

An Integrated Honeypot Framework for Proactive Detection, Characterization and Redirection of DDoS Attacks at ISP level

Anjali Sardana and R. C. Joshi¹

¹Indian Institute of Technology Roorkee,
Roorkee, Uttarakhand, 247 667, India
anjlsdec@iitr.ernet.in, rcjosfec@iitr.ernet.in

Abstract: Distributed Denial of Service (DDoS) attacks can effect the steady functioning of any network, posing a severe security threat. Concentrated single source DDoS attacks consume huge resources like bandwidth in a very small duration and have direct impact at ISP level, thus making them easily detectable. In contrast, diluted low rate DDoS attacks lead to graceful degradation of network over a longer duration and hence are mostly undetectable. The outcome of above attacks is that legitimate users are denied service. Though an array of schemes have been proposed for the detection of the presence of DDoS attacks, characterizing of the flows as a normal flow or a malicious one, and mitigating the effects of the attacks once they have been detected, there is still a dearth of complete frameworks that encompass multiple stages in the process of defense against DDoS attacks.

In this paper, we propose a novel honeypot framework that proactively detects the presence of attack, characterizes the TCP flows as attack or legitimate, and mitigates the influence of the attack by redirecting attack flows to honeypots. The detection has been achieved by an innovative entropy based scheme. Our detection mechanism adapts itself according to variation in attack loads in real time and calibrates the system to operate in one of the naïve, normal or best defense modes. The in stream flows are characterized on the basis of entropy value and mode of operation in moving time window. During mitigation, attack flows are redirected to autonomic dynamic honeypots present in the same network as active FTP servers. A dynamic honeypot engine (DHE) in honeypot controller (HC) module has been modeled to generate judicious mixture of honeypot and active FTP servers from a pool of servers depending on real time network conditions at ISP level. Goodput, mean time between failure and average response time have been evaluated for naïve, normal and best defense mode of operation.

We validate the effectiveness of the approach with simulations carried out at different attack strengths in ns-2 on a Linux platform. We also report our experimental results for detection over KDD 99 dataset. Results show that our proposed framework gives a drastic improvement in terms of average response time and good put. It meets the challenges of most of the existing solutions to DDoS with its ability to proactively detect variable rate attack in real time with minimum false alarms and minimum collateral damage. The proposed scheme has the potential to maintain stable network functionality even in the presence of attacks. It can be fine tuned according to the dynamically changing network conditions at ISP level.

Keywords: Distributed Denial of Service (DDoS), Dynamic Honeypot Framework, Detection, Characterization, Mitigation, ISP Domain.

1. Introduction

The Internet (originally known as ARPANET) was created to provide an open network for researchers [1]. Unfortunately, with the growth of the Internet, the attacks to the Internet have also increased incredibly fast. The widespread need and ability to connect machines across has caused the network to be more vulnerable to intrusions and has facilitated break-ins of a variety of types. According to [1], a mere 171 vulnerabilities were reported in 1995 which boomed to 8064 in the year of 2006. Apart from these, a large number of vulnerabilities go unreported each year. A Denial of Service (DoS) attack is commonly characterized as an event in which a legitimate user or organization is deprived of certain services, like e-mail or network connectivity, that they would normally expect to have. DoS attacks [2,3] inject maliciously-designed packets into the network to deplete some or all of these resources. The attack power of a Distributed DoS (DDoS) attack [4] is based on the massive number of attack sources instead of the vulnerabilities of one particular protocol. DDoS attacks, which aim at overwhelming a target server with an immense volume of useless traffic from distributed and coordinated attack sources, are a major threat to the stability of the Internet.

In many cases when a sustained high bandwidth attack reaches servers it is not possible to contain the attack at border gateway as the offending packets have already consumed the finite bandwidth available on the connection to the ISP[5][6]. In this case, having a good relationship and clear communication channels with ISP are essential. High bandwidth attacks have a direct impact on the ISP's network. Diluted low rate attacks are critical component and remain undetected until the network functionality becomes unstable thus targeting QoS. However, since ISP network are closer to the source of the attack they are in a better position to filter the offending traffic.

The number and assortment of both the attacks as well as the defense mechanisms is monstrous. Though an array of schemes has been proposed for the detection of the presence of these attacks, characterizing of the flows as a normal flow or a malicious one, identifying the sources of the attacks and mitigating the effects of the attacks once they have been

detected, there is still a dearth of complete frameworks that encompass multiple stages of the process of defense against DDoS attacks.

In this paper, we propose an integrated honeypot framework for defense against variable rate DDoS attacks at ISP level. By “integrated”, we mean, the framework will provide for the following activities in defense against DDoS attacks:

1. Real time *detection* of variable rate DDoS attack at ISP level.
2. Accurate *characterization* of the flows as attack or legitimate flows at POP,
3. Efficient *mitigation* of the effect of attack using autonomic dynamic honeypot redirection.

Honeypots [7], [8] a proactive detection mechanism, are machines that are not supposed to receive any legitimate traffic and, thus, any traffic destined to a honeypot is most probably an ongoing attack and can be analyzed to reveal vulnerabilities targeted by attackers. The entropic detection mechanism coupled with presence of honeypots makes the detection strong and proactive. Autonomic dynamic honeypots have been used in the framework to distract attackers from real targets within the network, and to mitigate the effect of ongoing attacks on active servers and collect data for research into attacker tools, methods, and motivations and for forensic analysis.

The rest of the paper is organized as follows. Section 2 gives a brief overview of some of the existing techniques to facilitate detection, characterization and mitigation of DDoS attacks along with some of their limitations. Section 3 gives a gist of our proposed integrated framework. Our detection scheme is charted in Section 4, whereas Sections 5 and 6 explain in detail our characterization scheme and its advantages. Section 7 describes the autonomic dynamic honeypot redirection. Section 8 gives the experimental design which is inclusive of the simulation testbed and the obtained results. Section 9 presents the advantages of our framework. We conclude our work and provide pointers to possible future work in Section 10.

2. Related Work

This section charts out the different work done in the areas of detection, characterization and mitigation of the effect of DDoS attacks and tracing back the sources of the attack.

2.1 Detection and Characterization

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. In practice, several signature-based detection systems have been developed and deployed at firewalls or proxy servers, such as Bro [9] and Snort [10]. By contrast, an anomaly based detection system observes the normal network behavior and watches for any divergence from the normal profile. Most of DoS detection systems are anomaly based [11]-[15]. However, their normal traffic models are mainly based on flow rates. Due to the

diversity of user behaviors and the emergence of new network applications, it is difficult to obtain a general and robust model for describing the normal traffic behaviors. 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. This becomes a bottleneck, especially for volume-oriented DDoS attacks that will be aggravated by the computational overhead of the detection scheme. In [16], and [17] 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. In [18] random projection technique has been used to reduce the dimensions followed by SVM algorithm that define thresholds detect intrusions.

Though schemes in [11]-[15], use volume based metrics to detect and characterize DDoS attacks and have been successful in isolating large traffic changes (such as bandwidth flooding attacks), but slow rate, isotropic attacks can not be detected and characterized because these attacks do not cause detectable disruptions in traffic volume. These suffer in the form large number of false positives/negatives hence more collateral damage when attack is carried at slow rate or when volume per attack flow is not so high as compared to legitimate flows.

2.2 Mitigation

In order to minimize the loss caused by DoS attacks, a reaction scheme must be employed when the attack is underway. Most solutions to the DDoS attacks try to develop a packet filter that can distinguish legitimate packets from illegitimate ones and hence drop illegitimate packets only [19]. The D-WARD defense system [6] is deployed at source-end networks, and autonomously detects and stops attacks originating from these networks. Wide deployment of D-WARD will motivate service-level attacks, on which we focused in this paper. In [20], Bohacek has suggested a mitigating approach that relies on routers filtering enough packets so that the server is not overwhelmed while ensuring that as little filtering as possible is performed. He has proposed a solution wherein packets should be filtered at routers through which the attack packets are passing. But, it is a reactive mitigation technique that also has the drawback that legitimate traffic packets may also be dropped en route to the destination.

In the Pushback framework [21], once a router suffers from sustained congestion, it tries to detect flow aggregates that are contributing the most to congestion. The congested router rate-limits the detected flow aggregate(s) and sends the aggregate signature (e.g., destination address) to up stream routers, which apply rate-limiting to the aggregate and recursively push the rate-limiting upstream toward attack sources. Pushback requires contiguous deployment; to overcome this limitation, Selective Pushback [22] proposes to send rate-limiting requests to routers sending traffic with

higher than “normal” rates. The detection of these routers and the profiling of normal traffic are performed via an enhanced probabilistic packet marking scheme.

In [23], Kalantari et al. have proposed a proactive method for mitigation of the effects of DDoS attacks wherein each router maintains a partition of active TCP flows into aggregates. Each aggregate is probed to estimate the proportion of attack traffic that it contains. Packets belonging to aggregates that contain significant amounts of attack traffic may be subject to aggressive drop policies to prevent attack at the intended victim. Again, in this case too, legitimate packets face the risk of being dropped. Also, proper definition of aggregates is a critical part of the approach. Moreover, aggregates have to be defined in advance of the attack so that their response measurements are taken to normal (non-attack) traffic in order to be compared later on with measurements under an attack, if any.

IP hopping [24] protects a public server, whose clients use DNS to look up its IP address. In IP hopping, the server changes its IP address without changing its physical location. All packets destined to the old IP address are filtered at the network perimeter by a firewall. However, as physical location is not changed, physical connections are not broken and states of attacked systems remain unclear. TCP-Migrate [25] and Migratory-TCP [26], which provide a framework for moving one end point of a live TCP connection from one location and reincarnating it at another location having a different IP address and/or a different port number, are used for mobility support and fault, or attack, tolerance. Mutable Services [27] is a framework to allow for reactively relocating service front-ends and informing only pre-registered clients of the new location through a secure DNS-like service. Sherif et. al. [28], proposed the proactive server roaming mechanism, which is a secure and light-weight mechanism to proactively change the location of the active server within a server pool. Legitimate clients keep track of roaming times and location of the roaming server using light-weight, one-way hash functions. However, the scheme incurred an overhead that caused performance degradation both in the absence of attacks and under low attack loads.

It is observed that most of the mitigation techniques in practice today, suffer from the following drawbacks:

1. They are reactive in nature.
2. They deploy packet dropping policies at the routers wherein even legitimate packets face the risk of being dropped.
3. In cases like [23], the topology of the network needs to be known in advance.

The mitigation technique used in the framework proposed in this paper does away with all these drawbacks as we shall see in Section 9.

3. Proposed Solution

In this section, we shall discuss in great depth the various facets of our proposed framework for defense against DDoS

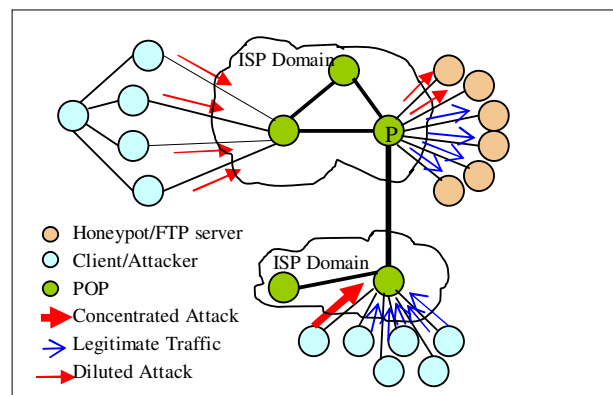
attacks.

The proposed framework provides for proactive mitigation against the effects of DDoS attack as described next. All the flows arriving at the Point of Presence (POP) of an ISP and destined to the server to be protected from DDoS attack are tagged as either legitimate or attack. Whenever a packet belonging to suspicious flow arrives at the POP, instead of sending that packet to the active FTP server or dropping it, it is redirected to honeypot server. This provides a *proactive* approach to mitigation against the attack because the FTP server is isolated from attack traffic and bandwidth of the links with FTP server will not be exhausted by the voluminous attack traffic.

A flow list is generated at the POP by sampling the in stream traffic in moving time window. It represents actual dynamics of the packets of the flow to which they belong. These will be subject to detection followed by categorization test described next.

During detection, we demonstrate the utility of a more sophisticated treatment of DDoS anomalies, as events that alter the distribution of traffic features. For concentrated as well as diluted attacks, traffic distributions have appreciable deviation from normal to provide signs of DDoS attack. The characterization of attack traffic is done by monitoring the flow list at the POP in moving time window and then choosing suspicious flows. For detection and characterization, entropy [29] (discussed in detail in section 4) will be used as the part of this framework. The entropy technique does away with the disadvantages of the previous methods based on aggregates[16], [17] and volume based metrics [11]-[14] as it is adaptable to varying network conditions and hence provides accurate proactive detection of variable attacks. Moreover, no hardware implementation as in [30] is required.

The proposed framework provides proper response mechanism in place. When the attackers get through, the network reacts and responds to the situation intelligently.



To get a better understanding of the proposed model,

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 goes under entropic test 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 redirected to one of the randomly selected honeypots.

Connections with the illegitimate flows are retained by honeypots. Honeypots respond in contained manner to the attacker's request before dropping them. They log information about the attack flows either for a fixed time interval equal to the time window for characterization (or a multiple thereof) or till the expiry of current eon (variable time duration, discussed later), whichever less. If the flow to a honeypot is retagged as legitimate in subsequent time windows, it is redirected to active FTP server hence minimizing the collateral damage and the effect of false positives.

In the dynamic network environment with variations in client and attack load and presence of sophisticated attacks, detection and mitigation process fine tune themselves in real time. The parameters like tolerance factor of the network, number and locations of servers and honeypots are varied to minimize number of false alarms and provide stable network functionality. Characterization and mitigation work on two different time scales. The details of the detection and autonomic dynamic honeypots are given in the sections to follow.

4. Statistical Entropy Based Detection

4.1 Entropy in brief

Our hypothesis to detect and characterize attacks treats DDoS anomalies as events that disturb the distribution of traffic features. For example, a DoS attack, regardless of its volume, will cause the distribution of destination address to be concentrated on the victim address. As proposed by Sardana et al [29], we use entropy to capture the degree of dispersal or concentration of a distribution. The sample entropy $H(X)$ is

$$H(X) = -\sum_{i=1}^N (p_i) \times \log_2(p_i) \quad (1)$$

where $p_i = n_i/S$. The value of sample entropy lies in the range $0 - \log_2 N$. The metric takes on the value 0 when the distribution is maximally concentrated, *i.e.*, all observations are the same. Sample entropy takes on the value $\log_2 N$ when the distribution is maximally dispersed, *i.e.* $n_1 = n_2 = \dots = n_n$.

4.2 Detection of Attack

Consider a random process $\{X(t), t = j\Delta, j \in N\}$, where Δ a constant time interval is called time window, N is the set of

positive integers, and for each $t, X(t)$ is a random variable. Here $X(t)$ represents the number of packet arrivals for a flow in $\{t - \Delta, t\}$. $X(t)$ It is found in our simulation without attack that Entropy $H(X)$ value varies within very narrow limits after slow start phase is over. This variation becomes narrower if we increase Δ *i.e.* monitoring period. We take average of $H(X)$ and designate that as normal Entropy $H_n(X)$. To detect the attack, the entropy $H_c(X)$ is calculated in shorter time window Δ continuously, whenever there is appreciable deviation from $H_n(X)$, attack is said to be detected. We assume that the system is under attack at time t_a , which means that all attacking sources start emitting packets from this time: the network is in normal state for time $t < t_a$ and turns into attacked state in time t_a . Let t_d denote our estimate on t_a . At time t_d following event triggers

$$\begin{aligned} (H_c(X) > (H_n(X) + a \times d)) \cup \\ (H_c(X) < (H_n(X) - a \times d)) \end{aligned} \quad (2)$$

$attack = true;$

Here $a \in I$ where I is set of integers and d is deviation threshold. Tolerance factor a is a design parameter and d is absolute maximum deviation in Entropy $H(X)$ from average value $H_n(X)$ while profiling for network without attack. In [29] it was observed that 'a' is a function of client and attack load ('a' is regulated by detection rate and false positive rate). However, in dynamic network conditions, the design parameter also depends on system and client requirements. In later sections we will show that system can be made auto responsive to attacks if average response time, mean time between failure and goodput also regulate the value of 'a'.

5. Attack Flow Characterization at Point of Presence

Once the attack is launched at POP P_s , we have aggregate of attack flows and normal flows. Let F represent set of active flows. Then

$$F = F_n \cup F_a (F_n \cap F_a = \phi) \quad (3)$$

Where F_n represent actual normal flows and F_a is set of actual attack flows. Our main task in this module is to find

$$F_a^* = \{f_1, f_2, \dots, f_m\} \subset F \quad (4)$$

the set of m malicious flows. Ideally,

$$(F_a^* \cap F_a = F_a) \text{ AND } (F_a^* \cap F_n = \phi) \quad (5)$$

Now the main problem is to find m :-

- as for diluted low rate attacks, m number of least measured packet arrival flows constitute F_a^* .
- and for concentrated high rate attacks, m number of highest measured packet arrival flows form F_a^* .

To answer these questions if we can find Φ_a , the expected total attack traffic then from following equation, we can find m and F_a^* .

$$\sum_{j=1}^m X_j^i(t_d + \Delta) \leq \Phi_a \quad (6)$$

Where i is designated flow, j varying from 1 to m for least or highest measured packet arrivals, and $X(t_d + \Delta)$ represent packet arrivals for flow i in next time window after attack is detected. The expected value Φ_a , is calculated as $\Phi_a = \Phi_{id} - \Phi_n$, where Φ_{id} is the total traffic received in $\{t_d - \Delta, t_d\}$ and Φ_n is averaged total traffic till $t_d - \Delta$ from the time bottleneck link utilization is 1.

The set of active flows F also referred to as Flow List (FL) in section 7 is updated at the beginning of each time window by tagging each flow as attack or legitimate

6. False Positives and False Negatives

False positives give the effectiveness of the system whereas false negatives give a measure of the system reliability. Variations in tolerance factor ‘a’ has been used to quantify false positives and false negatives [29] which assist in making decisions on optimal value of entropic thresholds. Our proposed detection scheme works in one of the three modes, namely best, normal and naïve defense. The mode of operation is chosen according to attack strength by varying ‘a’ so as to minimize false positives and false negatives. In case of best defense shown in figure 2, the normal entropy bandwidth during attack detection is chosen to be very small. The choice is made to reduce the false negatives to minimum, and is zero in ideal case. Figure also shows the operation of normal and naïve defense mode. Naïve defense has the lowest detection sensitivity level and hence it has lowest false positive rate. Once ‘a’ is decided, optimum entropic thresholds are automatically updated to calibrate the system. Decision of optimum value of ‘a’ under varying network conditions becomes a function of related output parameters as discussed in section 8.2.

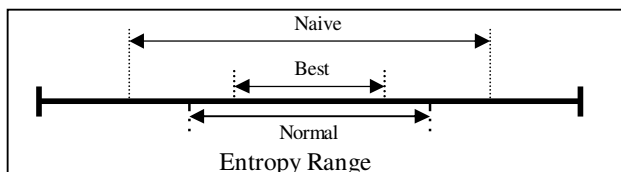


Figure 2. Best, Normal and Naïve Defense

7. The Redirection Scheme

7.1 Autonomous Dynamic Honeyspots

Our approach for dynamic honeypot generation is in response to flows identified as legitimate and attacks in flow list (FL). Given a server pool, we propose automatic generation of appropriate number of active servers to service the legitimate requests and adequate number of honeyspots to isolate the attacks at the same time. A honeypot controller (HC) has been modeled at the point of presence where dynamic honeypot engine (DHE) takes as input the flow list (FL) maintained during detection and characterization process and generates a judicious mixture of active servers and honeyspots from the server pool.

Honeypot Controller (HC) module performs three functions. First, it keeps a track of FL and state of flow. Second, depending on the attack load and client load, DHE

triggers the generation of appropriate number of servers and honeyspots and coordinates their timings and location. Third, 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 at a remote database located inside DMZ for later analysis. A higher-level responsibility of the honeypot controller is to manage and prioritize the selection of server set.

7.2 Service Model

Our network consists of a pool of N homogenous servers in a server pool with P active FTP servers and $(N-P)$ honeypot servers. Each destination in the network should be able to behave according to whether the corresponding node acting as an active server or honeypot. Thus the services have been replicated on all servers. We use FTP which is a TCP based service that can be replicated. We call the duration for which each node acts as a honeypot or active server as an eon. An FTP connection remains alive until either the legitimate FTP request is fulfilled completely or the tag of the flow in the FL changes at the start of next time window or if the number and location of servers and honeyspots change (eon expires) before the request is fulfilled. An illegitimate FTP request is directed to honeypot which retains the connection and responds to attacker in contained manner for a fixed time interval equal to the time window of characterization or till the current eon terminates, whichever is less. If the flow is retagged as attack before termination of current eon, then the connection with honeypot is retained. In case of false positive, the connection is migrated back to active FTP server in subsequent characterization time window.

7.3 Implementation Details

Autonomous Dynamic Honeypot based redirection involves following steps.

7.3.1 Deciding the number of honeyspots and servers

Let N_S and N_H represent the number of servers and honeyspots 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 honeyspots respectively such that following holds true:

$$(lengvec = (Length(subvecNS) + Length(subvecNH))) \text{ AND} \\ (subvecNS \cap subvecNH) = \phi \quad (7)$$

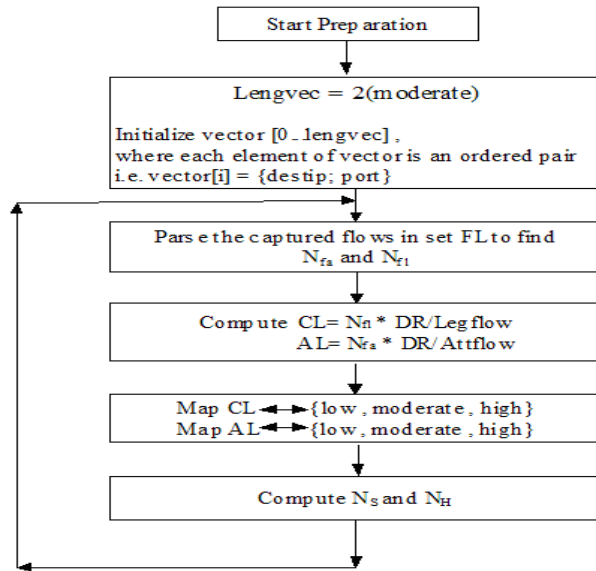


Figure 3. Steps at Honeypot controller engine module to compute N_S and N_H

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 steps at DHE at honeypot controller module to compute N_S and N_H .

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

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

7.3.2 Deciding the location of servers and honeypots

Having identified the number of servers and honeypots, next we utilize backward hash chain [28] to calculate the locations and achieve roaming which is inherent in autonomous dynamic honeypots. The scheme builds on connection migration mechanisms and provides a secure framework for issuing the roaming trigger proactively.

N_S active servers continuously change their location within a pool of $lengvec$ homogenous servers to proactively defend against DDoS attacks. Service time is divided into eons; at the end of each eon, CL and AL is mapped to N_S and N_H using table 1 and the service migrates from one set of servers to other in the server pool. A long hash chain is generated using a one way hash function $H(\cdot)$, and used in

backward fashion similar to Pay-Word scheme [31]. The last key in the chain, K_n , is randomly generated and each key, K_i ($0 < i < n$), in the chain is computed as $H(K_{i+1})$ and is used to calculate both the length, R_i , of service eon E_i , and set of current active servers F_i during E_i . Let S represent the set of indexes of the $vector[]$ array. Also let $P_{NS}(S)$ represent an ordered set of all possible subsets of S with cardinality N_S . The cardinality of $P_{NS}(S)$ is $N_P = (N_S / (lengvec! * (N_S - lengvec)!))$, where $lengvec$ is the number of servers in $vector[]$. Then for each service eon E_i , the set of current active servers is $P_{NS}(S)[MSB_{lg NP} H'(K_i)]$, where $H'(\cdot)$ is a one way hash function and $MSB_x(y)$ are the x most significant bits of y . The length R_i , of the service epoch E_i is uniformly distributed in the interval $[u, m+u]$ seconds as follows: $R_i = u + MSB_m(H'(K_i))$, where m represents the current threat level. Hence, the value of the system parameter m is changed adaptively depending upon N_S and N_H . Value of m is inversely proportional to the threat level, i.e. it is directly proportional to N_S and inversely proportional to N_H and is derived from table 2.

The value u represents a lower bound on the idle time of a server and should be long enough to analyze attacks. HCE keeps account of N_S , $lengvec$, u , $vector[]$, m and $P_{NS}(S)$ as roaming updates.

Table 2. Mapping N_S and N_H to m

		N_S				
		Low	Mod-Low	Mod	Mod+Low	High
N_H	Low	5	5	5	5	5
	Mod - Low	4	4	3	3	4
	Mod	3	4	3	4	3
	Mod + Low	2	1	1	2	2
	High	1	1	1	1	1

7.3.3 The redirection algorithm

The redirection algorithm performs the per-flow treatment of each flow in the Flow List (FL) in a time window at POP. The pseudo code is as follows:

HoneypotControllerPerFlow (FL)

```

For a flow in FL
If (Tag = attack)
    Parse the primary packet and search source and destination address (FDA and FSA)
    PDA = FDA
    NDA = PDA
    A: If (NDA = Destination address of honeypot)
        Forward the packet to NDA
    Else
        Replace NDA by destination address of honeypot
        Forward the packet to NDA
If (More Fragment = 0)
    Goto S
Else
    Parse next header of the flow for PDA
    NDA = PDA
    If (Tag = attack)
        Goto A
    Else

```

```

Goto B
Else
  Parse the primary packet and search source and destination address (FDA
  and FSA)
  PDA = FDA
  NDA = PDA
  B: If (NDA = Destination address of active FTP server)
    Forward the packet to NDA
  Else
    Replace NDA by destination address of server
    Forward the packet to NDA
  If (More Fragment = 0)
    Goto S
  Else
    parse next header of the flow for PDA
    NDA = PDA
    If (Tag = attack)
      Goto A
    Else
      Goto B
S: Stop

```

7.4 Mathematical model

At any active ftp server node i , the arriving traffic is the aggregate of several legitimate and possibly of some attack flows, where $l = (l_1, l_2, \dots, l_j, \dots, l(N_{fp}))$ and $a = (a_1, a_2, \dots, a_i, a(N_{fa}))$. Our set FL contains a and l . The total traffic rate λ_i arriving to node i is composed of two parts:

$$\lambda_i = \sum_n \lambda_i^n, n + \sum_d \lambda_i^d, d \quad (2)$$

where λ_i^n, n is the legitimate incoming traffic rate which belongs to normal flow n , and λ_i^d, d is the arrival rate of attack packets belonging to flow d .

Some attack traffic may be mistakenly taken to be normal traffic while some normal traffic will be mistakenly thought to be attack traffic. Any traffic characterized as attack is redirected at one of the randomly selected honeypot servers. Thus, a fraction f_i, n of normal traffic (the probability of false alarms) and a major fraction of attack traffic d_i, d (the probability of correct detection) will be redirected to honeypot as it arrives to the honeypot controller. If the node's attack detection mechanism were perfect we would have $f_i, n = 0$ and $d_i, d = 1$. However, characterization mechanism is such that probability of false negative is 0 i.e. practically $d_i, d = 1$. However there may be some false positives, so $1 > f_i, n > 0$. Once a packet is admitted into a honeypot controller, it is queued and then forwarded based on its selected destination address. We model each node by a single server queue with service time s_i representing both the time it takes to process the packet in the node and the actual transmission time. The traffic intensity parameter at active FTP servers is ρ_i :

$$\rho_i = s_i \left(\sum_n I_i^n, n(1 - f_i, n) + \sum_d I_i^d, d(1 - d_i, d) \right), \quad (3)$$

where for node i , I_i^n, n is the arriving traffic rate of normal flow n , and I_i^d, d is the arriving traffic rate of a DDoS flow d .

To evaluate the effectiveness of our scheme, we measure the goodput at each node. This establishes the effectiveness of our DDoS protection scheme, and also of how successful or unsuccessful the DDoS attack has been. The goodput $G(i)$ at each node is:

$$G(i) = \sum_n I_i^n, n(1 - L_i)(1 - f_i, n) \quad (4)$$

where L_i is the buffer overflow probability or probability of packets lost at server node.

During redirection, ideal situation at honeypot is described by equalities, $f_i, n = 0$ and $d_i, d = 1$

Table 3 gives the practical situation at node i .

Table 3. Mapping load to honeypots and servers

i	Best Defense	Naïve Defense
Active server	$((1 - d_i, d) = 0); (1 - f_i, n)$	$((1 - f_i, n) = 1); (1 - d_i, d)$
Honeypot	$d_i, d = 1; f_i, n > 0$	$d_i, d < 1; f_i, n = 0$

8. Experimental Design

8.1 Simulation Topology

The end systems or hosts (users PCs, PDAs, web Servers, and mail servers etc) in Internet connect to each other through a tiered hierarchy of Internet Service Providers (ISPs). Within an ISP's network, the points at which the ISP connect to other ISPs (whether below, above, or at the same level in the hierarchy) are known as points of presence (POPs). The interconnection of POPs of an ISP through high bandwidth links is called ISP backbone. In an ISP's network a POP is actually a group of connected core and access routers to which core routers of the same or other ISPs (private/public peer or NAT) and ISP's own customers and servers are connected respectively. Whenever two ISP are directly connected to each other, they are said to be peer with each other. Though, complexity of POP's connecting router will vary depending upon whether other ISP router (normally core) or own customer domain (normally access) is attached.

For simulation purpose, we use ISP level network with four cooperative ISP domains (1, 2, 3, and 4) where each domain has 10 POPs represented by single node each as shown in figure 4. One customer domain is attached to each POP which consists of legitimate and attacking hosts. Two POPs in every ISP are attached to other ISPs. ISP domain 4 has additional POP for connecting to our protected server. Our aim in this paper is to detect DDoS attack in ISP domain 4.

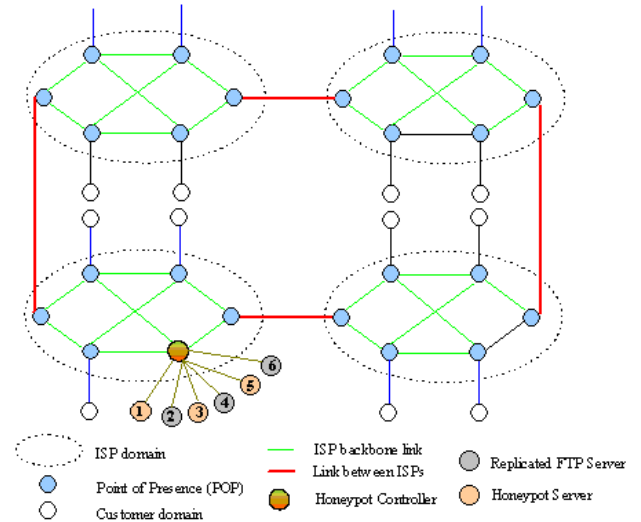


Figure 4. Simulation Topology

We have represented transit domain routers as POPs of the ISP and stub domains as customer domains attached to

POPs as shown in figure 4. Honeypot servers and the protected FTP servers vary in number after each eon. The POP link to server bandwidth for honeypot server is kept same as FTP server so that it exactly imitates the FTP server. Following table 4 gives topology generator parameters.

Table 4. Topology Generator Parameters

S.No.	Parameter	Value
1.	ISP domains	4
2.	No. of transit routers	12 (1 more in ISP 4 for connecting servers)
3.	Edge probability	.85
4.	Number of stub domains	10 / ISP
5.	Number of hosts	10 / stub domain
6.	Backbone link bandwidths	2.5GhZ
7.	Backbone link delays	0 seconds

There are four ISP domains with two peers each i.e. two other ISP domains are directly attached at POPs.

Table 5 provides the basic parameters set for simulation.

Table 5. Basic parameters of simulation

S.No.	Parameter	Value
1.	Simulation Time	60 seconds
2..	Number of legal sources	100 / ISP domain Total 4*100=400
3.	No. of attackers	1-25 / ISP domain. Total= 1-100
4.	Access bandwidth for legitimate customers	1Mbps
5.	Bottleneck Bandwidth	310Mbps
6.	Mean attacker rate	0.1-1.0Mbps(low rate) 2.7-3.7Mbps(high rate)
7..	Attack duration	20-50 seconds

We varied attack rates as given in S.No.6 and computed entropy and goodput. All the legitimate TCP connections are not initiated at the same time as SYN backlog is also limited in size as shown in S.No.3 below.

Table 6. Traffic Parameters

S.No	Parameter	Value
1.	Traffic arrival process at legitimate clients	Poisson
2.	Traffic generator at attackers (mean attack rate given in Table 2, S.No.6)	Attack tools available at www.nlanr.org
3.	Connection startup time	Random 1-8seconds
4.	Packet Size	1040bytes

Table 7. Attack detection parameters

S.No.	Parameter	Value
1.	Window Size	.2 seconds
2.	Tolerance factor a for entropy deviation	3-10

Simulations are carried at different values of tolerance factor a for different attack strengths.

8.2 Results and Discussion

8.2.1 Detection and Characterization

(a) Threshold setting

We conducted simulation experiments for finding out threshold for entropy under normal condition as per simulation parameters given in previous section. The

normal range of entropy is calculated by using frequency distribution of number of packets per flow ID (SourceIP, SourcePort, DestinationIP, and DestinationPort) in time windows of 0.2 seconds. Simulation is also carried by taking longer window of 1.0 second. Deviations are still lesser as expected however average is almost same.

It is found that entropy value also lies in small range of 8.382442 to 8.441088, whereas this varies depending upon network environment and type of application .The average is 8.407158, standard deviation is 0.012, and maximum absolute deviation from average is .03393 .Finalized simulation parameters are:-

Normal Entropy Value ($H_n(X)$):- 8.407158

Maximum absolute deviation from average (d):- 0.03393

(b) Detection of attack

When our network is put under diluted low rate attack, in first time window after attack is launched at 20 seconds, there is jump in entropy value. The positive jump and persistent high value as compared to normal reflects that it is a diluted low rate attack and the flows which are causing this anomaly have comparatively lesser frequency than already existing ones.

In case of concentrated high rate attacks, entropy value tends to be lower than normal. In our simulation using total attack strength of 300Mbps with 100 attackers, the entropy drops down to 7.1.

However initially it can rise but with proper adjustment of window and start time, the same can also be lumped. In this case, the flows which have comparatively higher share of packets are reasons of anomaly. Similar trends exist for high rate attacks at different attack strengths with variation only in deviation from normal value.

Zhang et al[32] show the performance of detecting attacks at a fixed threshold of sensitivity. If the threshold is set too low, then the false alarm rate will be low, but detection rate will be low, too. Similarly, a threshold set too high may end up detecting most intrusions, but suffer from a high false alarm rate. Hence the threshold should match the dynamics of the network. Figure 5 shows the ROC curves for KDD 99 datasets using our proposed scheme with varying tolerance factor under 1%, 2%, 5% and 10% attacks. The area under ROC curve in the figure shows that performance increases upto 17% in case of 10 % attacks.

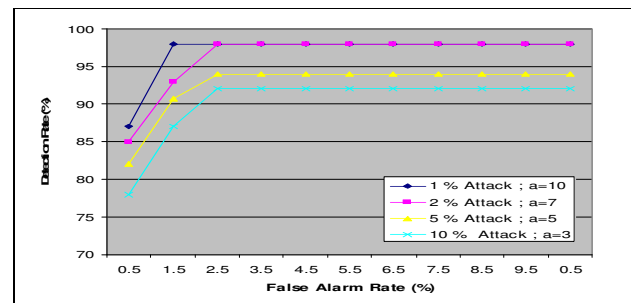


Figure 5. ROC Curve: Our proposed scheme

Hence, proper choice of tolerance factor a determines the optimum entropic thresholds during detection to calibrate the system. Value of a when reduced during high attack load gives better ROC performance, self calibrating the system.

From ns-2 simulations, we calculated the entropic thresholds by varying the values of a as tabulated below:

Table 8. Entropic thresholds with varying tolerance factor

Parameter a	Lower Bound	Upper Bound
3	8.305368	8.508948
4	8.271438	8.542878
5	8.237508	8.576808
6	8.203578	8.610738
7	8.169648	8.644668
8	8.135718	8.678598
9	8.101788	8.712528
10	8.067858	8.746458

We define three modes of operation, namely naïve, normal and best that take the values of a as shown in table 9.

Table 9. Mapping a to mode of operation

Mode of Operation	Value of a
Naïve Defense	8,9,10
Normal Defense	6,7
Best Defense	3,4,5

8.2.2 HoneyPot Based Redirection

Goodput, mean time between failure and average response time have been used as parameters to evaluate the three modes of operation under different attack strengths and network load. Goodput gives the measure of effectiveness of the system, mean time between failures gives system reliability and average response time gives a measure of the efficiency of proposed framework.

(a) Goodput

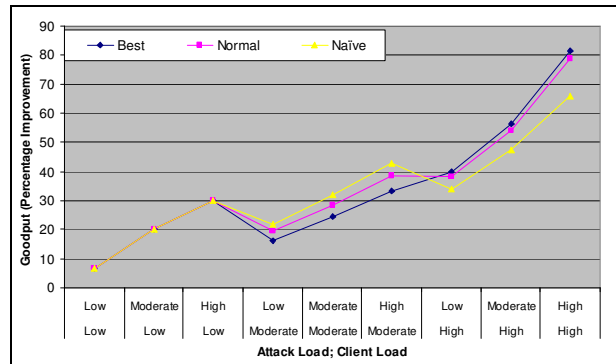


Figure 6. Percentage improvement in goodput with varying network load under naïve, normal and best defense

Figure 6 shows percentage improvement in goodput under three modes of operation. In case of low client load, best, normal and naïve defense schemes give equal improvement of 30% in goodput as compared no defense irrespective of the attack load. The three modes of operation give maximum goodput which is almost equal to ideal goodput and goodput with no DDoS. Value of N_s triggered by DHE is such that there are adequate numbers of active FTP servers to fulfill all legitimate requests at low client load.

In case of moderate client load, naïve defense gives better

performance as compared to best defense. An improvement in goodput up to 42% has been reported even in the presence of high attack load. This is so because value of N_s triggered is enough to fulfill legitimate requests. Because of a limited number of false positives in naïve defense, most of the legitimate flows reach the active servers directly than via honeypot as in case of best defense. Best defense causes more packet loss and collateral damage. For moderate client and particular value of attack, say moderate attack, best defense gives lower goodput than naïve defense. In case of naïve defense, the entropy range increases. Hence, rate of false positives reduce which implies that number of legitimate clients being detected as attackers reduce. The number of legitimate clients being sent directly to active FTP servers (instead of being detected as attacks in best defense and directed to honeypots) increase. Hence the goodput increases.

In case of high client load, up to 81% improvement in goodput is reported in case of best defense, as compared to no defense. Goodput in absence of attack is slightly lesser than ideal goodput due to loss of packets caused by buffer overflow in simultaneous presence of many legitimate requests flocking active FTP servers. For high client load and a particular value of attack load, best defense gives higher goodput improvement than naïve defense. This behavior is exactly opposite to that of moderate client load and can be explained as follows. As the client load is high, smaller entropy range of best defense sends a smaller selected set of legitimate client load to limited active FTP servers to be processed efficiently without loss. However, in case of naïve defense, higher entropy range allows more number of client requests to be sent to active FTP servers along with increased false negatives. It increases the processing load on limited available active FTP servers causing packets to drop and losses thus reducing goodput. With high client load and a particular scheme, say naïve, increase in attack load decreases the goodput slightly. This behavior is same as shown in case of moderate client load because of change in mapping according to changing attack load as in table 1.

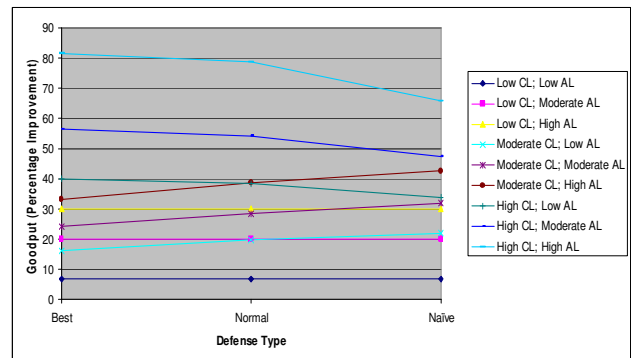


Figure 7. Defense type vs. percentage improvement in goodput

Figure 7 summarizes the results of the goodput. We conclude that tuning the system for best defense at high client load and naïve defense at low client load gives the best performance.

(b) Mean Time between Failures

Mean time between failures (MTBF) signifies the reliability. It is an attribute that quantifies attack tolerance of the network. Consider the network as a system whose input is the client request and output is the response to the request. Ideally, system is said to have failed if there is no response to the client request or when the average response time becomes infinite. However, a system must also guarantee a promised QoS where the response to a request should reach the client within a predefined interval or there is an upper limit on the average response time. For the purpose of simulation, we assume the system to have failed if the average response time becomes greater than the sum of duration of last five eons.

Figure 8 shows the variation in MTBF of a network with the increase in attack load for varying client loads in case no defense against DDoS is used. In the presence of high client load, even a slight increase in attack load causes the system to abrupt failure. The network is comparatively more stable with respect to moderate and low client load. However, even at low client load, the system fails in the presence of a very low attack load. Figure 9 shows that in case of high client load (~ 300 Mbps), best defense gives better stability with increase in attack load as compared to naïve defense. It is so because in best defense, there are no false negatives and all the attack flows are detected and directed to honeypots. Whereas in case of naïve defense, many attack flows will go undetected (with large number of false negatives) and will be directed to active server causing disruption in services and ultimately leading to the failure of the network with increase in attack load.

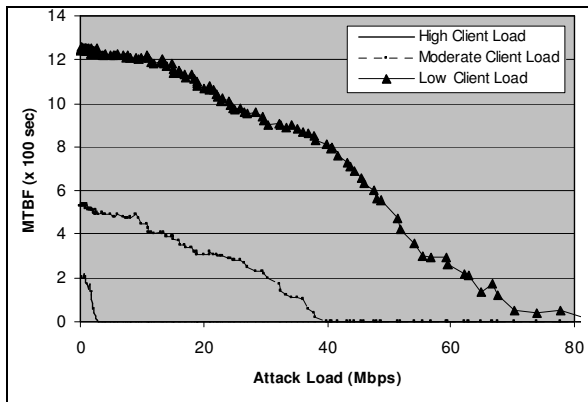


Figure 8. Variation of MTBF with Attack load

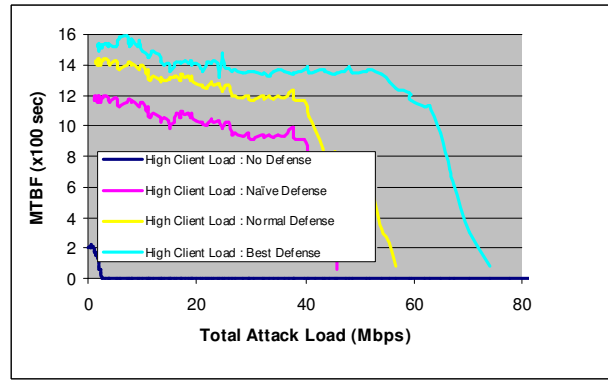


Figure 9. Variation of MTBF with AL; High CL

At moderate and low client loads (150 Mbps and 20 Mbps respectively), best defense has greater MTBF than naïve defense due to similar reasons as described. This is shown in figures 10 and 11 respectively.

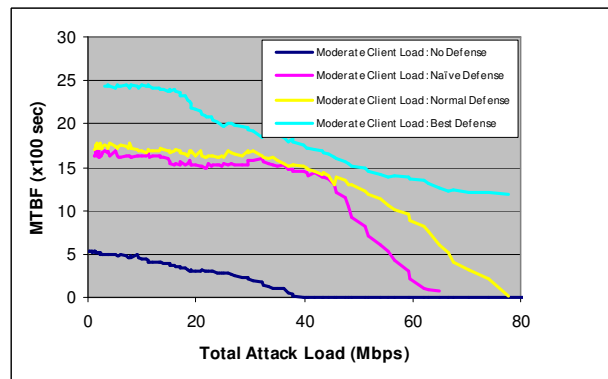


Figure 10. Variation of MTBF with AL; Moderate CL

It may be noted that no failure is reported in case of high and moderate client load (figure 9 and 10) in the presence of an attack load of upto 2 Mbps and 5 Mbps respectively in the presence of honeypots. However, at low client load, the network becomes much stable with no network failure reported even if there is an attack load of about 20 Mbps.

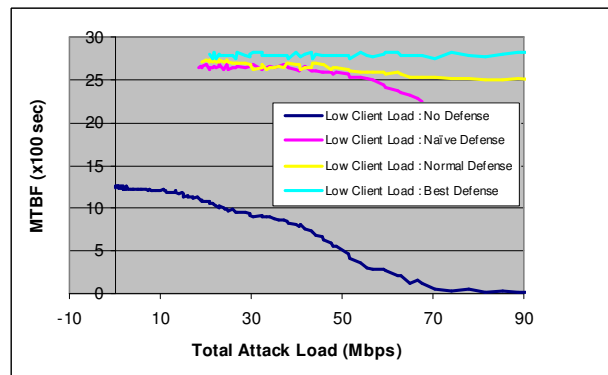


Figure 11. Variation of MTBF with AL; Low CL

Moreover MTBF remains consistent even if the attack load increases beyond 23 Mbps in case of low client load, normal and naïve defense. This demonstrates that the proposed framework has the potential to give stable network functionality even in the presence of attacks. Best defense

gives best performance for MTBF under any amount of network load.

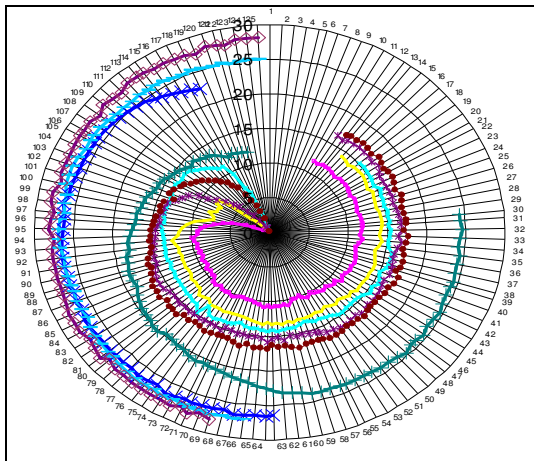


Figure 12(a). Variation of MTBF with AL; Naïve, Normal and Best defense ; The Radar Plot

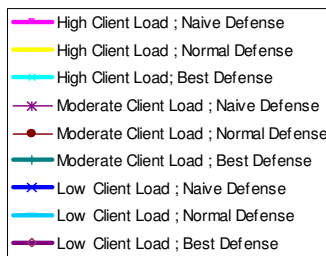


Figure 12(b). Key for the radar plot

(c) Average Response Time

Average Response Time (ART) signifies the efficiency of the system and gives a measure of functionality even in the presence of attack load. In case of low attack load, the naïve defense gives the best results with up to 78% improvement in ART as compared to no defense as shown in figure 13. Naïve defense has high false negatives and gives low false alarms. But as attack load is less, false negatives become insignificant. Due to low false alarm rate, most of the legitimate flows are directly sent to active FTP servers and

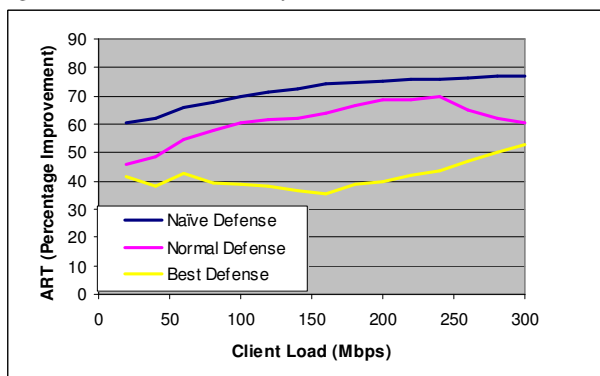


Figure 13. %age improvement in ART vs. CL (Low AL)

hence ART decreases. However, in case of best defense, ART is higher and hence percentage improvement is less as compared to naïve defense. This is so because best defense gives high false alarm rate and low false negatives. As the

client load increases, the numbers of false positives increase. These flows are first directed to the honeypots before being identified as legitimate flows and being redirected to active servers. Hence, this leads to an increase values of average response time with increasing client load in case of best defense.

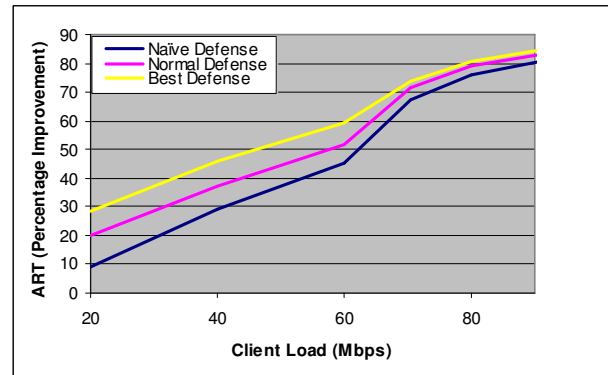


Figure 14. %age improvement in ART vs. CL (High AL)

The figure 14 shows percentage improvement in ART with increasing client load at attack load of 45 Mbps. With high attack load, best defense gives more improvement in ART values as compared to naïve defense. It is so because best defense gives high false alarms and low false negatives. As attack load is high, and false negatives are negligible, best defense is able to identify all attacks with zero false negatives and this isolates the server from high attack load, thus maintaining the stable ART. On the other hand, naïve defense gives low false alarms but high false negatives. So at higher attack loads, more attackers remain unidentified and flock the server, giving higher values of ART. After an increase in attack load beyond 45 Mbps, the MTBF abruptly decreases for high and moderate client loads. Hence the average response time curve shows abrupt and chaotic behavior.

The figure 15 shows that in case of low client load and low attack load, naïve defense gives the best performance with up to 60% improvement as compared to no defense. The improvement decreases with increase in attack load because at high attack load, naïve defense gives higher false negatives. Hence attack load may go to active servers causing

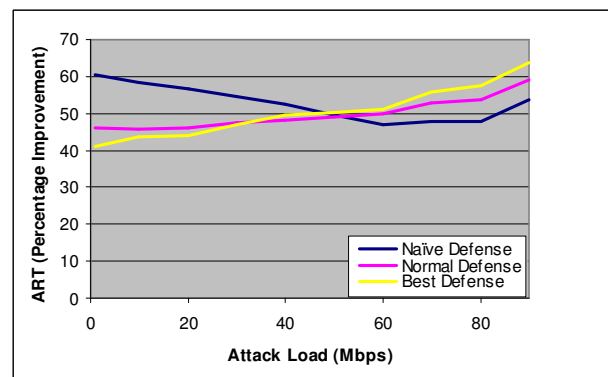


Figure 15. %age improvement in ART vs. AL (Low CL)

congestion and increased response times. Hence, at higher attack loads, best defense shows higher improvement in ART values.

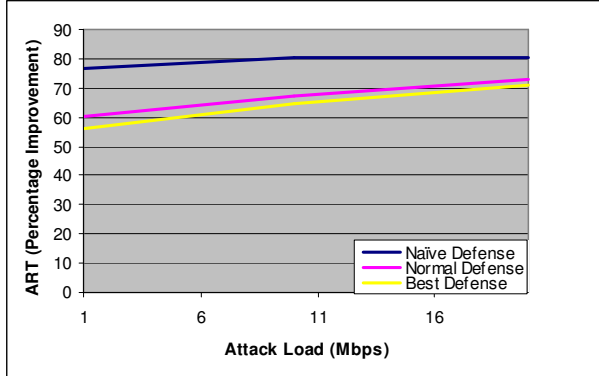


Figure 16. %age improvement in ART vs. AL (High CL)

Figure 16 shows percentage improvement in average response time with attack load in the presence of high client load (~300 Mbps). For an attack load up to 20 Mbps, naïve defense shows more improvement in ART values as compared to best defense. It is so because, with a high client load, small to moderate attack load, naïve defense does not give too many false negatives and only little attack traffic is diverted to active server. While in case of best defense with high client load and low to moderate attack load, large legitimate population is detected as attack and goes to honeypot before being directed to active server, thus increasing the values of ART and giving lower ART improvements.

In case of high client load and high attack load, there is a tradeoff between best defense (which reduces false negatives and hence illegitimate traffic to active servers, thus increasing ART) and naïve defense (which increases false positives and thus diverting the flows to honeypots before directing them to active servers, thus increasing ART). Hence it shows a chaotic behavior.

Entire results are tabulated in the following tables:

Table 10. Modes of operation for goodput

AL/CL	Goodput		
	Low	Mod	High
Low	Na, No, B	Na	B
Mod	Na, No, B	Na	B
High	Na, No, B	Na	B

Table 11. Modes of operation for MTBF

AL/CL	MTBF		
	Low	Mod	High
Low	B	B	B
Mod	B	B	B
High	B	B	B

Table 12. Modes of operation for ART

AL/CL	ART		
	Low	Mod	High
Low	Na	Na	Na
Mod	B	No	Na
High	B	Chaotic	Chaotic

Na : Naïve Defense
No: Normal Defense
B: Best Defense

From the above discussion it is clear that our proposed framework shows significant improvement in important network performance parameters with a minor tradeoff at high attack load and high client load. The priority of output parameters also determines the value of ‘a’. For ex., at low AL and high CL, ART goes in favor of naïve defense whereas goodput goes in favor of best defense. The framework is auto corrective. An increase in ART signifies the presence of attack which triggers lower value of ‘a’ in subsequent detection window to maintain stable network functionality.

9. Advantages of the proposed framework

1. Use of entropy as a measure for statistical deviation at POP enables quick detection of variable rate DDoS attacks.
2. Moving time window based monitoring provides for accurate real time characterization.
3. Honeypots help in retaining attack evidence for forensic purposes and enable smooth operation even under heavy load.
4. Defense model is scalable to the network conditions and attack load.
5. Our framework ensures minimum collateral damage. The connections with the attacks are retained with honeypots for an eon and in case of mischaracterized flow, they are redirected to active FTP server in subsequent time windows. So, legitimate packet loss is minimum.
6. Adaptable to attacker tricks due to the presence of honeypots.
7. As there are no lethargic tables for routing, redirection is based on simple rule generated in real time and states of servers are flushed after each eon, there are less memory overheads in terms of monitored traffic.
8. As we use a simple algorithm to parse the data packets and characterize them, the computation part reduces to just calculating the entropy of flows per time window. The computation of next server set and eon length also requires the use of light weight hash functions. Hence, there are less

computational overheads in terms of correlation / inference and analysis

9. Since there are no bulky update messages, there is less bandwidth overheads in terms of control traffic exchanged between defense modules

10. Conclusions

The framework presented in this paper provides an end to end solution for defense against both diluted degrading and high rate concentrated flooding DDoS attacks in ISP domain. Our characterization scheme calibrates the network to operate in one of the three modes of defense, namely, naïve, normal and best to reduce false positives and false negatives.

ROC curves demonstrate advantage of using the proposed scheme under high rate attacks.

Next, we presented dynamic honeypot based mitigation scheme where dynamic honeypot engine in honeypot controller automatically triggers the generation of appropriate number of honeypots and servers after each eon. Evaluation of the framework on parameters like goodput, mean time between failure and average response time under three modes of operation show that our scheme gives stable network functionality in case of smooth change in client load. In case of abrupt changes, it has the tendency to adapt itself to the network. It experiences a favorable attack load independent behavior for fixed number of attack machines and self optimizes the detection parameters. There is a high percentage improvement in goodput and ART in presence of autonomic dynamic honeypots with minor tradeoff at high client load. Our proposed framework mitigates DDoS attacks meeting the difficult challenges of keeping collateral damage and overheads in terms of memory, computation and bandwidth negligible. Hence the results are promising and show that the framework has the potential to provide stable network functionality even in the presence of high rate attacks.

Some of the avenues for further experimentation are with larger and heterogeneous networks. Policy to distribute keys to legitimate flows depending upon trust level (or duration of their legitimacy) can take the management off the HCE at the expense of increased computation and control messages. Future focus is to apply back tracking on attack flows going to the honeypot servers to reach the attack source and update the filtering rules at POP. Both of them hold promise for evaluating and improving our DDoS defense method. The overhead of state monitoring can be distributed amongst multiple ingress POPs of ISP.

References

- [1] CERT Statistics, URL <http://www.cert.org/stats/cert>
- [2] L. Garber, "Denial-of-Service Attacks Rip the Internet", *IEEE Computer*, vol. 33, no. 4, 2000. pp. 12-17.
- [3] K.J. Houle, G. M. Weaver, N. Long, and R. Thomas. "Trends in denial of service attack technology". *Technical Report Version 1.0*, CERT Coordination Center, Carnegie Mellon University, 2001.
- [4] F. Lau, S. H. Rubin, M. H. Smith, and L. Trajkovi. "Distributed denial of service Attacks". In *Proceedings of IEEE International Conference on Systems and Cybernetics vol. 3*, pp. 2275-2280, 2000.
- [5] C.M. Cheng, H.T. Kung, and K.S. Tan, "Use of spectral analysis in defense against DoS attacks". In *Proceedings of IEEE GLOBECOM 2002*, pp. 2143-2148, 2002.
- [6] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the source". In *Proceedings of ICNP 2002*, Paris, France, pp. 312-321, 2002.
- [7] L. Spitzner, "Honeypots: Simple, Cost-Effective Detection". 30 April 2003. URL: <http://www.securityfocus.com/infocus/1690>
- [8] L. Zhiguang "Honeypot: a Supplemented Active Defense System for network Security". *IEEE.*, 0-7803-7840-7/03, 2003
- [9] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time". *Computer Networks*, vol. 31, nos. 23-24, 1999.
- [10] M. Roesch, "Snort—Lightweight Intrusion Detection for Networks". In *Proceedings of USENIX Systems Administration Conf. (LISA '99)*, Nov. 1999.
- [11] T. M. Gil and M. Poletto. "Multops: a data-structure for bandwidth attack detection". In *Proceedings of 10th USENIX Security Symposium*, 2001.
- [12] 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.
- [13] B. Bencsath, and I. Vajda, "Protection against DDoS Attacks Based on Traffic Level Measurements". Presented at *Western Simulation MultiConference*. San Diego, California, USA., 2004.
- [14] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions," *ACM SIGCOMM*, 2005.
- [15] A. Sardana and R. C. Joshi, "Simulation of Dynamic Honeypot Based Redirection to Counter Service level DDoS Attacks". In *Proceedings of ICISS 2007, Springer LNCS 4812*, pp. 259-262, 2007.
- [16] Y. Xu, and R. Guerin, "On the Robustness of Router-based Denial-of-Service Defense Systems". *ACM SIGCOMM*, 2005.
- [17] J. Ioannidis, and S. M. Bellovin, "Implementing Pushback: Router-Based Defense against DDoS Attacks". *IEEE INFOCOMM*, 2003.
- [18] H. Deng, Q. Zeng, and D. P. Agrawal, "An Unsupervised Network Anomaly Detection System Using Random Projection Technique," *International Workshop on Cryptology and Network Security (CANS)*, Sept. 24-26, 2003.
- [19] B. Stephan, "Optimal filtering for denial of service mitigation," In *Proceedings of the 41st IEEE Conference on Decision and Control*, 2002, Vol. 2, pp. 1428 - 1433, Dec. 2002.
- [20] J. Mirkovic, J. Martin, and P. Reiher, "A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms". *Technical Report 020018*, Computer Science

Department, University of California, Los Angeles, 2002.

- [21] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network". In *ACM SIGCOMM Computer Communication Review*, volume 32, pp. 62–73. ACM Press, 2002.
- [22] T. Peng, C. Leckie, and K. Ramamohanarao, "Defending against distributed denial of service attack using selective pushback". In *Proceedings of ICT*, 2002.
- [23] M. Kalantari, K. Gallicchio and M. A. Shayman, "Using transient behavior of TCP in mitigation of distributed denial of service attacks". In *Proceedings of the 41st IEEE Conference on Decision and Control*, 2002, Vol. 2, pp. 1422 – 1427, Dec. 2002.
- [24] J. Jones. "Distributed Denial of Service Attacks: Defenses", *Technical report*, Global Integrity, 2000.
- [25] A. C. Snoeren, H. Balakrishnan, and M. F. Kaashoek. "The Migrate Approach to Internet Mobility". In *Proceedings of the Oxygen Student Workshop*, July 2001.
- [26] F. Sultan, K. Srinivasan, D. Iyer, and L. Iftode. "Migratory TCP: Connection Migration for Service Continuity in the Internet". In *Proceedings of ICDCS*, 2002.
- [27] P. Dewan, P. Dasgupta, and V. Karamcheti. "Defending against Denial of Service attacks using Secure Name resolution", In *Proceedings of SAM 2003*, 2003.
- [28] S. M. Khattab, C. Sangpachatanaruks, R. Melhem, D. Mosse and T. Znati. "Proactive Server Roaming for Mitigating Denial of Service Attacks", In *Proceedings of the 1st International Conference on Information Technology: Research and Education (ITRE'03)*, August 2003.
- [29] Anjali Sardana, Krishan Kumar and R. C. Joshi, "Detection and HoneyPot Based Redirection to Counter DDoS Attacks in ISP Domain" In *Proceedings of IEEE Third International Symposium on Information Assurance and Security*. Manchester, UK, pp. 191-196, Aug 2007.
- [30] Y. Xiang and W. Zhou. "Classifying DDoS packets in high-speed networks", In *International Journal of Computer Science and Network Security*, Vol. 6, No. 2B, February 2006, pp. 107-115.
- [31] R. L. Rivest and A. Shamir. "PayWord and MicroMint—Two Simple Micropayment Schemes", In *Proceedings of 1996 International Workshop on Security Protocols*, number 1189 in Lecture Notes in Computer Science, M. Lomas, editor, pages 69–87. Springer, 1996
- [32] J. Zhang and M. Zulkernine, "Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection" In *Proceedings of IEEE ICC 2006*, pp. 2388-2393.

Ramesh C. Joshi Born at Pithoragarh (UA, India) on June 1st, 1946, Dr. Joshi received his B.E. degree in electrical engineering from *Allahabad University*, Allahabad, India in 1967. He then received his M.E. and Ph.D. degrees in electronics and computer engineering from University of Roorkee, India, in 1970 and 1980, respectively.

In August 1970 he joined E&CE Dept., University of Roorkee as LECTURER. He then became READER in August 1981 and then PROFESSOR in September 1987. Currently he is working as PROFESSOR in Electronics and Computer Engineering Department, Indian Institute of Technology, Roorkee. He has served as Head of the Department twice from Jan 1991 to Jan 1994 and from Jan 1997 to Dec 1999. He has been Head of Institute Computer Centre, IITR from March 1994- Dec 2005. Presently he is actively involved in research in the areas of Information Security, Bioinformatics, Databases and Reconfigurable Computing.

Dr. Joshi is in expert panel of various national committees like AICTE, DRDO and MIT. He is also a member of National Industrial Research and Development Award Committee. Dr. Joshi has guided about 25 Ph.D. theses M.E./M.Tech. Dissertations (150), M.E./B.E. projects (200+). He has been principal investigator for projects from various national agencies like Ministry of Information and Communication, DRDO, DOE and AICTE He has published over 150 research papers in National/International Journal/Conferences. Dr. Joshi has also done 18 months training at ENSER Grenoble, France under Indo French Collaboration program. He has received a Gold Medal by Institute of Engineers in 1978 for best research paper.

Anjali Sardana Born at Bareilly (UP, India) on August 1, 1982, she received her B. Tech degree in Information Technology from U.P. Technical University, Lucknow, India, in 2004. She has been gold medalist during her undergraduate studies. She received her M.Tech. degree in Information Technology from Indian Institute of Technology, Roorkee in 2006. Currently she is pursuing her Ph.D. in Electronics and Computer Engineering Department, Indian Institute of Technology, Roorkee, India. Her recent significant work on Honeypots and DDoS includes implementation of low interaction honeypots and simulation of high interaction honeypots. Her research interests include Information Security and Bioinformatics.

Ms. Anjali has been recipient of 1st prize at National Level R&D contest organized by Govt. of India, "Best Engineer Award" in the year 2003-2004, has been recipient of three Gold Medals during undergraduate studies, has been honored twice by IMA and been a national level participant at ImagineCup'06 by Microsoft.