Honeypot Based Routing to Mitigate DDoS Attacks on Servers at ISP Level

Anjali Sardana and R. C. Joshi
Indian Institute of Technology, Roorkee, India
anjlsdec@iitr.ernet.in, joshifcc@iitr.ernet.in

Abstract

DDOS attacks generate flooding traffic from multiple sources towards selected nodes and cause obstruction in flow of legitimate information within a network. If the victim node is the server at ISP level requiring fast information processing, the entire network operation stops. We use various lines of honeypot based defense against such attacks. The first line of defense detects the presence of attacks. The second line of defense identifies and tags attack flows in real time. The work in this paper concentrates on the third line of defense, where a model for honeypot based routing has been proposed in response to identified attack flows. We propose the automatic generation of adequate server nodes to service client requests and honeypots to interact with attackers in contained manner. The judicious mixture of servers and honeypots at different time intervals provide stable network functionality at ISP level. We validate the effectiveness of the approach with modeling on Internet type topology and simulation in ns-2 on a Linux platform.

1. Introduction

One of the major problems on the Internet today is denial of service (DoS) [1] attacks against machines and networks. DoS attacks send a stream of packets at a victim that swamps his network or processing capacity, denying access to his regular clients. Distributed Denial-of-Service attacks (DDoS) degrade or completely disrupt services to legitimate users by eating up communication, computational, and or memory resources of the target through sheer volume of packets. DDoS attacks are amplified form of DOS attacks where attackers direct hundreds or even thousands of compromised "zombie" hosts against a single target [2].

Network level DDoS attacks aim at congesting the network, such as link capacity and router buffers, by flooding them with bogus packets. In service level DDoS attacks, a large number of attack machines manage to acquire services from a victim server consuming both service-level resources, such as server memory and processing time, as well as network-level resources along the path outward from the server.

Major portion of bandwidth consists of TCP and UDP traffic. Most of the public servers provide services through TCP [3], so protecting the TCP portion of the bandwidth is sufficient in protecting most of services. Our scheme works at service provider level for detection and redirection and serves on various lines of defense to protect a public domain server. Firstly, entropy variations at a POP identify the presence of attack. Secondly the flows are tagged as attacks in subsequent time windows.

In this paper, we focus on the last line of defense i.e. attack mitigation using honeypot based routing and redirection to protect the information stored on servers. The honeypots change in number providing deterrence from public domain servers. The flows tagged as attacks are directed to honeypots. It ensures smooth information flow for legitimate clients even in network under high attack. Connection retention with attack flow is to obtain information about the attackers by logging their actions. Hence, the model adopts a proactive behavior to circumvent any anticipated attacks.

Honeypots are computer systems placed on a network to attract attackers, allowing administrators to capture and analyze current attack methodologies and use that information to harden their systems and networks. One of the biggest challenges when deploying honeypots [4] is in maintaining the functionality of the total system and information protection on servers as the network behavior changes.

With honeypots [4], one of the configuration issues is how many honeypots to deploy. Also, it must be made sure that honeypots blend into the system and appear like any other entity on the network. Our work addresses these issues as described. Our aim is to protect information on an FTP server from DDoS attacks while providing smooth information flow. FTP services have been replicated on all nodes to make honeypot act like FTP server responding in contained manner to attack flows. The scheme also proposes a means for analyzing traffic, rapidly detecting attacks, and dynamically generating honeypots appropriate for that attack to interact with. Our honeypots are generated on the information gathered about the network to turn it into a set of honeypot definitions, and provides a means of deploying adequate number of honeypots.

We use GT-ITM, an ISP level topology generator and ns-2 as test bed for simulations. We show that entropy captures variable rate DDoS attacks [2] coupled with honeypots providing stable response in attacked network.

The remainder of this paper is organized as follows. Section 2 discusses background and related work. Section 3 describes the environment used as testbed. Section 4

978-0-7695-3151-9/08 $25.00 © 2008 IEEE
DOI 10.1109/ISIP.2008.115

explains the methodology. Section 5 describes simulation experiments and discusses the results. Finally section 6 concludes the paper.

2. Background and Related Work

A commonly used detection approach is either signature-based or anomaly-based. Signature-based approach inspects the passing traffic and searches for matches against already-known malicious patterns. By contrast, an anomaly based detection system observes the normal network behavior and watches for any divergence from the normal profile. Most of existing anomaly based solutions [5]-[9] to detect and categorize attacks use volume based metrics. These suffer from more collateral damage when attack is carried at slow rate. Their normal traffic models are mainly based on flow rates. As a result, legitimate traffic can be classified as attack traffic (false positive) and attacker traffic is classified as legitimate (false negative). To minimize the false positive/negative rate, a larger number of parameters are used to provide more accurate normal profiles. However, with the increase of the number of parameters, the computational overhead to detect attack increases. In practice, several signature-based detection systems such as Bro [10] and Snort [11] have been developed and deployed at firewalls or proxy servers.

Many mitigation approaches have been suggested. In [12], and [13] based on destination address, attack aggregate are found and then filtered using pushback technique. However in this case, collateral damage is more as legitimate traffic in that aggregate is also dropped. A network having proactive server roaming [3] mitigates the attack because the location of the server changes before the attack builds up (connection dropping effect). However, because of the absence of honeypots, the illegitimate traffic remains undirected and no information about the source of attack is logged (no filtering effect).

In a network with fixed static honeypots, the illegitimate traffic is directed to the honeypots and source of attack can be identified and filtered (filtering effect). But if the honeypot is compromised, attacker can use it to attack servers in the network. Also, because of sacrificing some servers to act as honeypots, distributing the load on all the servers outperforms the roaming honeypots scheme in the case of a high legitimate client load combined with a low attack load.

The proposed dynamic honeypots cover the above research gaps and are based on detection and characterization mechanism, as proposed by Sardana et. al. [9]

3. The Environment

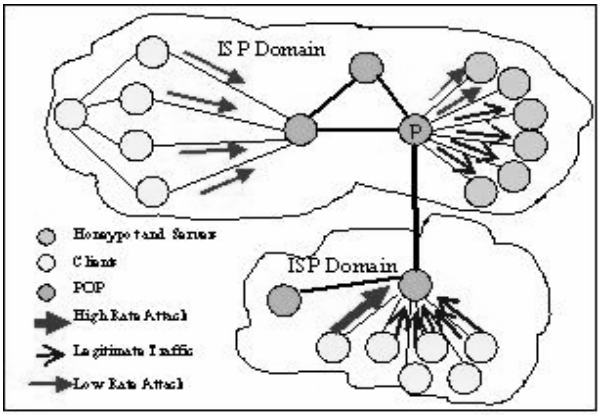

Figure 1. Topology to illustrate the proposed solution

Consider a sample topology shown in Figure 1. The topology considered is similar to the one used traditionally to depict a typical client-server scenario in the Internet. The clients (attack and legitimate) send their FTP requests to the server. The arrows in the figure indicate the presence of variable rate attacks coming from client domain. The set of routers (R) at POP (indicated by P) en route from the clients to the server proactively generate a flow list in moving time window containing information about each arriving flow and the number of packets belonging to that flow. The flow list (FL) undergoes entropic test [9] for attack detection and thus the flows are characterized as attack or legitimate. If a flow is characterized as a legitimate flow, only then will the routers belonging to set P be instructed to forward the packets to the randomly selected active FTP servers. If a flow is characterized as an attack flow, then the flow is routed to one of the randomly selected honeypots.

4. Honeypot Based Routing

Our approach for honeypot generation in response to flows identified as attacks in FL above is generation of required active servers and honeypots to interact in robust and realistic manner.

For simulation purpose, we have simplified ISP level network with each ISP domain having 10 POPs represented by single node each as shown in Figure 2. One customer domain is attached to each POP which consists of legitimate and attacking hosts. ISP domain has additional POP for connecting to protected servers. We assume a pool of N homogenous, geographically dispersed servers. We envision protected service as generic TCP based service that can be replicated.

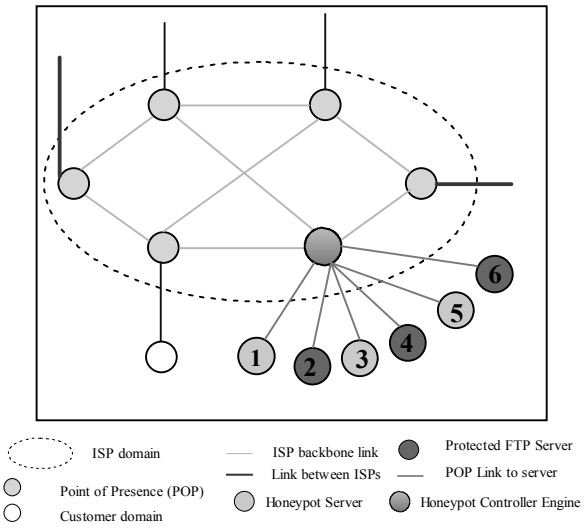

Figure 2. Simulated Topology

Honeypot Controller modeled at POP performs three functions. First, it keeps a track of FL and state of flow. Then it determines the attack load and client load using number of attack and legitimate flows and respective data rates and then triggers the generation of appropriate number of servers and honeypots.

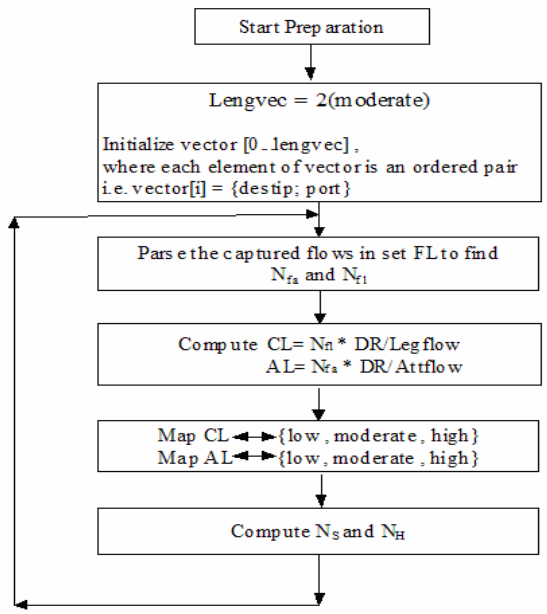

Figure 3. Implementation of dynamic honeypot engine at honeypot controller

Let N_S and N_H represent the number of servers and honeypots respectively. Our aim is to find out the optimum values of N_S and N_H depending upon the network load. We define *lengvec* such that *lengvec* = N_S +N_H. We define an array *vector []* of size *lengvec* whose elements are in the form of ordered pair set of destination IP address and port number of the honeypot or the server i.e. *vector[i] = {dest IP, port}* .We further define two arrays *subvecNS []* and *subvecNH []* whose elements are indices of the array *vector []* that correspond to destination IP address and port number of servers and honeypots respectively such that following holds true: *(lengvec= (Length(subvecNS) + Length(subvecNH))*

Let *CL* and *AL* represent client and attack load. N_{fl} and N_{fa} are number of legitimate and attack flows and *DR/Legflow* and *DR/Attflow* represent data rate per legitimate and attack flow respectively. Figure 3 shows the procedure trigerred once every time eon (a constant predefined time interval).

Table 1 is used for mapping CL and AL to right combination of servers and honeypots, N_S and N_H.

Table 1. Mapping load to honeypots and servers

Attack Load	Client Load	Number of Honeypots	Number of Servers
Low	Low	Low	2 moderate - low
Low	Moderate	Low	2 moderate - low
Low	High	Low	2 moderate – low
Moderate	Low	Moderate	Moderate - low
Moderate	Moderate	Moderate	Moderate
Moderate	High	(Moderate - low ; Moderate)	(Moderate; Moderate + low)
High	Low	2 moderate - low	Low
High	Moderate	(moderate; moderate + low)	(moderate - low; Moderate)
High	High	Moderate	Moderate

Honeypots and servers are randomly selected from the vector. The attack and legitimate traffic is redirected accordingly by variations in the dynamic field like destination address. The controller establishes a dedicated connection and adjusts routing information so that the attack traffic is forwarded to the randomly selected honeypot for interaction. All data from this interaction is logged for later analysis. A higher-level responsibility of the honeypot controller is to manage and prioritize the server set. Figure 4 gives the algorithm that is used for detailed analysis performed on each flow before determining and forwarding it to its appropriate destination.

Service Replication: Each destination in the network should be able to change its behavior according to the corresponding node being tagged as server or honeypot. Thus the services have been replicated on all servers. We use information at FTP which is a TCP based service that can be replicated. An FTP connection remains alive until either the legitimate FTP request is fulfilled completely or the server changes its location. The next active server set is selected randomly from the current pool of servers. An illegitimate FTP request is directed to honeypot which

retains the connection and responds to attacker in contained manner. In case of false positive, the connection is migrated back to active FTP server in subsequent time window. We use 1 Mbps and 310 Mbps links to model access and bottleneck bandwidth. Both clients and attackers request files of size 1 Mbits each with request inter-arrival times drawn from a Poisson distribution. We characterize both client and attack loads by the average bandwidth requested per second.

ALGORITHM : HoneypotControllerPerFlow (FL)

For a flow in FL
If (Tag = attack)
Parse the primary packet and search source
and destination address (F_{DA} and F_{SA})
$P_{DA} = F_{DA}$
$N_{DA} = P_{DA}$
A: If *(N_{DA} = Destination address of honeypot)*
Forward the packet to N_{DA}
Else
Replace N_{DA} by destination address of honeypot
Forward the packet to N_{DA}
If *(More Fragment = 0)*
Goto *S*
Else Parse next header of the flow for P_{DA}
$N_{DA} = P_{DA}$
If *(Tag = attack)*
Goto A
Else Goto B
Else Parse the primary packet and search source
and destination address (F_{DA} and F_{SA})
$P_{DA} = F_{DA}$
$N_{DA} = P_{DA}$
B: If *(N_{DA} = Destination address of active FTP server)*
Forward the packet to N_{DA}
Else
Tag = attack
Forward the packet to N_{DA}
If *(More Fragment = 0)*
Goto *S*
Else Parse next header of the flow for P_{DA}
$N_{DA} = P_{DA}$
If *(Tag = attack)*
Goto A
Else Goto B
S: Stop

Figure 4. Algorithm for honeypot based routing; per flow analysis

5. Results and Discussion

For the purpose of simulation, we use an intelligent mapping to select the active FTP servers for each eon. This prioritizes the selection of server set. The currently active FTP servers may continue servicing legitimate requests with some honeypots replaced by active FTP servers in case the value of N_S increases. This is shown in figure 5.

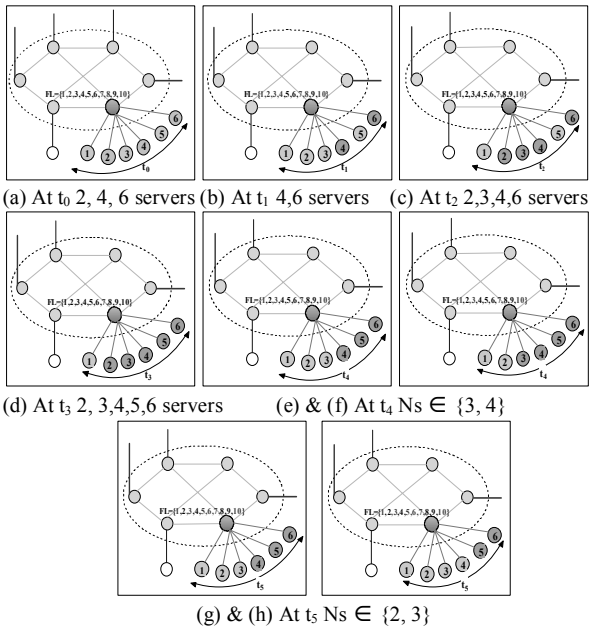

Figure 5. Scenario : Changing location of N_S and N_H

Goodput

Figure 6. shows simulated results for variation in goodput with client and attack load according to scenario in Figure 5. Six cases have been simulated. (a) Ideal goodput values, (b) No DDoS attacks, (c) Best defense (d) Normal defense (e) Naïve defense (f) No Defense.

Figure 6. shows that variation in goodput is independent of attack load because of the presence of honeypots. In case of low client load, and no defense, as the attack load increases goodput decreases. In case of moderate client load and particular value of attack, say moderate attack, best defense gives lower goodput than naïve defense. Increase in attack load decreases the goodput slightly. As the attack load changes, honeypot controller triggers new values of N_S and N_H, thus causing a slightly different and variable mapping of servers. This causes a slight change in goodput. Goodput decreases slightly with increase in attack load for high client load as N_H increases with lesser N_S available to service client requests causing higher drop in legitimate packets at active FTP servers. In case of high client load, best defense gives higher goodput than naïve defense. As the client load is high, a smaller selected set of legitimate client load is sent to limited active FTP servers to be processed efficiently without loss. However, in case of naïve defense, more number of client requests are sent to active FTP servers. It increases the processing load on limited N_S causing packets to drop thus reducing goodput. Intelligent mapping and selection of active FTP servers in every subsequent time eon gives high goodput values.

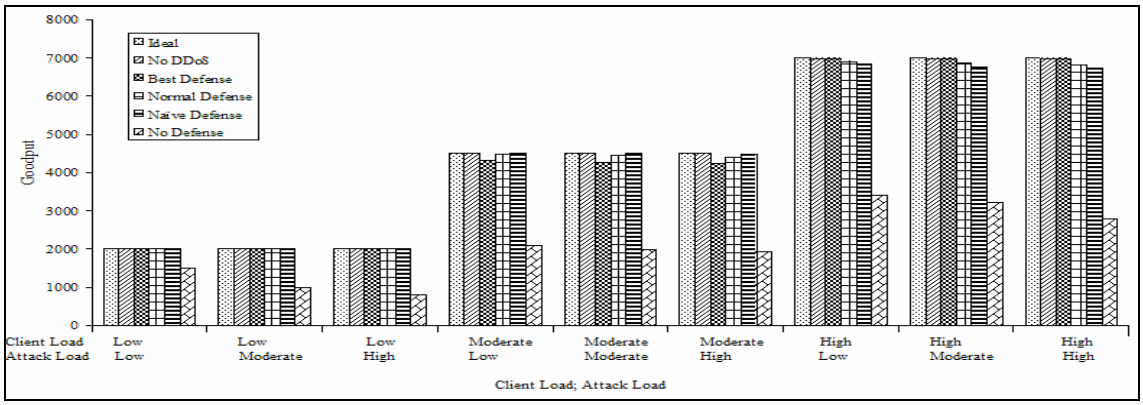

Figure 6. Variation of goodput with varying client load and attack load

6. Conclusions

The approach integrates active real time attack flow identification with determining required number of honeypots. The honeypot controller has been modeled at POP to trigger honeypot generation in response to suspected attacks and route the attack flows to honeypots. The performance of the proposed scheme is independent of attack due to presence of dynamic honeypots. It gives stable network functionality even in the presence of high attack load. In case of abrupt changes, it has a tendency to adapt itself according to network.

Some of the avenues for further experimentation are with larger and heterogeneous networks. Back tracking can be applied on attack flows to reach the attack source. Both of them hold promise for evaluating and improving our DDoS defense method and server information protection.

7. References

[1] CERT Coordination Center. *Denial of Service Attacks*. http://www.cert.org/tech_tips/denial_of_service.html.

[2] J. Mirkovic, and P. Reiher, "A Taxonomy of DDoS Attack and DDoS defense Mechanisms". *ACM SIGCOMM Computer Communications Review,* Volume 34, Number 2, April 2004.

[3] C. Sangpachatanaruk, S. Khattab, T. Znati ,R. Melhem, and D. Mosse', "A Simulation Study of the Proactive Server Roaming for Mitigating Denial of Service Attacks", *In Proc. 36th Annual Simulation Symposium 2003 (ANSS'03),* 7-14.

[4] Kuwatly, M. Sraj, Z. A. Masri, and H. Artail "A Dynamic Honeypot Design for Intrusion Detection", *In Proceedings 24th ICDCS'04, 2004 IEEE.*

[5] T. M. Gil, and M. Poletto, "Multops: a data-structure for bandwidth attack detection" *In Proceedings of 10th USENIX Security Symposium, 2001.*

[6] R. B. Blazek, H. Kim, B. Rozovskii, and A. Tartakovsky, " A novel approach to detection of denial-of-service attacks via adaptive sequential and batch sequential change-point detection methods", *In Proceedings of IEEE Systems, Man and Cybernetics Information Assurance Workshop, 2001.*

[7] B. Boldizsarand I. Vajda, "Protection against DDoS Attacks Based on Traffic Level Measurements", *Presented at Western Simulation MultiConference. California, USA., 2004.*

[8] Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions", *ACM SIGCOMM, 2005.*

[9] A. Sardana , K. Kumar and R. C. Joshi, "Detection and Honeypot Based Redirection to Counter DDoS Attacks in ISP Domain" In *Proceedings of* IAS 2007. Manchester, UK, pp. 191-196, *Aug* 2007.

[10] V. Paxson, "*Bro: A System for Detecting Network Intruders in Real-Time*", Computer Networks, vol. 31, nos. 23-24, 1999.

[11] M. Roesch, "Snort—Lightweight Intrusion Detection for Networks", *In Proceedings of USENIX Systems Administration Conf. 1999.*

[12] Y. Xu and R. Guerin R, "On the Robustness of Router-based Denial-of-Service Defense Systems", *ACM SIGCOMM, 2005.*

[13] J. Ioannidis and S. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks", *In Proceedings of IEEE INFOCOMM, 2003.*