NETWORK BEHAVIOR MONITORING AND ANALYSIS FOR DETECTION AND RESISTANCE OF DENIAL OF SERVICE ATTACKS

Omran Ali Bentaher
Faculty of Information Technology, Alasmarya University, Zliten, Libya
omalbeta@yahoo.com
Atia M. Albahbah
Faculty of Information Technology, Alasmarya University, Zliten, Libya
abahbah@yahoo.com

Abstract

Denial of Service (DoS) attacks is one category of internet threats that can cause significant loss of time and revenue[1]. With many ready to use tools available for creating Trojans, Viruses and Worms, it even doesn’t require any programming expertise to launch a DoS kind of attack. Accounting on certain vulnerabilities that exist in TCP/IP protocol, a DoS attack can be launched in a variety of ways. This includes largely flooding and Logical attacks. While flooding is about sending large quantities of legitimate commands to overwhelm the receiver, Logic attacks take advantage of and manipulate particular values of Header fields. This paper studies DoS attacks, by launching them first in a networked scenario and then demonstrating their effect. Then it develops a Host Based Intrusion Detection System (HIDS) to handle this kind of attacks on a single host. The HIDS works by taking following steps. It first tries to prevent the attack by Ingress filtering that is done on the basis of rules already defined. If in some cases some false positives let the illegitimate traffic pass through the filter, a detection scheme is there in place. Once an attack is detected, measures are taken to mitigate the effect of the current attack, and necessary updating is done to prevent such kind of attack in future. The results obtained clearly demonstrate the effects of attack and also demonstrate the way it is mitigated.

Keywords: Denial-of-Service attacks, Ingress filtering, Egress filtering ,Intrusion detection system.

الملخص

إن هجمات الحرمان من الخدمة (دوس) تعتبر واحدة من التهديدات التي تتعرض لها شبكة الإنترنت والتي يمكن أن تسبب خسارة كبيرة في الوقت و هدر موارد النظام. حيث انه لا يتطلب أي خبرة في

البرمجة لإطلاق هذا النوع من الهجوم . يستغل هذا النوع من الهجوم بعض نقاط الضعف الموجودة في TCP/IP بروتوكول للقيام بعدة أنواع منه. في هذه الورقة تمت دراسة هجمات الحرمان من الخدمة من خلال إطلاقها في شبكة ومن ثم إظهار تأثيرها على الهدف و ثم بناء نظام لكشف التسلل للتعامل مع هذا النوع من الهجمات على مضيف واحد. تم التحقق واختبار النظام منخلا لتنفيذ عدة هجمات على جهاز رئيسي ،ومن خلال مراقبة سلوك الشبكة عند الهدف تبين أن هناك استخدام كبير في موارد الجهاز المستهدف (وحدة المعالجة المركزية ، الذاكرة وموارد الشبكة) عندها يتم تحديد عنوان المهاجم من خلال تفحص الحزمة بعدها مباشرة يتم تطبيق قواعد محددة لفلترة الحزم المهاجمة بناءا على TCP/IP بروتوكول .

(i) I.INTRODUCTION

Since the first real world Denial of Service attacks that were publicized in early 2000 that caused significant financial losses to several major e - Business giants these attacks still persist and are constantly evolving in complexity and increasing frequency. We use computer networks for everything from banking and investing to shopping and communicating with others through email or chat programs[2]. So when computer application developed to handle financial and personal data, the real need for security was felt like never before. As you may not consider strangers reading your email, using your computer to attack other systems or sending forged email from your computer.

This paper focuses on different kind of Denial of Service (DoS) attacks, how they are performed and how we can detect and their countermeasures. Denial of Service attack is a process of blocking access to data or systems wherein a user or organization is deprived of resources they would normally expect to have. An attacker sends high-volume of traffic to its target. Denial-of-service (DoS) is a class of attacks in which an attacker attempts to prevent legitimate users from accessing service, such as a web site, data base server, or other system. This can be done by exercising a software bug that causes the software running the service to fail (such as the “Ping of Death” attack, sending enough data to consume all available network bandwidth, or sending data in such a way as to consume a particular resource needed by the service.

II. DoS Attacks Methods

1- Spoofing is the creation of IP packets using somebody else’s IP source addresses[3]. This technique is used for obvious reasons and is employed in several of the attacks discussed later. By examining the IP header , we can see that the first 12 bytes contain various information about the packet. The next 8 bytes, however, contains the source and destination IP addresses. Using one of several tools, an attacker can easily modify these addresses – specifically the "source address" field. Figure 1 shows Spoofed source IP address, illustrates the interaction between a workstation requesting web pages using a spoofed source IP address and the web server executing the requests. If a spoofed source IP address (i.e. 172.16.0.6) is used by the workstation, the web server executing the web page request will attempt to execute the request by sending information to the IP address of what it believes to be the originating system (i.e. the workstation at 172.16.0.6). The system at the spoofed IP address will receive unsolicited connection attempts from the web server that it will simply discard [4].

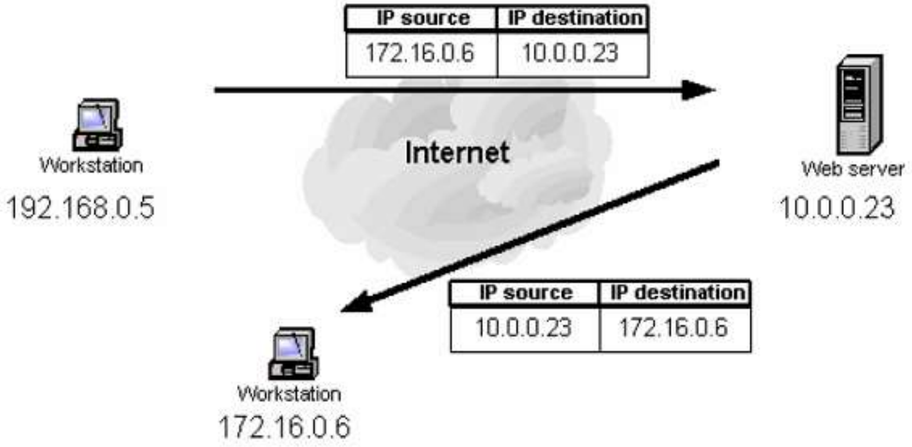


Figure 1: Spoofed Source IP address

2.Flooding

Flooding is one particular kind of tool to launch denial-of-service attack. A perpetrator sends a large number of different kinds of traffic, using a fake source address. The example of flooding attacks are SYN flooding, Smurf IP, UDP flooding and Mail Bomb.

• TCP SYN Flood Attack: This attack exploits the 3-way handshake used for a TCP connection setup [4]. The attacker sends a TCP SYN request to the victim using a spoofed IP address. The victim responds with a TCP SYN- ACK response and allocates memory for the potential connection. It waits for an acknowledgement (TCP ACK) from the attacker. However, since the IP address is spoofed, the victim does not receive a response from the attacker.

After a certain period of time, called the time-out interval, the victim closes the half-open connection and frees up the reserved memory. Since the memory resources at the victim are limited, if the attacker sends enough connection requests, and fast enough, it can tie-up the resources of the victim. Thus, connections from legitimate users cannot be processed.

• UDP Flood Attack: UDP does not require any connection setup procedure to transfer data. An attacker sends UDP packets to random ports on the victim system [5]. Since there have been no service requests from these ports, the victim has effectively wasted CPU cycles and memory resources to process these packets. Large number of such packets tie-up the victim's resources. And since UDP does not have congestion control, this attack can also be used to target the bandwidth resources of the victim's network.

• Smurf IP Attack: Forged ICMP packets are sent by the attacker to the broadcast address of a network [6]. All the systems on the network then send an ICMP reply back to the victim. This large volume of replies inundates the victim's bandwidth.

• Mail Bomb: A mail server can fail if a very large number of bogus emails are sent in a very short time. The focus of this project is on Flooding type of attacks [7].

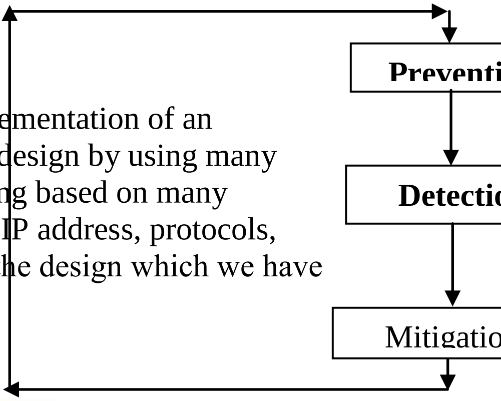
III. Countermeasures of DoS Attacks

Designing and implementing effective countermeasures against saturation denial of service attacks seems to be a simple and straightforward task [9]. But it poses several challenges, due to which these attack which security desiners understand well are still under way. Some of the reasons for these are:

- Availability of easy and free to use tools for launching and managing attacks, which even not so expert cracker can also make use of.
- Similarity of attack traffic to the legitimate traffic, makes them challenging to distinguish.
- Systems across the internet with poor security implementations serve as great havens for attackers. The number of such systems will continue to be always there.

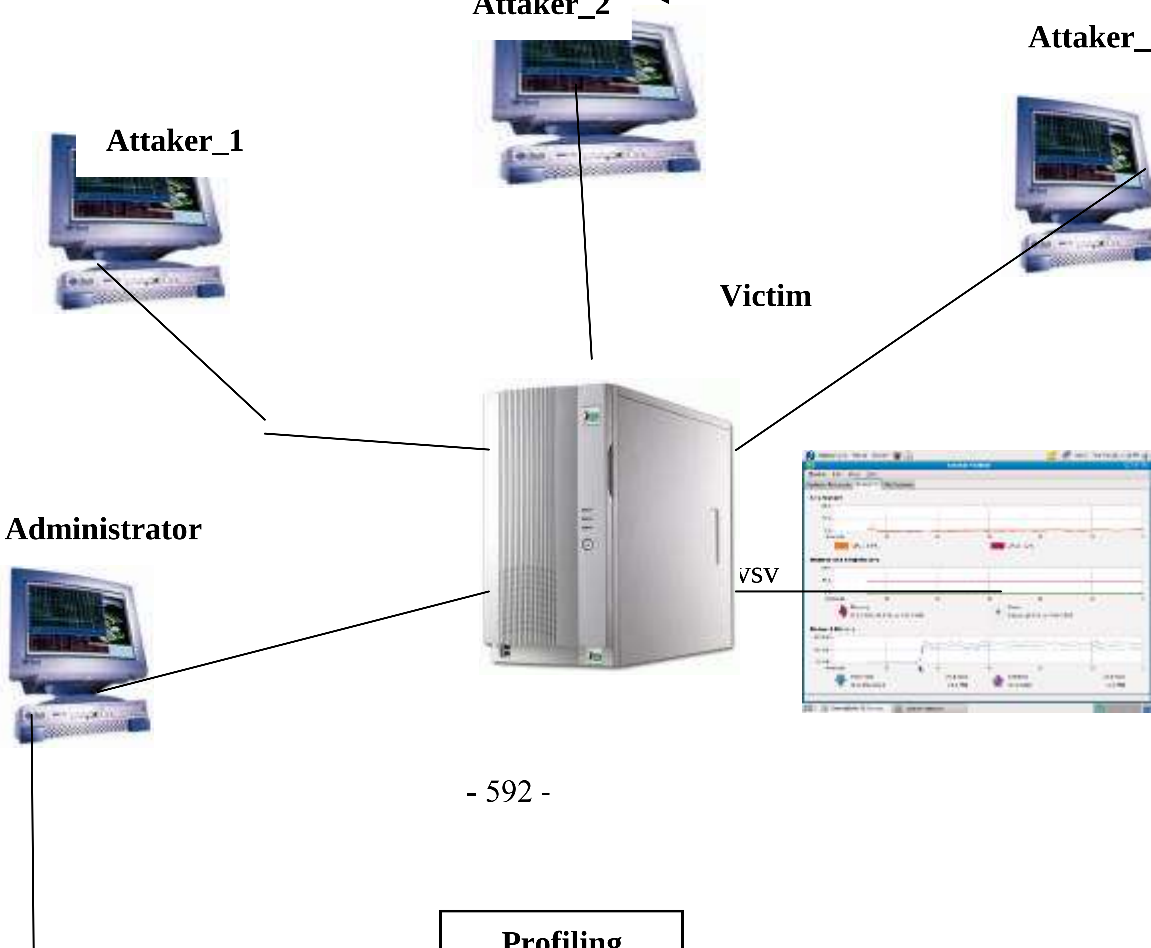
- The scaled form of DoS attacks that is distributed DoSattacks, provides further challenges in front of security professionals due to their technology and manner in which they are launched [8].

Even in presence of above challenges, a lot of research is being carried out to handle them. Given this the major interest has shifted to local detection and mitigation techniques, the counter measures include three phases as shown in following flowchart.



V. Overview of system architecture.

In this paper we describe the designed and implementation of an anomaly detection system ,which we tested this design by using many types of DoS attacks and applying ingress filtering based on many parameters which tcp packets caring like source IP address, protocols, TTL,……. The following architecture showing the design which we have implemented .



Architecture Design of HIDS

The attackers send useless packets just to saturate and consume the victim resources. At receiver side the sniffing program is there just to sniff the packets and mark them based on filtering rules. The filter will block the illegitimate packets and accept legitimate once. At receiver site also there is log file created for profiling purpose monitoring which system can decide whither some legitimate packets marked as illegitimate and blocked and need to modify the filtering rules to achieve the maximum correct result.

IV. The Process of modeling :

The figure below shows the modeling process is performed using a six-stage process.

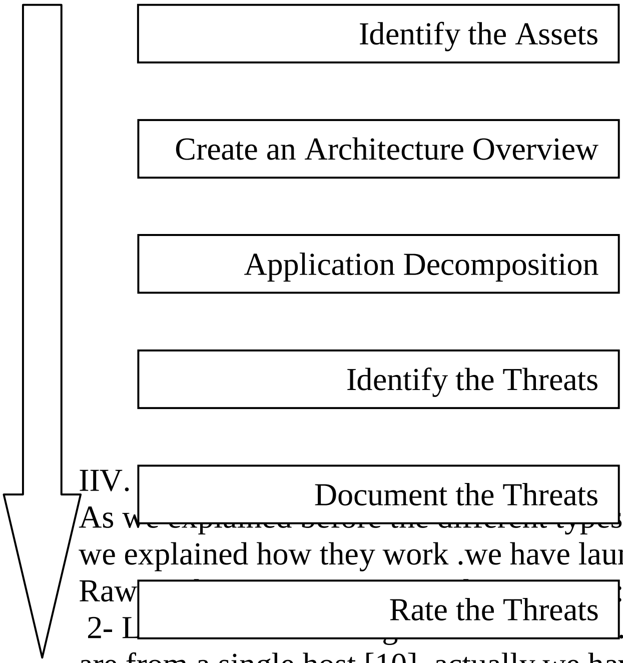


Fig 3.3 Threat modeling process followed to model the DoS attacks

IIV.

As we explained before the different types of denial of service attacks and we explained how they work .we have launched many DoS attacks using Raw :- 1- TCP SYN Flooding Attack. 2- L . All these attacks we launched are from a single host [10], actually we have used one host as an attacker packets to a victim to Establish a connection but the connection will remain half- Open connection as the victim would not receive the ACK from the host which it's IP address was spoofed as shown in fig 1 before . At the victim side sniffing tool was there which is created by our own code using raw socket program this system sifted incoming packets and mark them based on filter rules that administrator decide wither will be accepted or blocked them, then log file created automatically while marking the packets, by this log file we can realize and analyze packets and see the false positive and false negative so according to that we can update the filter rules which we used .we kept monitoring and watching the system monitor by using system monitoring tool supported by Linux and have seen many changes and variation in band width consumption , memory usage and CPU usage) and we took many snapshots showing those variation and they are mention below .

We realized that after flooding the system the victim resources effected .The filtering rules were based on different parameters available in headers that packets caring (i.e IP protocol , source IP address ,TTL ,….)

Test Cases:

These test cases shown the changes in input that we can make in our program and the output which is shown in next figures (many cases done in our design. The inputs where as shown in figure(2) and figure (3) . My input are :-

(i) -t which mean the target that I am going to attack (i.e 172.16.78.54).

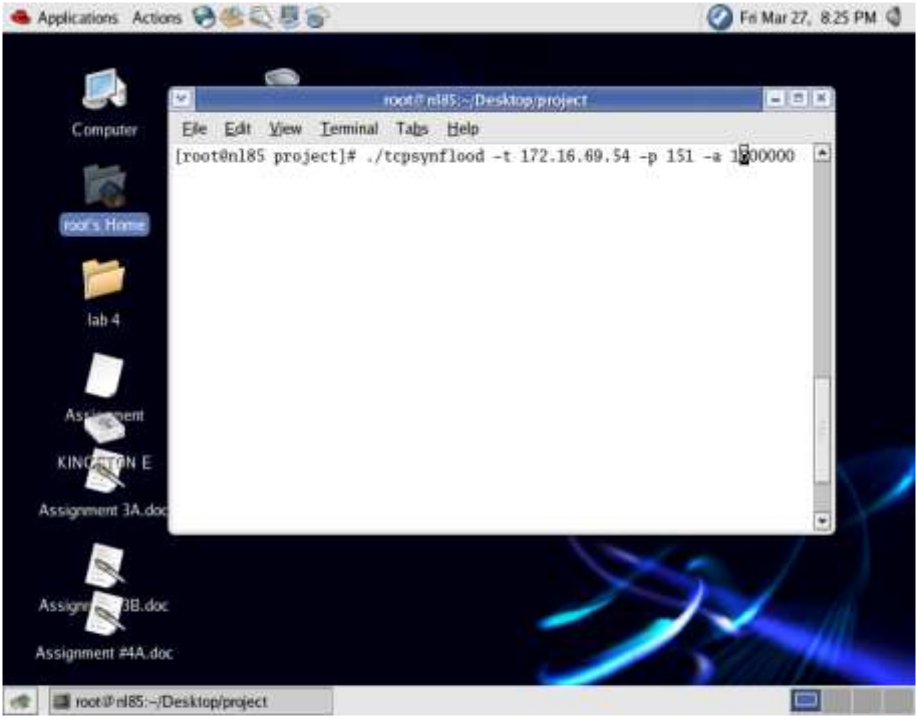
(ii) - p means the port number that the target going to receive from (i.e 150).

(iii) - a means the number of packets that it will flood the t .

The input is like that:-

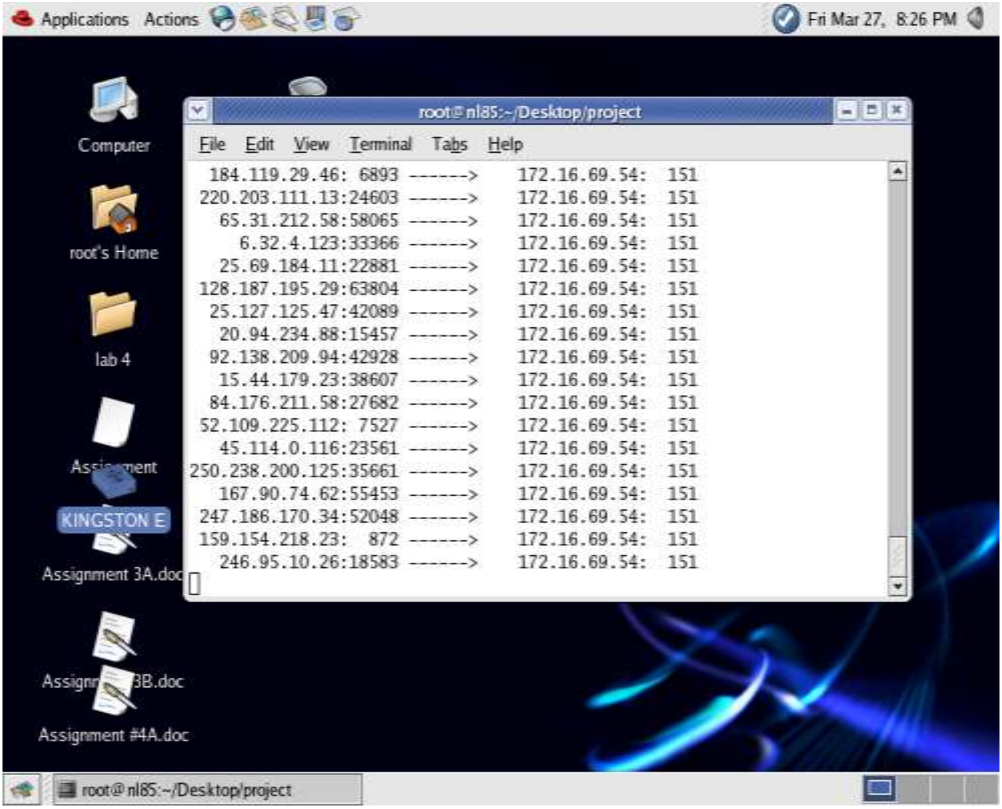
```
./tcpsynflood –t 172.16.69.54 –p 151 –a 100000
```

By implementing this test that means we flood the victim t 172.16.69.54 using port no. 151 by 100000 packets(100000 half open connection will created) that means we have used this much number(100000) spoofed IP addresses as shown in fig(4.4).



```
[root@nl85 project]# ./tcpsynflood -t 172.16.69.54 -p 151 -a 1000000
```

Fig (2) test case during running the program



```
  184.119.29.46: 6893 ------>    172.16.69.54:  151
 220.203.111.13:24603 ------>    172.16.69.54:  151
   65.31.212.58:58065 ------>    172.16.69.54:  151
     6.32.4.123:33366 ------>    172.16.69.54:  151
   25.69.184.11:22881 ------>    172.16.69.54:  151
 128.187.195.29:63804 ------>    172.16.69.54:  151
  25.127.125.47:42089 ------>    172.16.69.54:  151
   20.94.234.88:15457 ------>    172.16.69.54:  151
  92.138.209.94:42928 ------>    172.16.69.54:  151
   15.44.179.23:38607 ------>    172.16.69.54:  151
  84.176.211.58:27682 ------>    172.16.69.54:  151
 52.109.225.112: 7527 ------>    172.16.69.54:  151
   45.114.0.116:23561 ------>    172.16.69.54:  151
250.238.200.125:35661 ------>    172.16.69.54:  151
   167.90.74.62:55453 ------>    172.16.69.54:  151
 247.186.170.34:52048 ------>    172.16.69.54:  151
 159.154.218.23:  872 ------>    172.16.69.54:  151
   246.95.10.26:18583 ------>    172.16.69.54:  151
```

Fig (3) Spoofed IP addresses after run the flooding program

Test case 2:
The test cases onwards here are targeted for sniffing and filtering program that is deployed on the victim machine which prevents from the DoS attacks by filtering the unexpected packets based on filtering rules.
Input Values.

(ii) tcp[13]==0x02 and (dst port 151) and (host 172.16.69.54 - 172.16.69.60)

Command used to filter out the flooded packets
./sniftcp[13]==0x02 and (dst port 151) and (host 172.16.69.54 - 172.16.69.60)
Only those packets will be accepted which is tcp packet that has the SYN flag set and having destination port 151 and source ip address lying in 172.16.69.54 - 172.16.69.60 range only.

Test case 3:

Input Values.

(i) (host 172.16.69.54 - 172.16.69.60)

Command used to filter out the flooded packets

```
./snif(host 172.16.69.54 - 172.16.69.60)
```

Only those packets will be accepted which has source ip address in 172.16.69.54 - 172.16.69.60 range only.

Test case 4:

Input Values.

(i) (port 80)

Command used to filter out the flooded packets

```
./snif(port 80)
```

Only those packets will be accepted has port number 80 only.

Test case 5:

Input Values.

(hc 2)

Command used to filter out the flooded packets

```
./snif(hc 2)
```

Only those packets will be accepted which has hop count 2 and this applies for all source ip addresses.

Test case 6:

Input Values.

(host 172.16.69.54 - 172.16.69.60) and (hc 2)

Command used to filter out the flooded packets

```
./snif(host 172.16.69.54 - 172.16.69.60) and (hc 3)
```

Only those packets will be accepted which has hop count 2 and source ip addresses lies in range 172.16.69.54 - 172.16.69.60.

Test case 7:

Input Values.

(host 172.16.69.54 and hc 2) or (host 172.16.69.57 and hc 5)

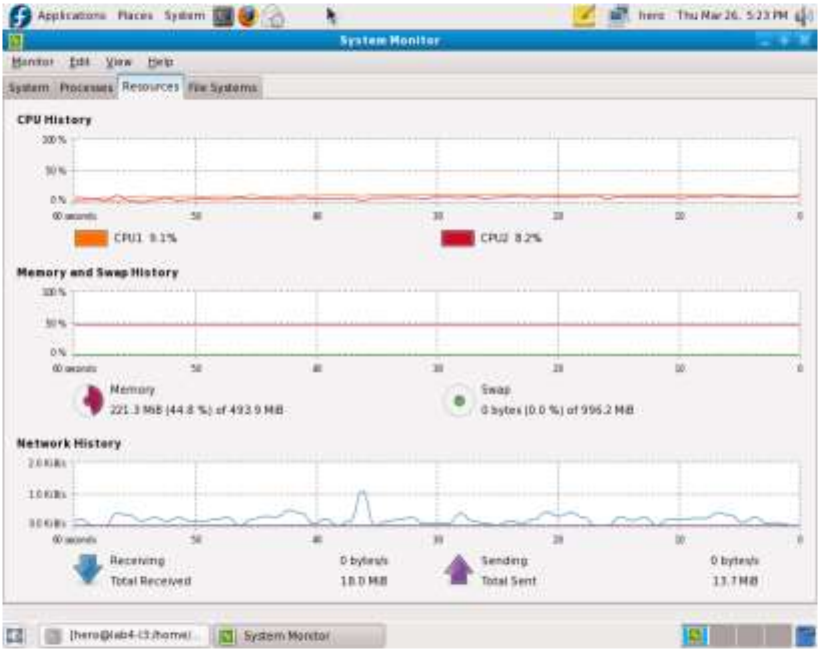
Command used to filter out the flooded packets

```
./snif(host 172.16.69.54 and hc 2) or (host 172.16.69.57 and hc 5)
```

The packets which come from host 172.16.69.54 and has hop count 2 also the packets which come from host 172.16.69.57 and has hop count 5 will only be accepted.

Results

Network behavior analysis (NBA) technology help you detect and stop suspicious activity on corporate networks well before it becomes harmful for the organization. Network behavior-analysis systems promise to add another layer of security to corporate networks by watching traffic for changes in typical actions. The systems typically perform a benchmark of traffic behavior and monitor for changes. Then if, for example, a relatively unused server begins to propagate many requests, the anomaly-detection system might suspect the host could be falling victim to a Denial of Service Attack. The products analyze network traffic through data gathered from devices such as IP traffic flow systems, or via packet analysis. They use a combination of signature and anomaly detection to alert security and network managers of any activity that appears to be out of the norm, providing a view of the network that lets managers analyze activity and respond before there's damage to systems and data. The next snapshots will show the result and the behavior of the system before and after the flood the victims and after apply the ingress filtering based on some parameters as we explained before .



CPU1 9.1%
CPU2 8.2%

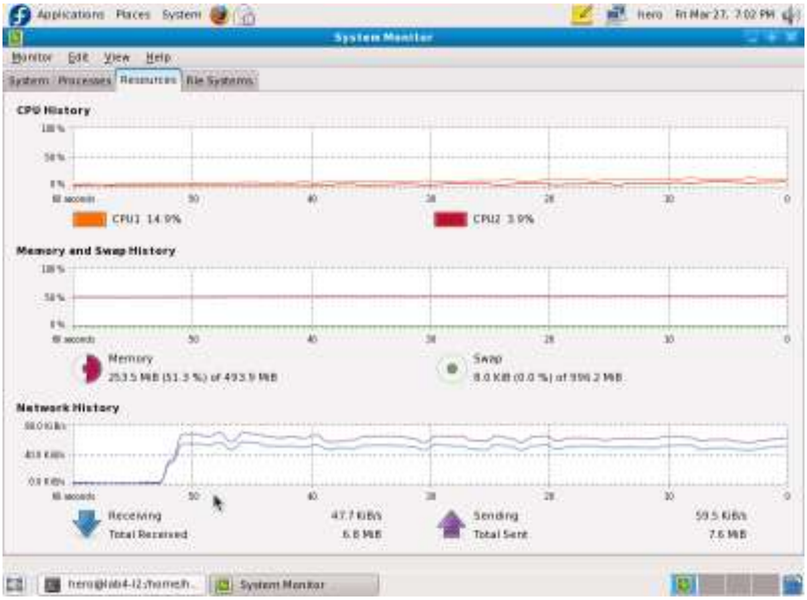
Memory
221.3 MiB (44.8 %) of 493.9 MiB

Swap
0 bytes (0.0 %) of 996.2 MiB

Receiving 0 bytes/s
Total Received 18.0 MiB

Sending 0 bytes/s
Total Sent 13.7 MiB

SnapshotNo. 1 When legitimate traffic going on.



CPU1 14.9%
CPU2 3.9%

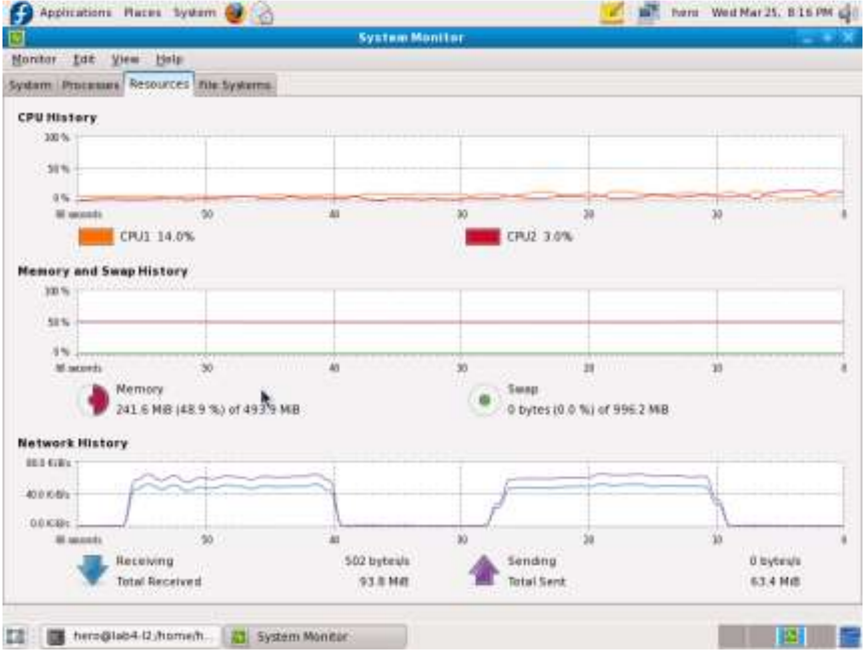
Memory
253.5 MiB (51.3 %) of 493.9 MiB

Swap
8.0 KiB (0.0 %) of 996.2 MiB

Receiving 47.7 KiB/s
Total Received 6.8 MiB

Sending 59.5 KiB/s
Total Sent 7.6 MiB

Snapshot No.2 After flooding the Victim by sending useless packets



CPU1 14.0%
CPU2 3.0%

Memory
241.6 MiB (48.9 %) of 493.9 MiB

Swap
0 bytes (0.0 %) of 996.2 MiB

Receiving 502 bytes/s
Total Received 93.8 MiB

Sending 0 bytes/s
Total Sent 63.4 MiB

Snapshot No. 3 after Server side protect himself by plying packet filtering.

Conclusion

we verified by implementing, launching, and monitoring the network behavior on the victim system that is possible to detect and filter the illegitimate traffic as we explained earlier.
Denial of Service (DoS) attacks is a serious threat for the Internet. DoS attacks can consume memory, CPU, and network resources. Results of launching attack show that it is difficult to detect the DoS attack in early stages. It is hard, though, to deploy ingress filters in all Internet domains. If there are some unchecked points, it is possible to launch DoS attacks from those points.

References

[1]- AhsanHabib, Mohamed M. Hefeeda, and Bharat K. Bhargava“DetectingService Violations and DoS Attacks” NDSS 2003, San Diego, California
[2]- ISS X-Force, “Internet Risk Impact Summary” Sep27,2002.URL: https://gtoc.iss.net/documents/summaryrep.
[3]- “RFC 791 – Internet Protocol: Protocol Specification”, Defense Advanced Research Projects Agency, September 1981.
[4]- “RFC 793 – Transmission Control Protocol: Protocol Specification”, Defense Advanced Research Projects Agency, September 1981.
[5]- “RFC 768 – User Datagram Protocol”, J. Postel, ISI, August 1980.
[6]- “RFC 792 – Internet Control Message Protocol”, J. Postel, ISI, September 1981.
[7]http:/www.iss.net/security_center/advice/Underground/Hacking/Methods/Technical/Spoofing/default.htm.
[8]- Wei Chen , Dit-Yan Yeung , “Defending Against TCP SYN Flooding Attacks”, International Conference on Systems and on Mobile Communications , Osaka, Japan, Nov. 2006.
[9]- P. Ferguson and D. Senie. Network ingress filtering : Defeating denial of service attacks which employ IP source address spoofing. International Journal of Network Management Volume 15 , Issue 1 (January 2005.
[10]-S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Technical Report 99-15, Department .
[11] - Cheng Jin, Haining Wang Kang G. Shin, “Hop-Count Filtering: An Effective Defense Against Spoofed Traffic” Conference on Computer and Communications Security, 2008.