	127K	259×10^6

$$Balanced\ Load = \frac{\sum_{i \in P} n_i}{|R|} \quad (1)$$

Furthermore, we define total number of flows assigned to router k by:

$$Total\ load_k = \sum_i \sum_j n_i^j * x_{ijk} \qquad \forall k \in R \quad (2)$$

Where,

$$x_{ijk} = \begin{cases} 1 & \text{if flows with the DestIP } j \text{ of Path } i \text{ are assigned to router } k \\ 0 & \text{Otherwise} \end{cases}$$

Hence our objective is:

$$Minimize \sum_{k=1}^{|R|} |Total\ load_k - Balanced\ Load| \quad (3)$$

We formulate a Linear Programming (LP) for our assignment/optimization as follows. We have fed LP formulation in the centers with the sets we already created in the setup process and the outcome will be the responsibilities for each of the routers by minimizing our load balancing objective.

$$Minimize \sum_{k \in R} Z_k^1 - Z_k^2,$$

Subject to

$$\sum_{k \in R^{Common}} x_{ijk} = 1, \quad \forall i \in P, \forall j \in J^{Common} \quad (4)$$

$$\sum_{k \in R^i} x_{ijk} = 1, \qquad \forall i \in P, \forall j \in J \quad (5)$$

$$x_{ijk} = x_{i'jk}, \quad \forall i, i' \in P, \forall j \in J^{Common}, \forall k \in R^{Common} \quad (6)$$

$$\sum_{k=1}^{|R|} \sum_{j=1}^{|J|} \sum_{i=1}^{|P|} n_i^j * x_{ijk} = \sum_{i=1}^{|P|} n_i \quad (7)$$

$$\sum_{j\in J}\sum_{i\in P} n_i^j * x_{ijk} + (Z_k^1 - Z_k^2) = \frac{\sum_{i=1}^{|P|} n_i}{|R|}, \qquad \forall k \in R \quad (8)$$

$$Z_k^1, Z_k^2 \geq 0, \qquad \forall k \in R \quad (9)$$

$$x_{ijk} \in \{0,1\}, \qquad \forall i \in P, \forall j \in J, \forall k \in R \quad (10)$$

Here, we briefly explain each of the constraints in our LP formulation.

(4) Ensures that for each path and all *DestIPs* of the flows that occur in at least two paths, all those flows with the same *DestIP* should be assigned to exactly one of the common routers, i.e. flows with the same *DestIP* cannot be split among common routers.

(5) This constraint ensures that all the flows' *DestIP*s of all the paths will be assigned to the routers lie on that particular path. This constraint at the same time ensures that all the flows with the same *DestIP* will be assigned to exactly one router, and cannot be split among common routers.

(6) It ensures that flows with the same *DestIP* in different paths will be assigned to only one of the common routers of those paths.

(7) This constraint ensures that all the flows of the particular path will be assigned to the routers on that path. In other words, the total number of flows in a particular path that are being assigned to the different routers should be the same as the total number of flows in that path.

(8) This constraint enables us to avoid a nonlinear objective function that includes absolute values; we impose these constraints using the variables defined in (9) so that the resulting formulation is linear.

(9) Z_k^1, Z_k^2 are two positive variables we defined for each router k to implement absolute function in our objective function (i.e. these variables are used to liberalize the objective function of the model).

(10) Shows that the decision variables (x_{ijk}) are binary variables.

The outcome of our LP formulation provides the routers' responsibility lists considering our load balancing objective. The promising goal of our assignment/optimization is to find a feasible assignment of all the flows to all the routers, assigning those with the same *DestIP*s to the common routers. In order to do this, we try to satisfy a simple load balancing objective, minimizing the difference between the load of each router and the average load of the network of interest (*Balanced Load*).

Figure 4 shows the outcome of our LP formulation for a sample AS with 5 routers, 3 paths, and 62 flows. There are two destination IPs which are common between paths 1 and 2. Routers R1, R2, and R3 are three common routers between three paths. Our LP formulation's main goal in figure 4 is to assign all the flows with the destination IP A in paths 1 and 2 to one of the common routers R3, R4, or R5 and all the flows with the destination IP C to one of the remaining common routers. Then it assigns all the remaining flows to other routers within the AS in such a way that it satisfies the load balancing objective which is to minimize the gap between the loads of all the routers within the AS and the balanced ideal load.

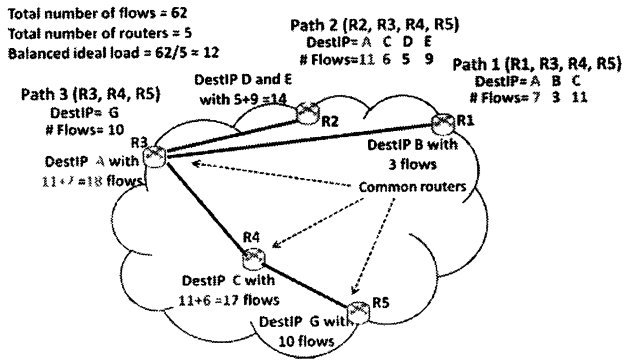

Fig. 4. Sample run of our assignment/optimization formulation

The nearer to the destination the flows get, the more chances are to assign those flows with the same destination IPs to the common routers. Hence, there might be no common router among all the paths to the same destination at the ASs closer to the source. In that case, our approach just fulfils the load balancing objective and assigns all the flows evenly among all the routers within those ASs. As we mentioned earlier, it is still worth to detect a fraction of attack flows and stop them earlier at ASs closer to the source.

IV. Experimental Results

In this section, we demonstrate how we implemented our assignment/optimization formulation using CPLEX. Our main goal was to verify the time feasibility of our designed formulation. We have considered three real-world topologies namely VSNL (India), Ebone (Europe), and AT&T (US). In each topology we assume:

1- Half of the maximum possible number of paths already presented in Table III (*e.g.* 60 paths for VSNL (India)).

2- Half of the maximum number of flows (*e.g.* 4 million flows for VSNL (India)).

3- Half of the destination IPs occur in at least two paths.

To estimate the total completion time (T_{total}) for our scheme:

Let (T_O) be the time it takes for the centers to compute the responsibility lists of the routers via our assignment/optimization formulation.

Let (T_C) be the time each center needs to send the responsibility lists to all the routers within the AS.

$$T_{total} = T_O + T_C \quad (11)$$

Table IV shows T_O for each topology which we computed through our implementation in CPLEX. The result on Table IV validates the feasibility of our approach and verifies that it can react to the dynamic nature of the network in near real-time.

We have also increased the number of common *DestIPs* in each topology in our measurement, in order to see if it affects the performance of our approach drastically. The outcome of our measurements showed that the number of common *DestIPs* does not affect the performance of our approach.

TABLE IV. THE TIME FEASIBILITY FOR 3 REAL-WORLD TOPOLOGIES

Topology	*Number of Routers*	*Time (Sec.)*
VSNL (India)	11	1.13
Ebone (Europe)	133	8.52
AT&T (US)	487	29.7

We estimated the worst case value of T_C for each topology by considering the formulation below:

$$T_C = \frac{Max\ (RTT)}{2} * \frac{Max\,(Size\ of\ Responsibility\ \ list\,)}{The\ size\ of\ RTT\ packet} \qquad (12)$$

Where,

RTT is the Round-trip time within each AS.

We believe that total completion time of our scheme even by considering the worst case scenario is still considerably lower than the time it takes for CSamp to perform the same intrusion detection and response approach; CSamp computes the manifests for all the routers, distributes the manifests among all the routers, and then it should centrally gather the sampled data from all the routers within each AS to further analyze, detect, and respond. Our scheme, as mentioned earlier, computes the responsibility lists and distributes them among all the routers, and then each router is capable of sampling, analyzing, detecting, and responding to the attacks without any further communication overheads. There are still possible improvements to our current version including: considering the router's constraints in such a way that we can guarantee that all the flows will be covered by the routers within an AS, covering link flooding attacks, pre-computing different possible route change scenarios to decrease the delay to compute new responsibility lists.

V. CONCLUSION AND FUTURE WORK

In this paper, we have introduced a collaborative IPS to distribute the sampling, detection, and response responsibilities among all the routers within the AS in such a way that each router can detect and respond to DDoS attacks. Our proposed approach achieves coordination, to eliminate redundant sampling, detection, and response, among all the routers in the network without exploiting any specific communication protocol. In doing so, our assignment/optimization formulation assigns disjoint flows to each of the routers within the AS in such a way that all the flows destined for the same host will be assigned to be sampled, analyzed, and properly responded at the same router. Therefore, the correlation between those flows can be captured by that specific router.

Our future research plan is to implement our collaborative approach fully and study how detection and response approaches will benefit from that. We would like to study real DDoS attack data to see how many attacks can be detected in each router which is facilitated by our collaborative approach. Facilitating routers within the AS to cover link flooding attacks that aim to congest a link or disrupt all the communications via that link (e.g. sending attack packets to a range of destination addresses or a subnet) is a future endeavor.

ACKNOWLEDGMENT

Special thanks to anonymous reviewers for their insightful comments and feedback. This work has been supported by NSF award CCF- 0720737.

REFERENCES

[1] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communications Review*, vol. 34, no. 2, pp. 39-53, April 2004.

[2] L. C. Chen, T. A. Longstaff, and K. M. Carley, "Characterization of defense mechanisms against distributed denial of service attacks," *Computers & Security*, vol. 23, no. 8, pp. 665-678, December 2004.

[3] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of DDoS attacks for large-scale internet," *Comput. Netw.*, vol. 51, no. 18, pp. 5036--5056, 2007.

[4] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks that employ IP source address spoofing," Internet RFC 2827, 2000.

[5] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the Source," in *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, Washington DC, USA, 2002.

[6] T. M. Gil and M. Poletto, "MULTOPS: a data-structure for bandwidth attack detection," in *SSYM'01: Proceedings of the 10th conference on USENIX Security Symposium*, Washington D.C., USA, 2001.

[7] E.Y.K. Chan et al., "Intrusion Detection Routers: Design, Implementation and Evaluation Using an Experimental Testbed," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1889 - 1900, 2006.

[8] R. Chen, J.-M. Park, and R. Marchany, "RIM: Router interface marking for IP traceback," in *IEEE Global Telecommunications Conference (GLOBECOM '06)*, 2006.

[9] A. John and T. Sivakumar, "DDoS: Survey of Traceback Methods," *International Journal of Recent Trends in Engineering ACEEE (Association of Computer Electronics & Electrical Engineers)*, vol. 1, no. 2, May 2009.

[10] R. Mahajan et al., "Controlling high bandwidth aggregates in the network," *SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 3, pp. 62-73, 2002.

[11] A. Yaar, A. Perrig, and D. Song, "Pi: A Path Identification Mechanism to Defend against DDoS Attacks," in *IEEE Symposium on Security and Privacy*, 2003, p. 93.

[12] M. Sung and J. Xu, "IP Traceback-Based Intelligent Packet Filtering: A Novel Technique for Defending against Internet DDoS Attacks," *IEEE Transactions on Parallel and Distributed Systems*, pp. 861-872, September 2003.

[13] X. Liu, X. Yang, and Y. Lu, "To filter or to authorize: network-layer DoS defense against multimillion-node botnets," in *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, NY, USA, 2008, pp. 195-206.

[14] K. Argyraki and D. R. Cheriton, "Scalable network-layer defense against internet bandwidth-flooding attacks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 4, pp. 1284-1297, 2009.

[15] T. Anderson, T. Roscoe, and D. Wetherall, "Preventing Internet denial-of-service with capabilities," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 39-44, 2004.

[16] B. Parno et al., "Portcullis: protecting connection setup from denial-of-capability attacks," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 289-300, 2007.

[17] X. Yang, D. Wetherall, and T. Anderson, "TVA: a DoS-limiting

network architecture," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1267-1280, 2008.

[18] A. Yaar, A. Perrig, and D. Song, "SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks," in *IEEE Symposium on Security and Privacy*, 2004.

[19] X. Liu, A. Li, X. Yang, and D. Wetherall, "Passport: secure and adoptable source authentication," in *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, San Francisco, CA, USA, 2008, pp. 365-378.

[20] X. Yang, "A DoS Limiting Network Architecture," http://www.cs.duke.edu/nds/ddos/.

[21] M. R. Sharma and J. W. Byers, "Scalable Coordination Techniques for Distributed Network Monitoring," in *Proc. of PAM*, 2005, pp. 349-352.

[22] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, 2004.

[23] V. Sekar et al., "CSAMP: a system for network-wide flow monitoring," in *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, San Francisco, CA, USA, 2008, pp. 233-246.

[24] A. Shaikh and A. Greenberg, "OSPF Monitoring: Architecture, Design and Deployment Experience," in *NSDI*, 2004.

[25] Spring N., R. Mahajan, and D. Wetherall, "Measuring ISP Topologies With Rocketfuel," in *ACM SIGCOMM*, 2002.