DETECTION AND PREVENTION OF SYN FLOOD DOS ATTACK BY RMON/SNMP

Lavita kathuria[1]
Noida International University, UP, (INDIA)

ABSTRACT

Denial of Service (DoS) attacks are a serious threat for the Internet. DoS attacks can consume memory, CPU, and network resources and damage or shut down the operation of the resource under attack. RMON is a special purpose Management Information Base (MIB) designed for the SNMP (Simple Network Management Protocol), which minimize the load of traffic on the network. RMON analyze the network traffic. The problem is SYN Food DOS attack which send high-rate small packets and consume lots of resources. Due to this resources are busy all the time. And the new connection made by the client is not proceed further. This half-open connection is the SYN Flood DOS attack which can clash the server. Therefore, in this paper, we try to find out the solution to detect and prevent this type of DOS attack by the use of the DOS attack, RMON, SNMP, Time Out, Timestamp, TCP Sequence Number, SYN cookies.

Keywords: DOS attack, RMON, SNMP, Time Out, Timestamp, TCP Sequence Number, SYN cookies.

I INTRODUCTION

An incrementally deployable mechanism to defend against SYN Denial-of Service (DoS) attacks, which are serious threats to the availability of Internet services. There are two common types of DoS attacks: (a) the bandwidth attack designed to consume all available resources on the target's Internet connections, and (b) the SYN flooding attack, where TCP SYN packets with falsified unreachable source addresses are continuously sent to the target so that all slots for accepting TCP connection requests are tied up waiting for acknowledgment packets that never come. Both attacks are serious threats. For example, in 2000, Yahoo, eBay and Amazon were brought down for hours by a bandwidth DoS attack while Microsoft was the target of a serious SYN DoS attack. SYN attacks are still relatively easier to perform because they require significantly less resources than bandwidth attacks.[1]

Resource consumption attacks which degrade the ability of the device to function, such as opening many simultaneous connections to the single device. Bandwidth consumption attacks which attempt to overwhelm the bandwidth capacity of the network device.[2] Network with small bandwidth may suffer from high bandwidth consumption instantaneously if it becomes target.[2]

The goal of the work is to increase the difficulty of carrying out SYN attacks by using the RMON with SNMP and prevent the SYN attack by using the SYN cookies and Time out. Defending against SYN DoS attack is an active _eld of research. A number of defences such as adjusting the timeout values for SYN/ACK packets, firewall-based approaches, Berkeley/ Linux/Reset cookies, and random dropping of SYN packets have been proposed [7], [8] [6].

Each half-open connection will remain on the memory stack until it times out, it will retransmit the SYN-ACK 5 times, doubling the time-out value after each retransmission. The initial time-out value is 3 seconds, so retries are attempted at 3, 6, 12, 24, and 48 second.[5]

TCP timeout mechanism was designed for loss recovery in the presence of severe congestion. TCP *fast* retransmits a lost packet when detecting three duplicate acknowledgements (DUPACKs). In case of severe congestion, however, the sender may not be able to receive enough DUPACKs. When an outstanding packet has not been acknowledged for an interval of Retransmission Timeout (RTO), it incurs a timeout. TCP interprets timeout as an indication of severe congestion and reduces its window to one packet. It then retransmits the lost packet and waits for the ACK. Note that since TCP is self clocked, in case the ACK does not come in, TCP will not send any other packet until it times out again. In most implementations, a minimum RTO (minRTO) of 1 sec is imposed following the recommendation in [18]. This fixed value makes TCP vulnerable to low-rate DoS attacks. Initially, the attacker sends a burst of packets to the bottleneck queue and fills it quickly (assuming the attacker can send at a high rate). TCP packets in pre-existing connections traversing the bottleneck are dropped due to buffer overflow, forcing these connections to time out. Since TCP connections have minRTO = 1 sec, they will time out virtually at the same time, i.e., after 1 sec. The attacker temporarily holds its transmission until 1 sec later when TCPs start to transmit again, forcing them to another round of timeout. Such periodic attacks have low overall rates but can cause huge damage to TCP.[9]

We divide the approaches for dealing with DoS attacks into two main categories: *detection* and prevention approaches. The detection approaches capitalize on the fact that appropriately punishing wrong doers (attackers) will deter them from re-attacking again, and will scare others to do similar acts**.** The obvious way to detect an attack is just waiting till the system performance decreases sharply or even the whole system collapse.[12]

We propose a more effective method for detecting attacks before they harm the system. We propose to use RMON and packet filtering method. RMON monitor the network traffic and packet filtering method filter the packet from DOS Attack.

II LITERATURE SURVEY

DOS is an acronym for Denial of Service attacks. It is a generic term that refers to an resource-depleting attack on a server on a network or on the Internet such that the server would no longer be able to serve legitimate users. The term DoS does not define what mechanisms are being used in such an attack, as there are many ways a DoS attack can be achieved. and mechanisms to achieve this, and DoS does not itself does not define what, as there can be many ways of it is usually done either by exploiting a known vulnerability of the server software in order to crash or cripple the server, or by flooding the server with excessive amount of traffic such that all available resources on the server are consumed, rendering the server incapable of serving further requests from legitimate users.

Classic DoS attacks consist of only one machine attacking the victim. Usually, a single machine is not powerful enough to consume all the bandwidth that the server has, therefore a bandwidth-depleting DoS attack is normally not feasible. However, Other forms of resource-depleting attacks can be done, such as exploiting known vulnerabilities of the server software in order to crash or cripple the server, or

by exhausting the memory resource of the server through TCP SYN-flooding, which we will be explained in more details later.

TCP SYN flooding is a type of DoS attack that generates many bad TCP SYN packets. In a normal connection between the client and the server using TCP, the client first initiate the connection using a TCP SYN packet, then the server responded with a SYN/ACK packet, and then finally the client returns an ACK packet to establish the connection. This is commonly known as a 3-way handshake. In a TCP SYN attack, the client (attacker) initiates a 3-way handshake but never finishes it. In other words, it only ever sends TCP SYN packets, with no final ACK packets. This will cause the server to reserve a memory slot for each unfinished connection. Once the server's memory is filled up with unfinished connections due to flooding of these packets, the server will stop accepting any more incoming requests until these memory slots are freed. Usually, these TCP SYN packets use bogus IP addresses to prevent tracing of the attacker. Non-existent IP addresses will also ensure no replies are returned to the server.[13]

The idea of SYN cookies is that the server tries to store authentication information in the server sequence number of the SYN/ACK packets. This idea is similar to cookies used in web sites where the server stores information of the current session in a cookie and return it to the client together with the web page, hence the name SYN cookies. When the server replies with a SYN/ACK packet, it uses a specially designed formula to calculate the server sequence number (used as an authentication cookie) and passed it into the SYN/ACK packet. The cookie value calculated by the formula is determined by the source address, the source port, the client address, the client port, the client sequence number, client MSS, a counter that changes approximately every minute and a secret value that changes at every server boot. These information are merged together and encrypted using the MD5 one-way hash. When the client later replies with the ACK packet to complete the connection establishment, theserver will verify the server sequence number of the ACK packet to ensure the client is a legitimate user. Since the cookie is encrypted with a one-way hash, it is difficult to reverse-engineer the cookie directly. The cookie formula is also designed so that it will not give out a slightly smaller number than a recent value between the same hosts and ports. This is to prevent confusion with older packets from previous connections that may still be around.[13]

We use Remote Monitoring (RMON) data to model of network traffic behavior and to detect flow-based, bandwidth exhaustion DoS attacks. RMON [15] is a Management Information Base (MIB) specification for use with the Simple Network Management Protocol (SNMP) [16] that monitors attributes of network devices. Within the family of RMON MIBs, RMON1 identifies attributes of low-level ethernet traffic that can be used to characterize network utilization through byte, packet, and error counts. The RMON1 specification is supported by most enterprise routers manufactured by CISCO and NORTEL [17] and thus removes the need for special purpose DDoS detection hardware. Moreover, using the SNMP management model requires no reconfiguration of the network being observed. An SNMP network manager periodically queries an agent for attributes of interest and SNMP traffic and only introduces a small amount of load on the network.[14]

Reno-based TCP variants have two mechanisms associated with data retransmission: fast retransmit/fast recovery and timeout. When losses are sporadic, the sender may retransmit a lost

packets upon receiving three DUPACKs. When losses are dense, however, there may not be a sufficient number of DUPACKs to trigger fast retransmit. If the sender has not received an ACK on an outstanding packet for a certain period of time since the packet was sent, it times out. TCP interprets the timeout as indication of severe congestion. It reduces its congestion window to one packet and retransmits the lost packet. Time between when the packet was sent and when the timeout occurs is called RTO. Base RTO is computed as *max{SRTT + 4 x RTTV AR; minRTO}*. The minimum RTO, minRTO, was recommended as 1 sec in [1] for the purpose of achieving near-maximal throughput. It has been shown that *SRTT* +4x*RTTV AR* is usually less than 1 sec, so effectively most TCP connections have the base RTO = 1 sec. In addition, TCP uses Karn's clamped retransmit backoff algorithm in case of consecutive timeouts: each successive RTO is double the value of the previous until it reaches 64 times the base. In other words, if *SRTT* +4x*RTTV AR* < *minRTO* = 1*sec*, then RTO can be 1, 2, 4, 8, . . . , 64 sec. For more details on the TCP timeout mechanism, please see [9][10][11].

III PROPOSED WORK

The proposed work is discussed under heads of section 3.1 and section 3.2.

3.1 Architecture of Detection and prevention of SYN flood DOS attack by RMON/SNMP

The architecture illustrated in figure 1.1, different components have different working. In this architecture the detection and prevention module which works together to detect the DOS attack and prevent from it. The working of the components is given below:

Firstly, the client and the attacker tries to make the connection with the server with TCP connection with the three way-handshake mechanism. Suppose the client is making the request but the client is not responding to server response is SYN+ACK

Because the attackersend the requests to the server but attacker not reply. The attacker spoofed the source IP address so in this way the client sent the ACK to the Non-exist host7.so this is the case of SYN flood DOS attack.Now the detection mechanism words on it. The RMON Probe analyse the network traffic. It only analyse and capture the packets from network. Decrease the load of SNMP.SNMP send query to the client for sending good packet.

Packet filtering module filter the good packet from capture packet. and the return captured packet compare with the threshold if it exceed then it will be discard. Time out is the when client send good packet the it will loss, the timeout retransmit the packet again. And now the prevention part, if the SYN+ACK match with the TCP Sequence number if it match then connection is right. but the connection can be duplicate then to avoid this, there is timestamp mechanism which find out the uniqueness of the connection. Then the connection is established.

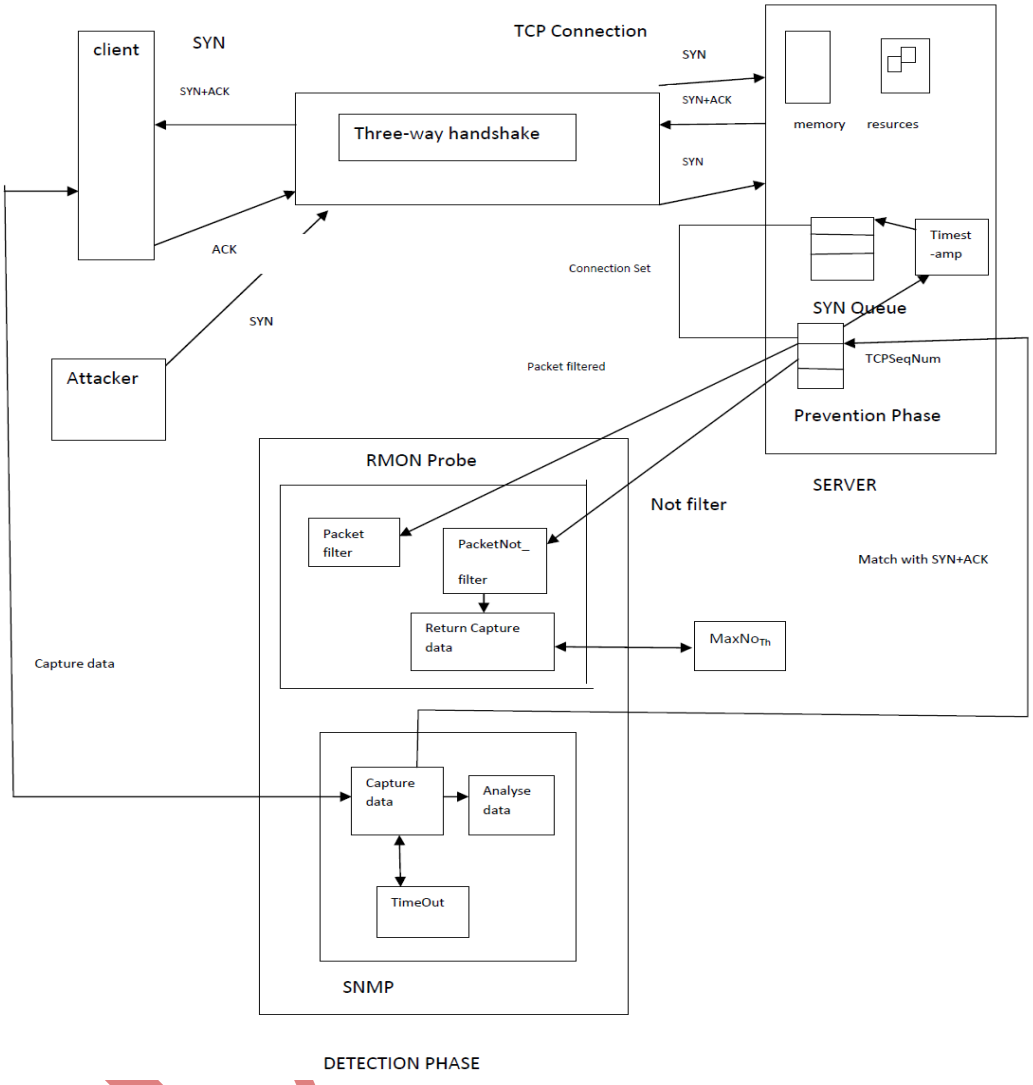

Figure 1.1:Architecture for detecting and preventing Dos attack

3.2 Mechanism of Detection and prevention of SYN flood DOS attack by RMON/SNMP

This mechanism is discussed in phases as follows:

Phase –I

FlowBasedDOSAttack ( )

{

- Attacker going to contact the victims.
- Start consuming target node resources like CPU cycle, memory and bandwidth.

- The end to end attack like TCP SYN attack will exhaust available memory on target machine with connection requests.

}

CPUCyclesAttack( )

{

- CPU prevent legitimate processes from being executed because memory is full of all TCP SYN attack then no legitimate process get response by the CPU.
- Try to decrease the priority of process from higher priority value to lower priority value such that it become starved for CPU time.

}

MemoryStealingAttack ( )

{

- TCP SYN packet received at server and the memory is blocks with TCP SYN which is used for TCP connection control.
- If server receive a large number of TCP SYN packets then sever will run out of memory.
- The attacker use IP spoofing technique to prevent legitimate packets.
- The victim spoofed the address of non-existing hosts to receive number of TCP SYN.
- Victim cannot receive TCP ACK packet because of there is no host respond to it.
- **SYN Flood Attack Connection( );** //the half open connection saved in server memory and attack occur

.}

BandwidthConsumptionAttack ( )

{

- **TCP Three-way Handshake Connection ( );** //The host and attacker will connect with TCP three-way handshake.
- Server creates a long and separate data structure for every incoming SYN Packets.
- When server send request information into the memory stack then the server will wait for the conformation send by the client.
- If the request is waiting to be confirmed by the server, it will remain in memory stack.
- The source IP addresses used in SYN flood attacks then the server will not receive confirmation packets for requests.
- Half-open connection will remain on the memory stack until it times out. This causes stack full.

- Attacker can consume bandwidth by transmitting any traffic on the victim's network connection.
- Attacker send TCP SYN packet to Consume all available bandwidth between an ISP and victim's site.
- The attack traffic such as IP raw packets and ICMP primarily target to consume victim's bandwidth.
- Server send large number of small packets which required a large number of processing resources on the victim's side.

}

TCPSYNAttack( )

{

TCPThree-wayHandshakeConnection( )

{

- The client requests a connection by sending a SYN (*synchronize*) message to the server.
- The server *acknowledges* this request by sending SYN-ACK back to the client.
- The client responds with an ACK, and the connection is established.

}

SYNFloodAttackConnection ( )

{

- The SYN flood attack not responding to the server by sending expected ACK code.
- The malicious client not send expected ACK back or spoofing the source IP address of SYN.
- The server will send the SYN-ACK to falsified IP adress,so that the non-exist host will not send ACK.
- Because of half-open connection it will consume resources on server unil no new connection send by the host.

}

}

Phase – II: Detection

Detection overflow-based DOS attack

{

SetThreshold ( )

{

- It set the thresholds on traffic indicators when it exceeded, it indicate the presence of DoS traffic.

}

RMON_ Probe( )

{

- **Capture packet ( );** //Probe is the monitoring device and act as a server to capture all packets.
- RMON software agent collects information from network traffic and analyze packet.
- Host and application communicate via SNMP over the network.
- **Return Capture packet ( );** //Remote probe devices will reduce the SNMP traffic on network & processing the load on management station.
- And the Information which is collected from network is transmitted to the management station when it is required.
- **Bandwidth Consumption Attack();**//RMON detect bandwidth DOS attack.
- **Flow Based DOS Attack( );**//RMON detect flow-based DOS attack.

}

Capture_ packet ( );

{

- Probe must capture all packets on the locally-attached network, it also including packets of third parties via fetching IP addresses.
- These packets are analyzed to collect network addresses, protocol usage information.
- **Time Out(** *max{SRTT + 4 x RTTV AR; minRTO}* **)**//retransmit the lost data

}

ReturnCapture_ packet ( )

{

- It try to collect data.
- **Set_ Threshold( );** //compare the thresholds value with return capture data.

- It sends traps to management station.
- Management station is simply the host.
- SNMP will not use these captured data.

}

SNMP( )

{

- **Packet Filtering Process( );** //SNMP network manager periodically queries an agent for sending packet filtering
- SNMP traffic and only introduces a small amount of load on the network.
- Because of packet filtering SNMP takes fewer resources.

}

PacketFilteringProcess ( )

{

If (packet Start_ size=64 && packet End_ size= 1518)//packet size in octets long,in case of Ethernet

{

Packet filter without the DOS attack;

}

}

PacketNotFilter( )

{

If(packetStart_size< 64 && packetEnd_size<1518)

{

Packet not filter the DOS attack;

}

}

<u>Phase –III: Prevention of_flow-based DOS attack</u>

{

Step-1.

SYNCookies()

{

- SYN cookies allow a server to avoid dropping connections when the SYN queue fills up.

- The server behaves as if the SYN queue had been enlarged. The server sends back the appropriate SYN+ACK response to the client but discards the SYN queue entry.
- **TCPSequenceNUM( );** //If the server receives a ACK response from the client, the server will reconstruct the SYN queue entry using information encoded in the TCP sequence number.

}

TimeOut(*max{SRTT + 4 x RTTV AR; minRTO}* **)**

{

- Every time it will increase the Timeout time.
- the sender may retransmit a lost packets upon receiving three DUPACKs
- If(*SRTT +4xRTTV AR < minRTO)* = 1*sec*,
- Gives Maximam throughput
- RTO=1,2,3……..64sec

}

TCPSequenceNUM( )

{

- The server tries to store authentication information in the server sequence number of the SYN/ACK packets.
- When the server replies with a SYN/ACK packet, it uses a server sequence number and passed it into the SYN/ACK packet.
- The cookie value is Calculated by the source address, the source port, the client address, the client port, the client sequence number, client MSS, a counter that changes approximately every minute and a secret value that changes at every server boot.
- When the client later replies with the ACK packet to complete the connection establishment,The server will verify the server sequence number of the ACK packet to ensure the client is a legitimate user.

}

TimeStamp( )

{

- If (SYN+ACK=TCPSeqNum && TCPSeqNum!=SYN queue)
- {
- Packet is unique;

}

}

}

3.3. Flow Chart Of Detection And Prevention Of SYN Flood DOS Attack

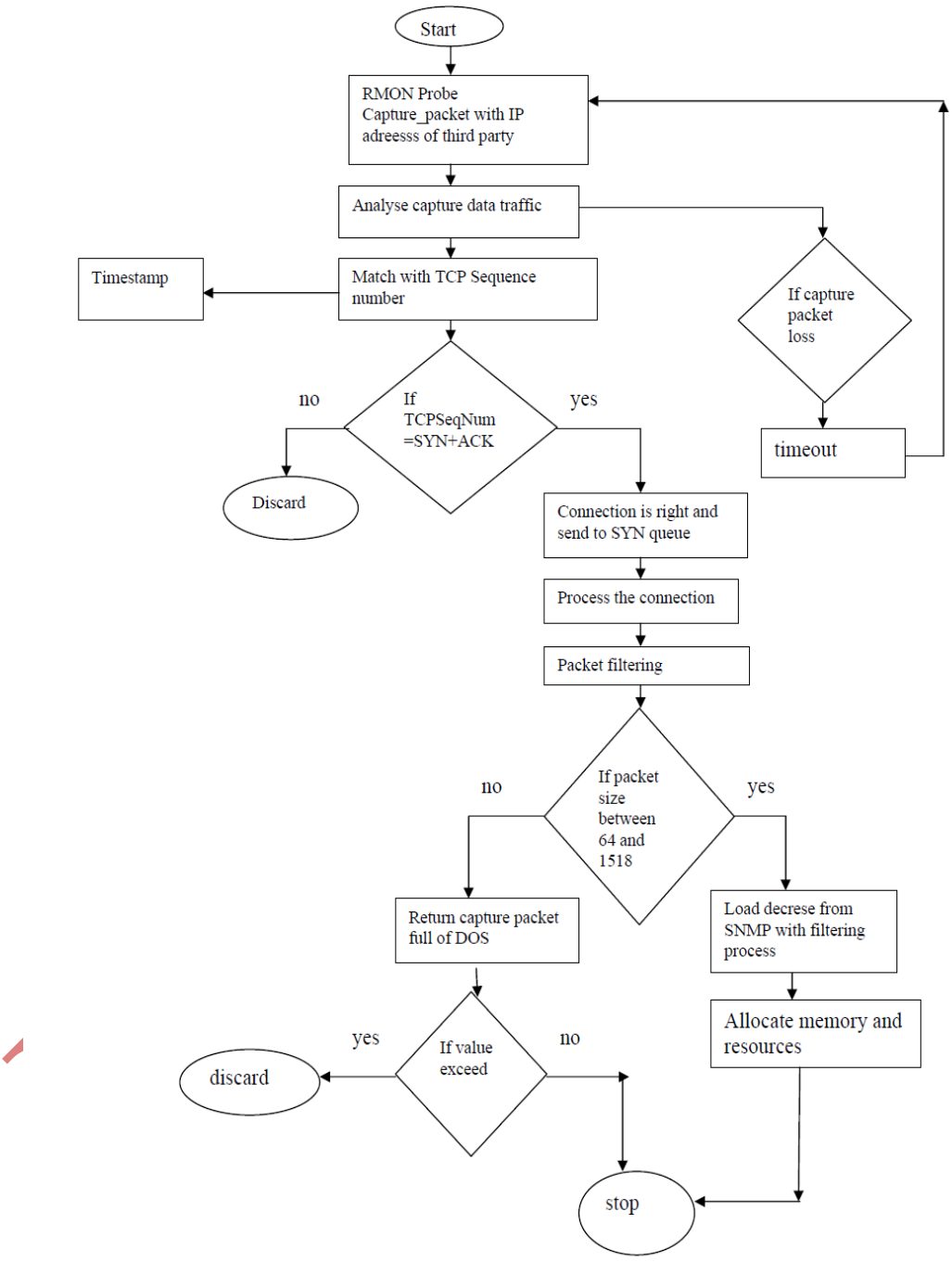

Figure 5:Detection and Prevention of SYN Flood DOS attack

IV CONCLUSION

In this proposed work, use the TCP Sequence number, SNMP, SYN cookies, Timestamp, Time Out, three-way handshake.SYN Flood DOS attack send high-rate data in to a small-small packets. So because of this these packets take large number of resources and consume bandwidth. So the solution of this problem is SNMP (Simple Network Management Protocol).SNMP sends small amount of load on the network with the filtering process which filter all the packet and separate the packets with DOS

attack or without DOS attack. And discard the DOS traffic and send trap to management station. And prevention technique use in this paper is SYN cookies and TCP sequence number. TCP sequence number, the server will verify the server sequence number of the ACK packet to ensure the client is a legitimate user. In this paper, the using of Time Stamp is to prevent it from duplicate packet and compare it in SYN cookies. In this paper, the using of Time Stamp is to prevent it from duplicate packet and compare it in SYN cookies. The use of the Timestamp is providing the uniqness of the connection. In the previous paper the problem is the server can not retransmit the loss packet .so the solution is the Timeout, which retransmit the loss packets. Our system can be improved by moving the filtering functionality.

V FUTURE SCOPE

In this proposed work, "Detection and prevention of synchronization flood DOS attack by RMON/SNMP" provide effective detection and prevention mechanism by the use of the TimeOut, Timestamp, SYN cookies, RMON and SNMP. But there is still problem in this propose work is Synchronization flood attack still take CPU cycle attack and memory stealing attack.

REFERENCES

[1] Kyoungwon Suh and Thu D. Nguyen, "*A Practical Defense Against SYN Denial-of -Service Attacks"*,technical report DCS-TR-452.

[2] Raja Azrina Raja Othman,*"Understanding the Various Types of Denial of Service Attack"*, International Food Research journal 18: 137-148 ,2011.

[3] L. Ricciulli, P. Lincoln, and P. Kakkar, "*TCP SYN Flooding Defense"*,. In Proc. of Communication Networks and Distributed System Modeling and Simulation Conference, San Francisco, CA, Jan. 1999.

[4] Christoph L. Schuba and Ivan V.Krsul et al., "*Analysis of a Denial of Service Attack on TCP*",. in IEEE Symposium on Security and Privacy,Oakland, CA, May 1997.

[5] L.Kavisankar and C.Chellappan, "*CNoA: Challenging Number Approach for uncovering TCP SYN flooding using SYN spoofing attack*", International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.5, Sep 2011.

[6] Bharathi KrishnaKumar, P.Krishna Kumar "*Hop Count Based Packet Processing Approach to Counter DDoS Attacks*" International Conference on Recent Trends in Information,Telecommunication and Computing, 2010.

[7] Ramana Rao Kompella, Sumeet Singh, and George Varghese, "*On Scalable Attack Detection in the Network*" in international journal on IEEE/ACM Transaction on Networking, Vol. 15, No. 1, 2007, pp. 14-25.

[8] Mohd.KhairilSailan,Rosilah Hassan,Ahmed Patel "*A Comparative Review of IPv4 and IPv6 for Research Test Bed*" 2009 International Conference on Electrical Engineering and Informatics Selangor, Malaysia 2009.

[9] Guang Yang, Mario Gerla and M. Y. Sanadidi, "*Defense against Low-rate TCP-targeted Denial-of-Service Attacks*", ISCC 2004: 345-350.

[10] P. Karn and C. Patridge, "*Improving Round-Trip Time Estimates in Reliable Transport Protocols"*, In Proceedings of ACM SIGCOMM 1987, Aug. 1987.

[11] J. F. Kurose and K. W. Ross, "*Computer Networking: A Top-Down Approach Featuring the Internet"*, 2nd edition, Pearson Addison Wesley, Jul. 2002.

[12] Ahsan Habib, Mohamed M. Hefeeda, and Bharat K. Bhargava, "*Detecting Service Violations and DoS Attacks"*

[13] Sau Fan LEE, "*Analysis on Some Defences against SYN-Flood Based Denial-of-Service Attacks",*

[14] William W. Streilein, David J. Fried, Robert K. Cunningham, "*Detecting Flood-based Denial-of-Service Attacks with SNMP/RMON**",

[15] Waldbusser, S., *RFC 2819*. 2000, ftp://ftp.isi.edu/in-notes/rfc2819.txt.

[16] J. Case, M. Fedor, *RFC 1157*. 1990, ftp://ftp.isi.edu/in-notes/rfc1157.txt.

[17] .*Baystack380Features*,http://www.nortelnetworks.com/products/02/bstk/switches/aystack_380 /features.html.

[18] M. Allman and V. Paxson, "*On estimating end-to-end network path properties."*, In Proceedings of ACM SIGCOMM 1999, Cambridge, MA, Aug. »Sep., 1999.

[19] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni. "*Analysis of a denial of service attack on tcp"*. In *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 1997.

[20] Lawan A. Mohammed and Biju Issac, *"Detailed DoS Attacks in Wireless Networks and Countermeasures"*, Int. J. Ad Hoc and Ubiquitous Computing, Vol. 2, No. 1, 2006.