Finding DDoS Attack Sources: Searchlight Localization Algorithm for Network Tomography

Omer Demir
Department of Information Technology
Turkish National Police, General Directorate of Security
Ankara, Turkiye
Email: omerdemirkos@gmail.com

Bilal Khan
Department of Mathematics & Computer Science
John Jay College, CUNY
NewYork, USA
Email: bkhan@jjay.cuny.edu

***Abstract*—Among the challenges facing the Internet, DoS/DDoS are a critical concern for Internet Service Providers. DDoS attacks can cause country-wide infrastructure problems, and can disrupt communications on a national level. Frequently, Botnets are used to carry out source-spoofed DDoS attacks. The problem of tracing such attacks has been the subject of significant inquiry.**

Here, we leverage the fact that a Botnet requires significant exposure to risk, and investments of time and resources. Thus, as a capital resource, it is likely that a Botnet will, over its lifespan, be used to execute *multiple* criminal DDoS attacks on different targets. Here, we report on new techniques that leverage information obtained over *sequences* of source spoofed Botnet-led DDoS attacks, demonstrating the efficacy of these techniques at pinpointing potential attacker locations.

DDoS attack flow descriptions can be collected in many ways, using a coordinated DDoS sensor agents (e.g. as described by the authors previously in [1]). Here, as a theoretical contribution, we provide formal statement of the attacker localization problem. We develop an new algorithm for localizing attack sources from sequences of DDoS attacks.

***Index Terms*—DDoS; source localization; source spoofing.**

I. Background

Denial of service (DoS) occurs when legitimate users are prevented from getting access to shared resources or services [2]. When a DoS attack is mounted concurrently from large numbers of distributed sources, it is called a Distributed Denial-of-service (DDoS) attack. Although DoS/DDoS attacks are just one of many challenges facing the Internet, they have been identified as a critical concern by Internet Service Providers (ISPs), see e.g. Arbor Networks survey [3].

The damage caused by DDoS attacks can vary in scale, ranging from specific to widespread. Attacks can cause country-wide infrastructure problems which can disrupt communications on a national level. A recent example of an attack at this scale was seen on June 25th 2008, when Radio New Zealand International reported that an attack on the National Telecommunications Authority–the monopoly Internet provider of Marshall Island–caused a complete shutdown of email traffic to the country for a week [4]. Attack dynamics frequently cross national boundaries. One such case included the web site of Rapid Satellite of Miami, a US company in Florida, which was attacked by European hackers who may have been hired by one of its competitors [5].

Core features of the Internet's design have implications for the feasibility of DoS/DDoS and the challenges in its mitigation: (i) Packet switching makes IP traceback difficult; (ii) core Internet components lack the capability to provide sufficient security and authentication since they must process high volumes of traffic, security-related computations are left to edge components. The core provides high volume capabilities, but it is too late to act against DDoS at the victim side! Finally, the Internet's distributed and heterogeneous administration makes deployment of DDoS defense mechanisms difficult.

Attackers typically require bandwidth amplification to carry out a DDoS attack. In a model of **unwitting accomplicies**, regular machines behaving according to poorly designed protocols are somehow orchestrated by attackers so that they cause DDoS. Examples of this phenomenon include SYN ACK and ICMP echo reply floods. Alternately, users may unwittingly surrender their machines by clicking on some link or installing malicious programs. In such cases, a malicious "Bot" is installed on the compromised computer, turning it into a **dedicated attacker** (see, e.g. Stacheldraht [6]).

Key observation. We start from the observation that a malicious attacker is unlikely to instrument a Botnet merely to execute a single attack. This is because the construction of a Botnet requires significant investment of time and resources, exposing criminal elements to significant risk. Once established, a Botnet represents a capital investment from which revenue can be derived, thus making it very likely to be used to execute *multiple* attacks over time.

Previous work. Several attack source identification and attack traceback techniques have been proposed like Backscatter [7], controlled flooding [8], router-centric traceback [9], iTrace [10], packet marking [11], and hash-based traceback [12]. In 2010, the authors described a distributed system capable of traceback of malicious flow trajectories in the wide area despite source IP spoofing. The system requires the placement of traffic volume sensing agents (hereafter referred to as "DDoS sensors") on a small subset of the inter-autonomous system (AS) links of the Internet. Once deployed, these agents are able to respond to DDoS attacks, as a self-organizing system capable of reconstructing topological and temporal information about the structure of malicious flows. The system effectively recovered DDoS flow structure even with very moderate sensor deployment levels (on 10%-15% of the links) in the current Internet topology [1].

Motivating Question. The question naturally arises as to whether there is some way in which these DDoS sensor agents can leverage information collected over longer timescales to further determine the most probable locations of malicious nodes. Stated as a concrete example, in December 2010, when the hacker group Anonymous launched attacks on PayPal, Visa, Mastercard, Amazon, the Swedish prosecutor's office, etc. [13], could information from the *sequence* of multiple attacks have been correlated in a manner to provide information about the location of the attacking nodes? In this paper, we present new techniques that leverage information obtained over *sequences* of source-spoofed DDoS attacks, showing how techniques can pinpoint potential attacker locations.

Assumptions. In developing our approach, we will assume a continuity in the composition of the set of attacking nodes. That is to say, we will assume that the same set of attackers (Bots) carry out a sequence of attacks on different victims, over time. We count each attack against a victim as a separate one. We will not consider changes in membership of the set of attackers, nor will we consider changes to network topology, or changes to the routing table. This idealized setting will allow us to evaluate the effectiveness of proposed schemes in aggregating attacker location information from multiple attacks over long time scales. There are currently no such systems to our knowledge which combines attack information collected from self organizing agent systems to determine attacker locations. In this sense our system is a novel system in aggregating attacker location information to determine the actual locations of attackers. In future, we plan to quantify the sensitivity of our approach to these assumptions.

II. Mathematical Model

Let $G=(V,E)$ be a **network** on nodes V and $E \subset V \times V$ be a set of bidirectional **links**. The **routing tables** are modeled as a function $R : V \times V \to V$ which is consistent with the link relation E. For vertices $v, d \in V$ the n^{th} **flowstep** from d to v, is defined inductively (for $n \geqslant 0$) as $f(d,v,0)=d$, and $f(d,v,n+1)=R(f(d,v,n),v)$. The sequence of flowsteps $F(d,v)=(f(d,v,i); i=0,1,...)$ is called the **flow** from d towards v with respect to routing table R. A routing table R is **consistent** if for all d,v in V the flow $F(d,v)$ is eventually v; it is called **symmetric** if for all u,v in V, the flow $F(u,v)$ is the reverse sequence of the flow $F(v,u)$. Here we assume a symmetric, consistent routing table R on a network G.

A **flow reconstruction problem** (FRP) is a tuple (A,D,v) where $D \subseteq V$ is a set of attacking nodes, $v \in V$ is the DDoS victim, and $A \subseteq E$ is the set of DDoS sensor agents. Figure 1 illustrates a sample flow reconstruction. Where $D=(A_1,A_2,A_3,A_4)$, v is the victim, smiling faces are agents and dotted lines are logical links between the agents. A **valid solution** to an FRP instance is a logical network $L=(S,E_S)$ on a subset of agents $S \subset A$ and $E_S \subset S \times S$, where (1) every agent in the solution set S lies on the flow from some attacker to the victim; (2) if two agents are connected by a logical link in L then they appear successively in the flow from some attacker to the victim (and there exists no agent in between them). A solution $L=(S,E_S)$ is said to be **maximal** if additionally (1) every agent that lies on the flow from some attacker to the victim is in the solution set S; (2) if two agents are connected by a logical link in L then they appear successively in the flow from some attacker to the victim (and there exists no agent between them in the flow). In our previous work [1], we presented methods for computation of maximal solutions to the distributed instances of FRP, using a distributed deployment of DDoS sensors. In particular, the authors proved that these sensors (and the associated distributed protocols) would successfully self-organize in response to the attack, and coordinate to produce maximal valid solutions to the FRP instance. We also provided strong experimental metrics for the system's performance [14]. Here we assume such a sensor system (or a functional equivalent) that is capable of producing descriptions of attack flows.

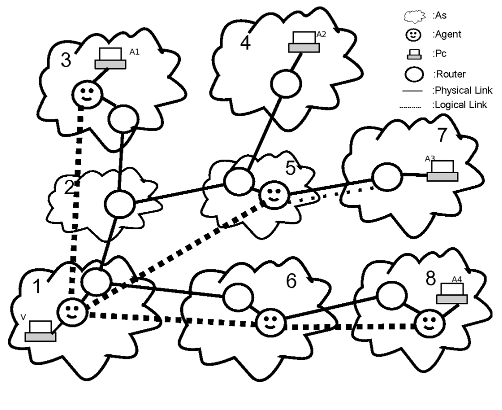

Fig. 1. Sample flow reconstruction

For a fixed network G and consistent symmetric routing table R, an instance $\mathcal{P}$ of the **attacker localization problem** (ALP) is a set of FRPs, $\mathcal{P}=(P_1,P_2,...,P_n)$ where the $P_i=(A,D,v_i)$ differ only in the victim identity v_i, together with a set of corresponding maximum valid solutions $\mathcal{L}=\{L_i=(S_i,E_{S_i}) \mid i=1,\ldots,n\}$. A solution to an instance of ALP is a **suspicion map**, $s : V[G] \to \mathbb{R}$ assigning a "suspicion level" to each vertex in V. Note that for every fixed threshold $\tau \in \mathbb{R}$, the solution s can be thought to assert that the vertices $SS(\tau)=\{v \in V | s(v) \geqslant \tau\}$, are justifiably "$\tau$-level **suspects**" in the attack sequence $\mathcal{P}$. We call $SS(\tau)$ the **suspicion set** (at level τ).

Given this formal structure, how might we assess a solution (s,τ) to an instance $(\mathcal{P},\mathcal{L})$ of ALP? Fixing a suspicion threshold τ, we seek to minimize the fraction of suspects who are actually innocent (i.e. the false positive rate):

$$FPR(\tau)=\frac{|\{v \in V \backslash D | s(v) \geqslant \tau\}|}{|\{v \in V | s(v) \geqslant \tau\}|+\epsilon},$$

where $\epsilon > 0$ is a fixed very small constant that prevents denominator from being zero. In addition, we seek to minimize the fraction of attacking nodes who escape being suspect (i.e.

the false negative rate):

$$FNR(\tau) = \frac{|\{v \in D | s(v) < \tau\}|}{|D|}$$

We expect a natural trade off between the two measures above, since it is easy to achieve either very low FPR or very low FNR. To see this, consider a function s which assigns all vertices in V a suspicion score of 1; then FPR(2)=0, while FNR(0)=0. In general, however, the problem of finding a solution s that achieves low FPR(τ) and FNR(τ) simultaneously for the *same* value of τ, is likely to be very difficult. An algorithm that achieves this can be useful for locating sets of attackers who collaborate repeatedly to attack different victims over time. This is what we seek to achieve.

III. The SLANT Algorithm

Next we present a formal description of the **Searchlight Localization Algorithm for Network Tomography** (SLANT) to solve ALP instances. We will also describe a (preliminary) centralized implementation of SLANT.

Consider a network $G = (V, E)$ and consistent, symmetric routing table R. For every pair $a \in A, v \in V$ of distinct vertices, we define the cone $C(a, v)$ to be those $u \in V$ for which the flow from u to v (with respect to R) passes through a. Formally, $C(a, v) = \{u \in V | a \in F(u, v)\}$. If agent a witnesses a DDoS attack on v, then $C(a, v)$ represents a set of candidate locations for attackers.

Given an instance of ALP, $(\mathcal{P}, \mathcal{L})$, consider the ith constituent FRP instance $P_i = (A, D, v_i)$, together with its corresponding solution $L_i = (S_i, E_{S_i})$. Relative to each agent $a \in S_i$ we may compute the cone $C(a, v_i)$. Two questions arise: (i) How should the cones $\{C(a, v_i) \mid a \in S_i\}$ be combined to produce the suspicion map s in response to attack i? (ii) How should changes in the suspicion function in response to successive attacks be accumulated, over the sequence $(i = 1, \ldots, n)$ of attacks?

Rather than committing to any single answer, we provide a general framework for answering these questions. Regarding (i), the cones $\{C(a, v_i) \mid a \in S_i\}$ obtained by analyzing attack i are to be combined by defining

$$\Delta s(u, i) = \sum_{a \in S_i} \sigma(u, C(a, v_i), S_i),$$

for each $u \in V$, where σ is the **multi-agent information combining function**. In this paper, we take

$$\sigma(u, C, S) \quad = \quad \begin{cases} 1/|S| & \text{if } u \in C \\ 0 & \text{otherwise.} \end{cases}$$

Regarding (ii), changes in the suspicion function $\Delta\tau(u, i)$ are accumulated over the sequence $(i = 1, \ldots, n)$ as

$$s(u) = \sum_{i=1}^{n} \rho(\Delta\tau(u, i), n),$$

where ρ is the **multi-attack information combining function**. In this paper, we take $\rho(x, n) = x/n$. Informally, our choice of ρ considers each separate DDoS attack additively. Our choice of σ ensures each vertex accumulates a suspicion score equal to the *fraction* of agent cones in which the vertex lies, normalized by the total number of attacks n. Thus, roughly speaking, a vertex achieves a suspicion score of 1 if and only if it is in every agent cone reported in every attack. Given ALP instance $(\mathcal{P}, \mathcal{L})$ the **SLANT algorithm** operates by:

- Defining $s(v) := 0$ for all v V.
- For each attack i, where $i = 1, \ldots, n$:
  - For each DDoS sensor agent a in S_i, compute the cone $C(a, v_i)$.
  - Then,for all $u \in V$, compute
    $$\Delta\tau(u, i) = \sum_{a \in S_i} \sigma(u, C(a, v_i), S_i)$$
  - Update $s(v) := s(v) + \Delta\tau(u, i)/n$.
- Output the suspicion map s.

IV. Experiments on the Grid

To illustrate the effectiveness of the $SLANT$ algorithm, we will show its operation in a toy scenario of a grid network with *one* attacker $|D| = 1$. We use an $\sqrt{n}$ by $\sqrt{n}$ vertex network, structured as a Cartesian lattice of links. The routing table R is constructed by running Dijkstra's algorithm from all nodes. After creating the vertices and the links we build the DDoS sensor set A by placing an agent along every k^{th} horizontal and every k^{th} vertical link, resulting in an agent density of $1/k$. For example, when $k = 2$, the resulting DDoS sensors A have density equal to 50% (of the links). After that we uniformly randomly (without replacement) select the attackers from the original n nodes, forming the set D. In our experiments, $|D| = 1$ or 10. We then chose a sequence of n victims $v_1, \ldots v_n$, uniformly at random (with replacement). In the experiments, the number of DDoS victims n varies from 10 to 80. Note that the input to a grid experiment is completely characterized by $|V|$, $|k|$, $|D|$ and n.

Once the experiment setup is specified, the simulation operates as follows. The attackers D sequentially attack the specified random sequence of victims $v_1, \ldots v_n$, yielding n FRP instances. We consider maximal valid solutions to these FRP problems (relative to the DDoS sensor agents A). Together, the n problems and solutions constitute an instance of ALP. The SLANT algorithm operates on the ALP instance to produce a suspicion map s. Given the suspicion map s, we consider the functions $|SS(\tau)|$ and $FNR(\tau)$ to evaluate the performance of SLANT; parameters varied are k, n, $|V|$.

A. Visualizing $SLANT$ localization

We illustrate how the suspicion map changes as the number of attacks n progresses. We use a 100 node grid topology, fixing agent density at 50%. There is exactly one attacker ($|D| = 1$) at coordinates $(7, 2)$. The sequence of victims attacked were chosen uniformly at random on the grid. Figure 2 shows the sequence of suspicion maps s after $n = 1, 4$ and 10 attacks have been conducted. The color distribution represents the suspicion scores, normalized against the maximum value of s (over all network vertices). The sequence of figures

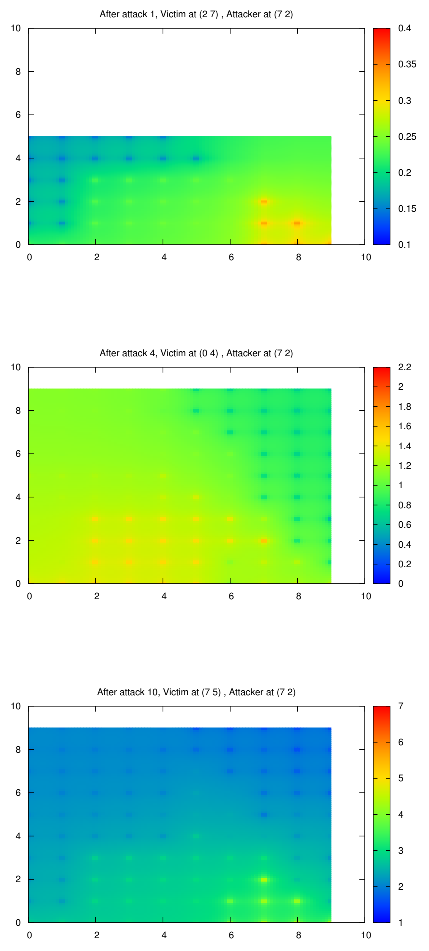

Fig. 2. Progressive localization as more DDoS attacks occur.

illustrates the operation of the $SLANT$ algorithm and gives an intuition behind its promise. We now proceed with a more systematic and quantitative evaluation of SLANT.

B. Varying network size

Purpose. The purpose of this experiment is to see how $|SS|$ and FPR vary with network size. We used grid networks, varying network size $|V|$ between 100 and 1000. We fixed $|D| = 1$, $n = 80$ attacks, $k = 2$.

Results. Figure 3 shows τ vs. $|SS|$ and FPR. In all graphs that follow, we show both curves on the same graph by plotting $10 * FPR$ and $|SS|$. In the graphs: as the threshold value increased, the FPR and the number of vertices in the suspicious set SS decreased. The sole attacker clearly always has the highest suspicion score (since there are no increases in the FPR curve). The $|SS|$ curve also monotonically decreases– expected behavior, since the suspicion scores assigned to vertices do not change as we increase the threshold. Between $\tau = 0.041$ and $\tau = 0.049$, the one vertex in the SS is the true sole attacker. This experiment shows that in this case, the $SLANT$ algorithm could find exactly who the real attacker was. An analogous result holds for the larger 1000 node network: exact localization is achieved for $\tau \in [0.024, 0.025]$.

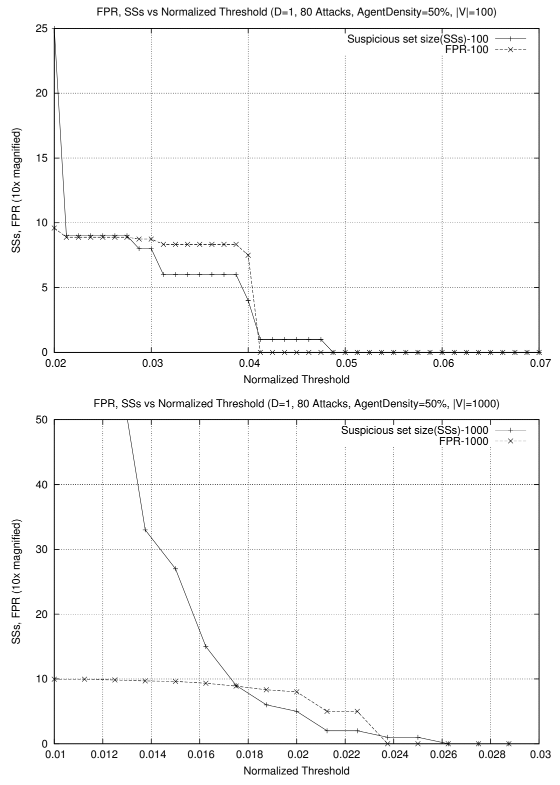

Fig. 3. SLANT on grid networks (100,1000 nodes)

C. Varying the number of attacks

Purpose. The purpose of this experiment is to understand the effect of number of attacks n has on $|SS|$ and FPR. We used grid networks, varying the network size $|V| = 100, 1000$ and taking the number of attacks $n = 10, 40$. We fixed the number of attackers $|D| = 1$, and set $k = 2$.

Results. Figure 4 shows the experiment results for different number of attacks and network sizes. The four figures consider network sizes of 100 and 1000, after 10 or 40 attacks.

From the figures, we see that in graphs 1,3 and 4, there is a point when the FPR is 0 and $|SS|$ is 1. Thus, in these cases, we can exactly identify the true sole attacker. Graph 2 shows that when network size is 1000 and the number of attacks is small (10) we can not exactly determine the attacker, since SS drops to 0 before FPR becomes 0. However as the number of attacks increases to 40 (in Graph 4) we can determine the attacker's location precisely. These experiments demonstrate that as the network size increases we need to have more attacks to exactly determine the attacker. The $SLANT$ algorithm's success depends on the number of attacks n (more are needed as the network size increases).

V. Experiments on General Networks

Next we examine the performance of SLANT in more general artificial networks, and when more than one attacker

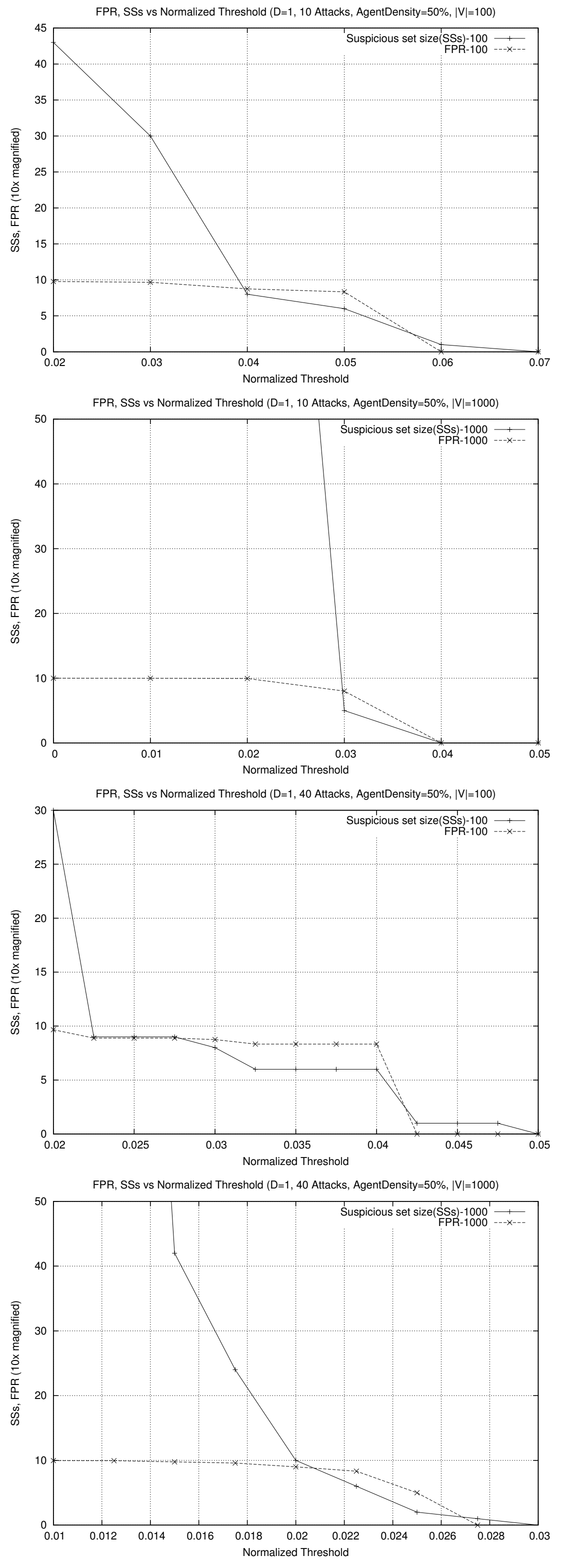

Fig. 4. SLANT on grid network (100,1000 nodes; 10,40 attacks)

is present. For these experiments the networks are devised following Waxman's model [15]. Within a geometric square of side $5\sqrt{|V|}$, we place $|V|$ nodes uniformly at random. Then we add random links between nodes, with probability inversely proportional to an exponential function of the Euclidean distance between the two nodes. Link addition continues until the desired edge density is reached. In our experiments, we took the edge density to be 2.1; this value was chosen because it is close to the value obtained through analysis of the present Internet topology [16]. The routing table R is established by running Dijkstra's algorithm from all nodes. The DDoS sensor set A is created by splicing them into $\epsilon|E|$ edges, chosen using the M3-greedy placement algorithm described in [17]. After that we randomly select the $|D| = 1$ or $|D| = 10$ attackers, forming the set D. We then chose a sequence of n victims $v_1, \dots v_n$, uniformly at random. In the experiments, n varies from 10 to 80. Note that a Waxman experiment is characterized by $|V|$, ϵ, $|D|$ and n. The simulation operates as in the case of grid experiments. The two metrics considered are $|SS|$ and FPR; parameters varied are $|V|$ and $|D|$.

A. Localization in a Waxman network with one attacker

Purpose. The purpose of this experiment is to determine how well SLANT performs in Waxman type networks when there is one attacker. We use a Waxman network, varying $|V| \in \{200 and 800\}$. The agent density ϵ =15% is more realistic than in the earlier grid experiments, there are $|D| = 1$ attackers, and $n = 80$ attacks.

Results. Figure 5 shows the performance of the algorithm in different sized Waxman networks when there is one attacker ($|V| = 200, 800$ nodes). All the curves decrease monotonically, which agrees with earlier conclusions drawn about the much more restricted class of grid networks. In both graphs, we can see that there is a point where $FPR = 0$ and $|SS| = 1$. In conclusion, $SLANT$ performs favorably in Waxman network; moreover, it performs better in Waxman networks than in grid networks of comparable size (while requiring much lower levels of DDoS sensor agent deployment).

B. Localization in a Waxman network with many attackers

Purpose. The purpose of this experiment is to determine how well SLANT performs in Waxman type networks when there is more than one attacker. The setup is exactly as in the previous experiment, except that the number of attackers $|D| = 10$, which are randomly placed at the outset.

Results. Figure 6 shows the performance of the algorithm in different sized Waxman networks when there are ten attackers. All of the curves in all of the graphs decrease monotonically. Note that, unlike the cases when there is only one attacker, in principle, the FPR curve does not have to decrease monotonically–there can be cases when it increases: this can happen in cases when an increase in τ removes an attacker from SS. In all figures, the FPR curve goes to 0 before the $|SS|$ curve reaches 0. Note that, unlike the one attacker case, here $|SS|$ can (and often is) greater than 1. Nonetheless, even with an agent density of 15%, $SLANT$ was able to determine locations for several parties involved in the $DDoS$ attacks.

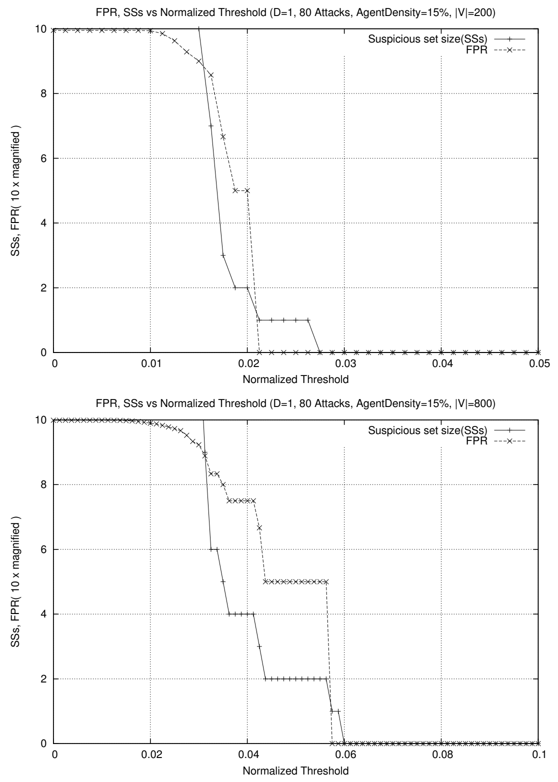

Fig. 5. SLANT on Waxman networks with one attacker (200,400 nodes)

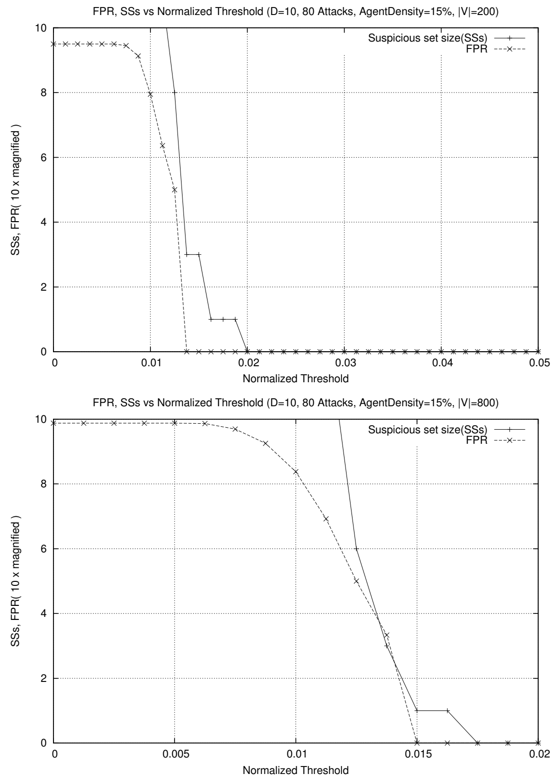

Fig. 6. SLANT on Waxman networks with 10 attackers (200,400 nodes)

VI. Conclusion and Future Work

We have shown that the SLANT algorithm is able to leverage information obtained over *sequences* of source spoofed Botnet-led DDoS attacks. The performance of the algorithm is promising, in terms of its scalability to large networks and numbers of attacking nodes. As such, the SLANT algorithm solves instances of the newly formulated attacker localization problem. SLANT's solutions enjoy low FPR and moderate suspicious set sizes under a suitable choice of threshold τ. This performance is manifested even at modest DDoS sensor agent deployment levels of 15%. As future work, the authors plan to consider the problem of automated selection of good threshold values τ, and a distributed implementation of SLANT.

References

[1] O. Demir and B. Khan, "Reconstruction of malicious internet flows," in *Proceedings of International Wireless Communications and Mobile Computing Conference*, 2010.

[2] V. D. Gligor, "A note on denial-of-service in operating systems," *IEEE Trans. Softw. Eng.*, vol. 10, no. 3, pp. 320–324, 1984.

[3] Arbor Networks, "Arbor networks worldwide infrastructure security report," http://www.arbornetworks.com/en/docman/worldwide-infrastructure-security-report-volume-iv-2008.

[4] T. R. Irirangi and O. Aotearoa, "Marshalls internet still affected after cyber attack," 2008. [Online]. Available: http://www.rnzi.com/pages/news.php?op=read|\&id=40547

[5] J. Kirk and IDG NewsService, "Two europeans charged in US over DDOS attacks," 2008. [Online]. Available: http://www.pcworld.com/businesscenter/article/151829/two_europeans_charged_in_us_over_ddos_attacks.html

[6] D. Dittrich, "The Stacheldraht distributed denial of service attack tool," 1999, accessed Dec. 1, 2010. [Online]. Available: http://staff.washington.edu/dittrich/misc/stacheldraht.analysis

[7] B. Gemberling, C. Morrow, and B. Greene, "ISP security-real world techniques. presentation, nanog," Tech. Rep., 2001. [Online]. Available: http://www.nanog.org/meetings/nanog36/presentations/greene.ppt

[8] Burch and Hal, "Tracing anonymous packets to their approximate source," in *LISA '00: Proceedings of the 14th USENIX conference on System administration*. Berkeley, CA, USA: USENIX Association, 2000, pp. 319–328.

[9] D. Cotroneo, L. Peluso, S. Romano, and G. Ventre, "An active security protocol against dos attacks," *Computers and Communications, IEEE Symposium on*, p. 496, 2002.

[10] Bellovin, "ICMP traceback messages," RFC draft, September 2000. [Online]. Available: 'http://tools.ietf.org/draft/draft-bellovin-itrace/draft-bellovin-itrace-00.txt

[11] S. Stefan, W. David, K. Anna, and A. Tom, "Practical network support for IP traceback," *SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 4, pp. 295–306, 2000.

[12] Snoeren and A. C., "Hash-based IP traceback," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2001, pp. 3–14.

[13] J. Burns and R. Somaiya, "Hackers attack those seen as wikileaks enemies," http://www.nytimes.com/2010/12/09/world/09wiki.html, December 9, 2010.

[14] O. Demir and B. Khan, "Quantifying distributed system stability through simulation: A case study of an agent-based system for flow reconstruction of ddos attacks," in *Proceedings of 1st International Conference on Intelligent Systems, Modelling and Simulation*, 2010.

[15] B. M. Waxman, "Routing of multipoint connections," pp. 347–352, 1991. [Online]. Available: http://portal.acm.org/citation.cfm?id=128991

[16] H. Young, H. Bradley, A. Dan, A. Emile, L. Matthew, and S. Colleen, "The IPv4 routed /24 as links dataset11/15/2009," http://www.caida.org/data/active/ipv4_routed_topology_aslinks_dataset.xml, Accessed December 1, 2010.

[17] O. Demir, "A survey of network denial of service attacks and countermeasures," CUNY Computer Science Department, Tech. Rep., 2009.