12-2009

Evaluation of Windows Servers Security Under ICMP and TCP Denial of Service Attacks

Hari Krishnea Vallalacheruvu
University of Texas-Pan American

EVALUATION OF WINDOWS SERVER SECURITY UNDER ICMP AND TCP DENIAL OF SERVICE ATTACKS

A Thesis

by

HARI KRISHNA VELLALACHERUVU

Submitted to the Graduate School of the
University of Texas-Pan American
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2009

Major Subject: Computer Network Security

EVALUATION OF WINDOWS SERVER SECURITY UNDER ICMP AND TCP

DENIAL OF SERVICE ATTACKS

A Thesis
by
HARI KRISHNA VELLALACHERUVU

Approved as to style and content by:

Dr.Sanjeev Kumar
Chair of Committee

Dr. Weidong Kuang
Committee Member

Dr. Jun Peng
Committee Member

December 2009

ABSTRACT

Vellalacheruvu, Hari Krishna, Evaluation of Windows Server Security under ICMP and TCP Denial of Service Attacks. Master of Science (MS), December, 2009, 115 pages, 50 figures, 4 tables, 80 references, 2 appendices.

Securing server from Distributed denial of service (DDoS) attacks is a challenging task for network operators. DDOS attacks are known to reduce the performance of web based applications and reduce the number of legitimate client connections. In this thesis, we evaluate performance of a Windows server 2003 under these attacks. In this thesis, we also evaluate and compare effectiveness of three different protection mechanisms, namely SYN Cache, SYN Cookie and SYN proxy protection methods, to protect against TCP SYN DDoS attacks. It is found that the SYN attack protection at the server is more effective at lower loads of SYN attack traffic, whereas the SYN cookies protection is more effective at higher loads compared to other methods.

DEDICATION

The completion of my master studies would not have been possible without the love and exceptional support of my family. Specially, my father Koteswara Rao and my mother Anjamma motivated and supported me by all means to accomplish this degree. Thank you for your love and patience.

ACKNOWLEDGMENTS

First and foremost I offer my sincerest gratitude to the chair of my thesis committee, Dr.Sanjeev Kumar, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. I thank Dr. Kuang Weidong and Dr. Peng Jun for their willingness to serve as the committee members. Work in this thesis is based upon the grant awarded to Dr. Kumar by the National Science Foundation (NSF) under Grant No. 0521585.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

In 21st century the use of Internet and the technology in the field of Internet services like Online Shopping, Information Data base and Data Sharing are advancing at a very fast pace. According to the World Fact book of Central Intelligence Agency, Internet users in United States are around 223 million in 2008 [1] Third place in the world. The total business IP traffic will reach 7 Exabyte's per month by 2011, While growing at a staggering 35% compound annual growth rate over 2007-2012 [2]. On the other hand, Internet attacks are also increasing with the advancement in Internet usage and newer technologies. Denial of service (DoS) attack is one of the most common forms of security exploitation after virus attacks. Any form of attack, which causes service denial to legitimate users, is called denial of service attack. The attackers can make use of different vulnerabilities like protocol misinterpretation, software flaws, memory leaks and many more to cause denial of service attacks. Some of the DoS attacks consume the available resources such as Bandwidth, Processing power and Memory of the victim, so that the victim has no more resources available to provide services to legitimate users. According to recent studies, TCP accounts for 95% of the total traffic volume, and 80% of the total number of flows in the Internet [3]. In recent years, Transmission Control Protocol based DDoS-attacks are reportedly witnessed more frequently. These attacks

are used by the hacker community to cause service denial to the clients of Web servers, DNS servers, FTP servers and e-mail servers. Mitigating DoS attacks require the development of robust and resilient network protocols and security solutions. Recent DoS attacks on July 4Th, 2009, on several web sites operated by major government agencies in United States & South Korea, including the departments of Homeland Security and Defense, Federal Trade Commission and the Federal Aviation Administration caused wide spread outage over the long weekend. North Korea was suspected for launch of the Cyber warfare.

1.1 Statement of the problem

Distributed denial of service attacks (DDoS) are a threat to the future of online services and the Internet itself [4]. In reference [5], they have tried to estimate how prevalent the denial of service attacks on the Internet today is. Even though Internet community made quite a significant effort to propose ways to detect [6-7], trace back the source [8-9] and defend [10-14] against distributed denial of service attacks, hackers community is still able to exploit the vulnerabilities and cause the denial of service. Hackers are targeting different types of vulnerabilities to launch denial of service attacks. Most of them are using flaws in the code of networking protocols like IP, ICMP & TCP and the lack of security robustness in these protocols. As mentioned earlier TCP based denial of service attacks are used frequently to cause denial of service.

Before deploying any application on the server and provide services to the users, it is always recommended to do proper testing on the base protocols configuration in the lab environment and estimate the security robustness of the server against most common network security attacks. Network operators are implementing various ways to protect the

servers from the denial of service attacks. Proper configuration of the firewall on the end server also plays a vital role in defending against denial of service attacks. We observed the windows server 2003 firewall behavior and ability to defend against ICMP and TCP denial of service attacks. The behavior of windows 2003 server operating system under ping based ICMP denial of service attack was discussed in chapter 2.

TCP SYN flood attack is the most common as well as perilous TCP denial of service attack. More details regarding this attack is explained in chapter 3.

Most of the Network level detection, trace back and defense techniques proposed were based on traffic filtering, traffic shaping and on some statistical flow of TCP packets in the network. It is also important to have a defensive mechanism at the end server or end router in case if the attacker is able to reach the end server. This thesis mainly evaluates three different defense mechanisms against TCP denial of service attacks. The three popular defense mechanisms used to protect against TCP SYN flood attack are SYN cache, SYN cookies and SYN Proxy based protection methods. The performance evaluations of each of the mentioned defense mechanisms are done by creating a real time traffic environment at the Network Research Lab, UTPA. The concept of SYN attack protection mechanism implemented in windows server 2003 and its performance was observed in real time traffic. Performance evaluation is conducted to evaluate SYN Cookies and SYN proxy protection mechanisms that are implemented in Juniper J4350 router. Internet Engineering Task Force (IETF) have not standardize these TCP SYN protection methods as of yet. The evaluation of SYN flood protection methods in this thesis will help the network operators to estimate the amount of risk that can be tolerable.

1.2 Protocol Background

1.2.1 Transmission Control Protocol

Transmission Control Protocol (TCP) layer provides connection-oriented, reliable and byte stream data transfer services to the application layer. Unlike Internet protocol (IP), TCP makes sure that every segment sent from the source is reached to the destination without any errors. Connection-oriented [15] means, before any two computers can send data to each other they should establish a connection between them. A stream of 8-bit bytes is exchanged between the applications through the TCP connection. TCP also provides flow control and congestion control. Flow control is the process of managing the rate of data transmission between two hosts to prevent a fast sender from outrunning the buffer of a slow receiver. TCP provides flow control by frequently updating the receiver window size to the sender. A network is said to be congested when too many packets try to access the same router's buffer on the course to destination, resulting in an amount of packets being dropped at the router. Various congestion control methods used in TCP are discussed in RFC 2581.

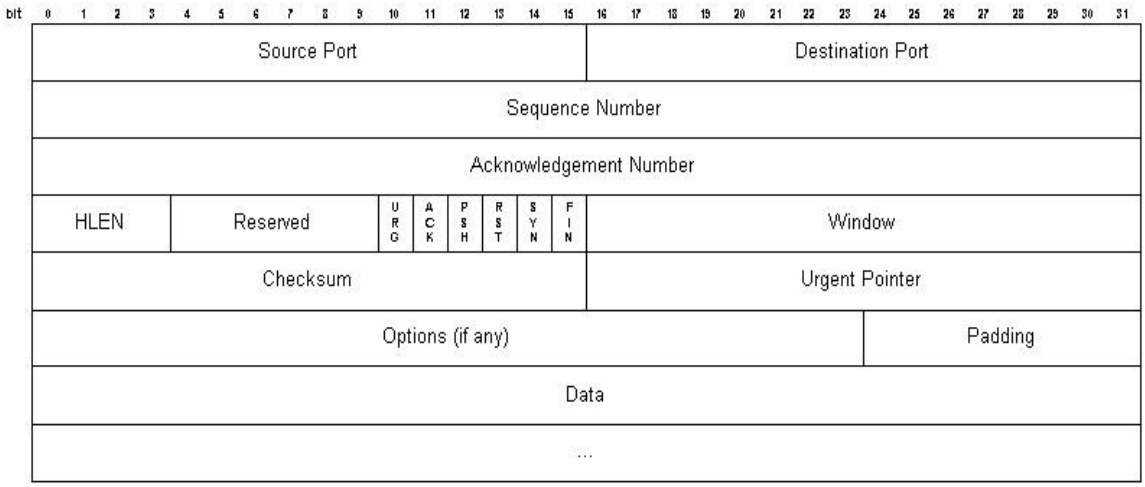

Fig 1.1 TCP header format

1.2.2 Three-Way Handshake

TCP uses three-way handshake to establish a connection between any two nodes. The client sends a SYN request with its sequence number to the server. When a SYN is received by server for a local TCP port where the connection is in the LISTEN state, then the state transitions to SYN-RECEIVED. The Transmission control block (TCB, a data structure to store all the state information for an individual connection) is initialized with information from the header fields of the received SYN segment. In second step the server respond with an ACK to received SYN and it will also send its own sequence number (SYN) to the client. In the last step, the client responds with final ACK segment. After the last ACK is received by the server, connection state changes from SYN_RECEIVED to ESTABLISH state. The real data transfer between the client and the server is initiated only after the three-way handshake is complete.

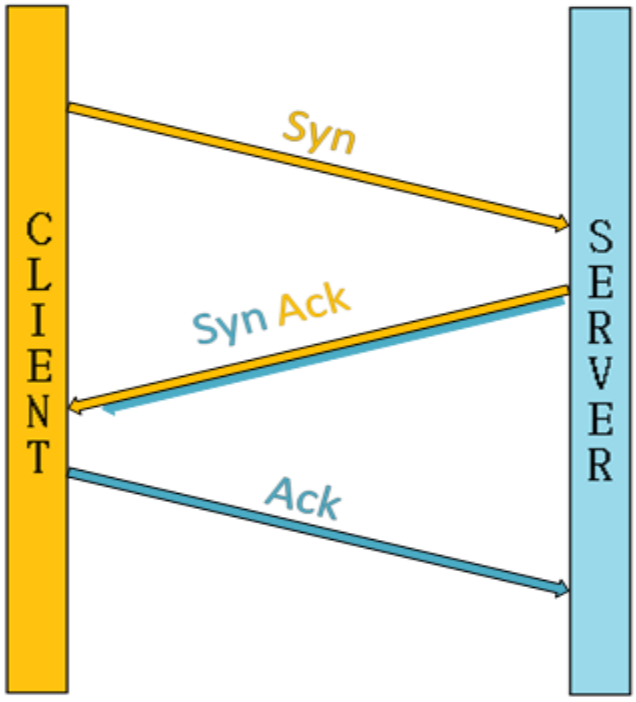

Fig 1.2 TCP Three-Way handshake

1.2.3 TCP Data Structures

For any TCP connections, under BSD style network code, there are three memory structures that need to be allocated by both the end-points.

Socket

The socket structure holds information related to the local end of the communication link like protocol used, state information, addressing information, connection queues, buffers and flags.

IP Control block structure (IPB)

TCP uses Internet Protocol control block structure at the transport layer to hold information such as TCP state information, IP address information, port numbers, IP header prototype and options, and a pointer to the routing table entry for the destination address.

TCP Control block structure (TCB)

TCP control block structure contains TCP specific information such as timer information, sequence number information, flow control status, and out of band data. The combined size of these data structures for a single TCP connection may typically exceed 280 bytes [16].

Different operating system may use different approaches to create data structures but the above information gives an idea about the TCP data structures. More details about Transmission control block size (TCB) and the state information stored within the TCB are included in chapter 3.

1.2.4 TCP Connection Termination

While it takes only three TCP segments to establish a connection, it takes four to terminate a connection. This is caused by TCP's half close. Since TCP is a full duplex (that is, Data can be flowing in each direction independent of the other direction), each direction must be shutdown independently. Either end can send FIN when it is done sending data. When TCP receives a FIN, it must notify the application that the other end has terminated that direction of data flow [17].

The receipt of a FIN only means that there will be no more data flowing in that direction. TCP can still send data after receiving a FIN. The normal scenario of TCP connection termination and the connection states of client and server are shown in the figure 1.3. The client normally initiates connections, with the first SYN going from the Client to the server. Either end can actively close the connection. Often, it is the client that determines when the connection should be terminated, since the client processes are often driven by an interactive user, who enters command like quit to terminate the connection.

Once after the three way-hand shake is complete the connection states of the client and server are in ESTABLISHED state and the data flow in both the directions take place. Suppose that the application on client side is done transmitting data to the server and it no longer need a TCP connection. The application using TCP signals that the connection is no longer needed. The client TCP sends a segment with FIN bit set and transitions to FIN-WAIT-1 state to request that the connection be closed. The server receives the client's FIN and it sends an ACK to acknowledge the FIN, the connection state changes to CLOSE-WAIT. When this ACK is received by the client the state

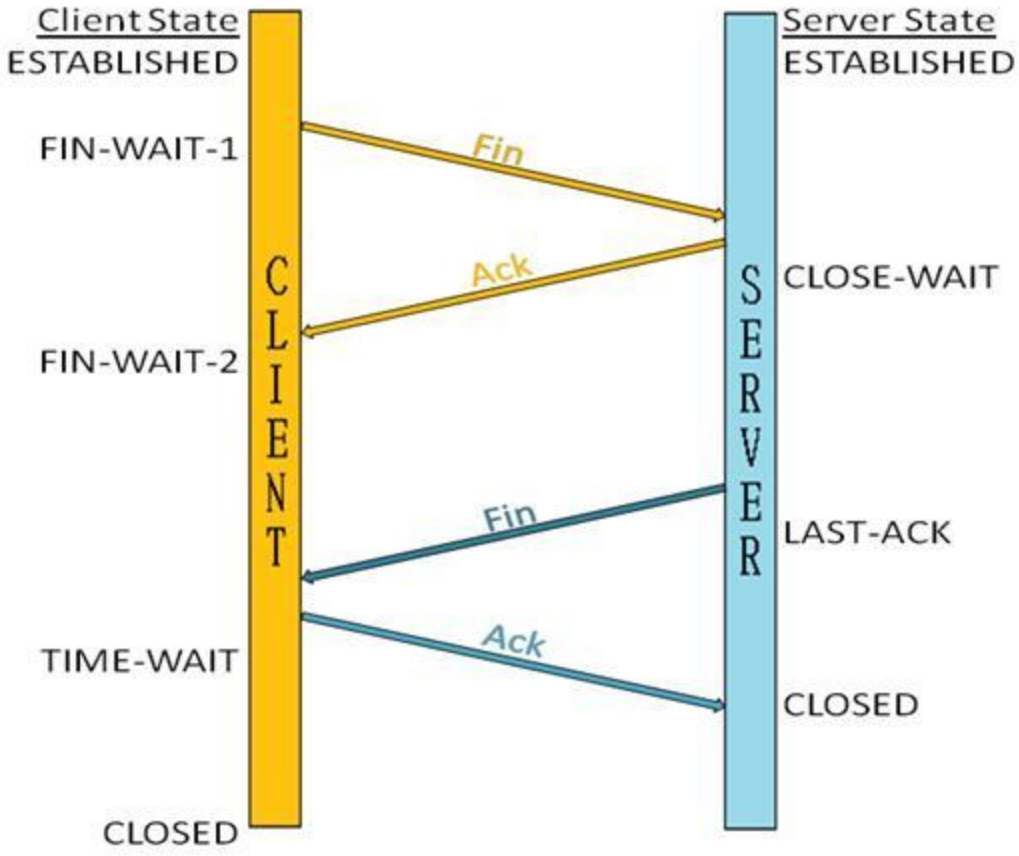

Fig 1.3 TCP Connection Termination

changes to FIN-WAIT-2 as shown in the fig 1.3. At this point, the data flow from client to server is closed but the server may still send data to the client. When the application on the server is done transmitting data, it also issue connection close signal to the TCP layer and TCP layer send a segment with FIN bit set. The client acknowledges the servers FIN. When the ACK is received by the server, the TCP connection is closed on the server. However, the client waits 2 * Maximum Segment Life (MSL) time after sending the ACK to server FIN (TIME-WAIT) to go to CLOSE state. This is because in case if the final ACK is lost on the way to server, the client needs to retransmit the final ACK.

1.2.5 TCP Connection States

Fig 1.4 shows the TCP connection states transitions, together with the causing events and resulting actions, but addresses neither error conditions nor actions that are not connected with state changes [15].

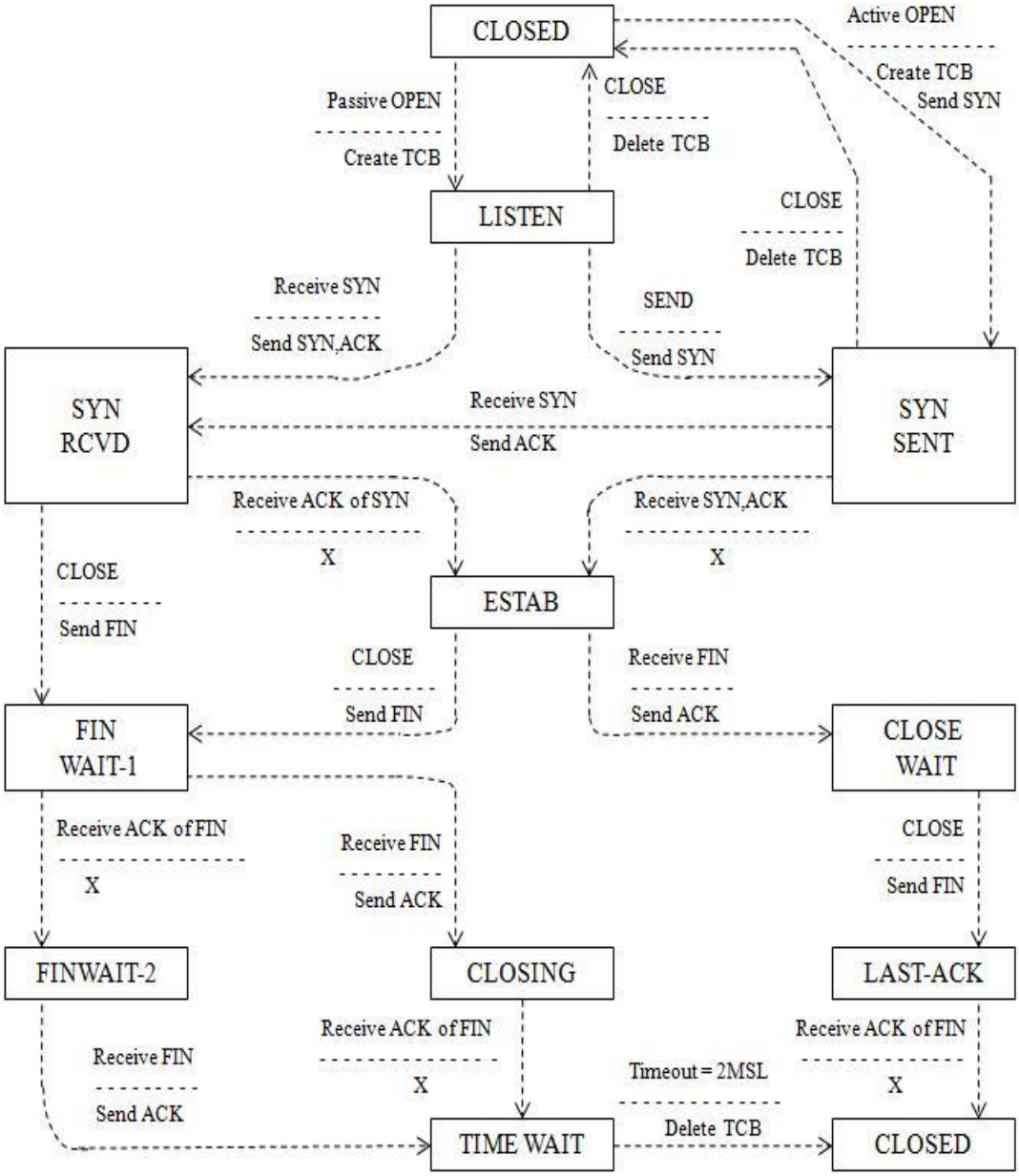

Fig 1.4 TCP Connection State Diagram

1.2.6 Internet Control Message Protocol

Internet Control Message Protocol (ICMP) is a set of messages that provides services that are not part of IP. ICMP, as defined in the RFC 792 [18], is used when a gateway or destination want to communicate with the source, for example to report problems like data delivery error, IP header problem, Connectivity problem, Address mask discovery and diagnostics. The purpose of these control messages is to provide feedback about problems in the communication environment. Different types of ICMP messages are defined and each ICMP message type has its own purposes. For example when a routing or delivery error occurs, a router or the destination discards the offending datagram and attempts to report the error by sending an ICMP destination unreachable message to the source IP address of the offending packet. ICMP echo request and echo reply messages are used to check whether the given host is reachable or not across the network. When we send echo request message received by any host, it responds with echo reply message. This feature is provided by ping utility. Ping measures the round trip time and records any packet loss. Ping is also used for self-test of Network interface card (NIC). The echo request /reply messages are used to provide diagnostics features like Ping, Traceroute and PathPing Utility in windows server 2003. According to RFC 792, every system must send an ICMP reply message for every request messages. But now a day's many network administrators are restricting or even blocking the ICMP messages flow in their networks. The reason behind this is to secure their network from ICMP based attacks, even though they are unable to use some of great features mentioned.

1.2.7 Hypertext Transfer Protocol

Hypertext transfer protocol (HTTP) is an application level protocol used for communications and data transmission between client computers and HTTP server also referred as Web servers. The first version of protocol was standardized by IETF in RFC 1945 and later versions are documented in RFC 2068 and RFC 2616. HTTP is a request/response protocol. A client wishing to receive a resource from an HTTP server issues a request message containing a request method, Uniform Resource Locator (URL), protocol version ID, and resource specific information. Even though most of the people are familiar with the naming conventions of HTTP, we would like to introduce some of them to understand this thesis.

Client: Any program that establishes a connection to an HTTP server to issue requests. Typically, this is a web browser such as Internet Explorer, Mozilla or Google Chrome.

Server: A process that accepts HTTP requests for connections from the client programs, and provides response data [19].

Message: The basic unit of communication between a client and a server. Messages are usually sent as a part of a TCP connection between a short lived TCP ports on the client to TCP port 80 (In general) on the server.

Request: A message from the client to the server that requests a resource. Most HTTP interactions involve a client sending a GET request message to the server.

Response: A message from the server to the client that returns information initiated by a request message.

1.3 Traffic Flow between a Client and Web Server

Any computer must have a valid IP address, Network Mask, Default Gateway and Domain Name Server (DNS) IP address to communicate with any host in another network as shown in the figure 1.4. First thing, we are going to do to request a web page

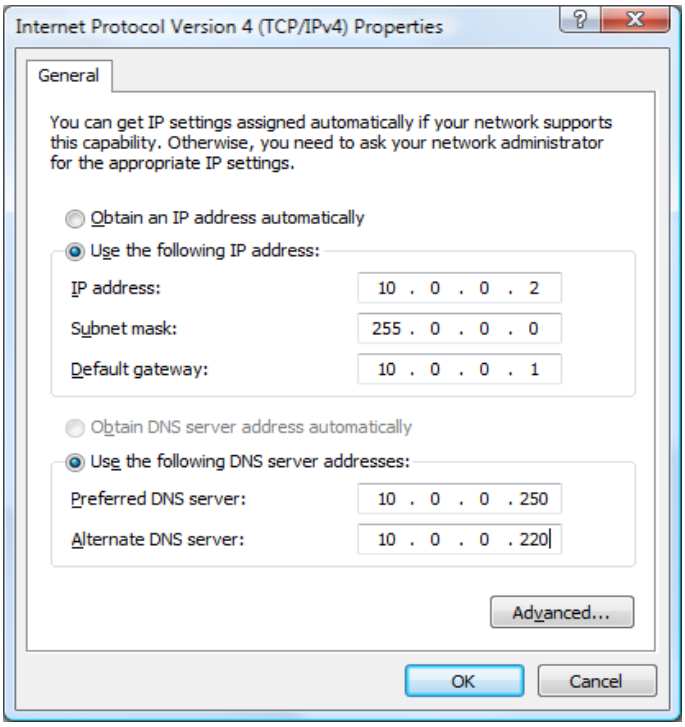

Fig 1.5 Internet Protocol (TCP/IPv4) Properties

from the server is to type in the URL in the web browser. Internet cannot identify machines with names and only way to do so is with their IP addresses. The browsers have to find the IP address of the web server with the help of DNS server. Since the DNS server is in the same network as of client for the above example, the client needs to know the hardware address of the DNS server to communicate. Since Ethernet support broadcast, the client will broadcast an Address resolution protocol (ARP) request in the local network requesting the MAC (Media Accesses Control) address of the host with IP addresses 10.0.0.250. The DNS server will send the ARP reply message, which includes

its hardware address to the client IP & MAC address in ARP request message. Then the client computer will send a request for the IP address of the corresponding URL entered by the user to the DNS server and the DNS server respond with the IP address of the web server. After receiving the IP address of the web server, the client will check whether the web server is in the same local network or outside the local network. In this particular

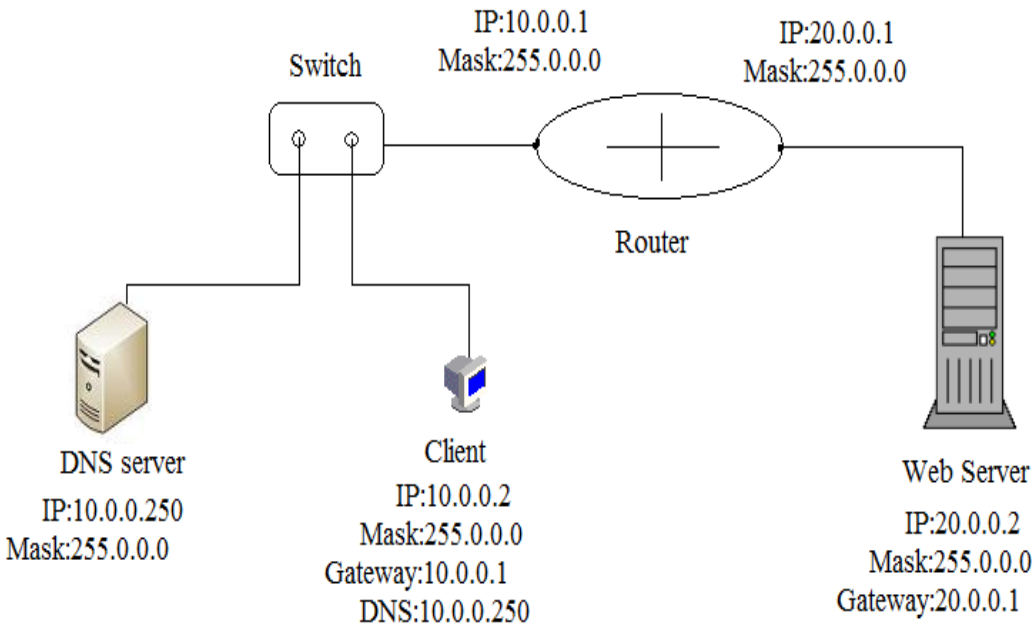

Fig 1.6 Sample Network

Case the web server is located in a different network as shown in the figure 1.5. Therefore, the client has to forward the data to the gateway in order to reach the destination web server. The client request the gateway hardware address as it does for the DNS server. After receiving the MAC address of the gateway, the client will send a TCP SYN request segment with destination IP address as the Web server IP at layer 3. However, when it comes to Ethernet layer header the destination MAC address is the Gateway MAC address. When the packet reached the router interface, the router

recognizes the destination network is on the other port based on routing table and forwards the packet to port with IP 20.0.0.1. The router port will find the MAC address of web server and forward the SYN packet by changing the MAC destination address field to Web server Mac address. Since the MAC addresses and the corresponding IP addresses are saved in the hosts ARP table for some time after resolving for the first time communication. Therefore, the MAC address resolving is not required on the way of web server reply back to client. Off course, the client has to complete the three-way handshake (SYN, SYN-ACK, and ACK) before it sends the GET request message to the server for a web page and the server respond with the requested data [20].

From the above discussion, it is clear that the destination MAC addresses are only significant in local network and it keeps on changing as the packet pass-through different networks. However, the destination IP address will be the same until the packet reaches the destination. Intermediate routers will use only network ID part of the IP address to route packets and the end router is the only one, which uses the host ID part of the IP address to reach final destination.

1.4 Denial of Service Attacks

Denial of service (DoS) has long been an open security problem of the Internet and most dangerous one due to the fact that they threaten not only technical aspects of trade but also give rise to financial expenses. The goal of DoS attack is to completely tie up certain resources like server clusters, memory, server or router CPU cycles, and network bandwidth, so that legitimate users are not able to access a server. A successful DoS attack achieves two objectives: Overpowering the victim and concealing the offender's identity. In recent years, many companies and government agencies have been affected by this kind of attack, e.g. eBay, Amazon, Buy.com, Department of Homeland Security and Defense, Federal Trade Commission and Federal Aviation Administration [21, 22]. The increasing menace of these attacks parallels the difficulty in detecting, preventing and neutralizing their effects.

A common strategy used by an attacker to cause DoS on a given target is to flood it with a continuous stream of packets that exhausts its connectivity. DoS attacks that use this kind of strategy are called Brute force attacks. There are two types of DoS attack based on the intensity. In high-rate attack, each malicious client violently sends data to the server. Even when the number of malicious clients is lesser than the number of usual clients, the attack traffic can still overwhelm the legitimate traffic. In low rate attack, the number of malicious clients is far greater than the number of normal clients. The comprehensive attack traffic is irresistible even when each malicious client sends data at a low rate, making it impossible to differentiate from a normal client [23].

1.4.1 Classification of denial of service

Classification of denial of service attacks can be done based on different criteria. DoS attacks can be simply classified into five categories based on the attacked protocol level as shown in the figure.

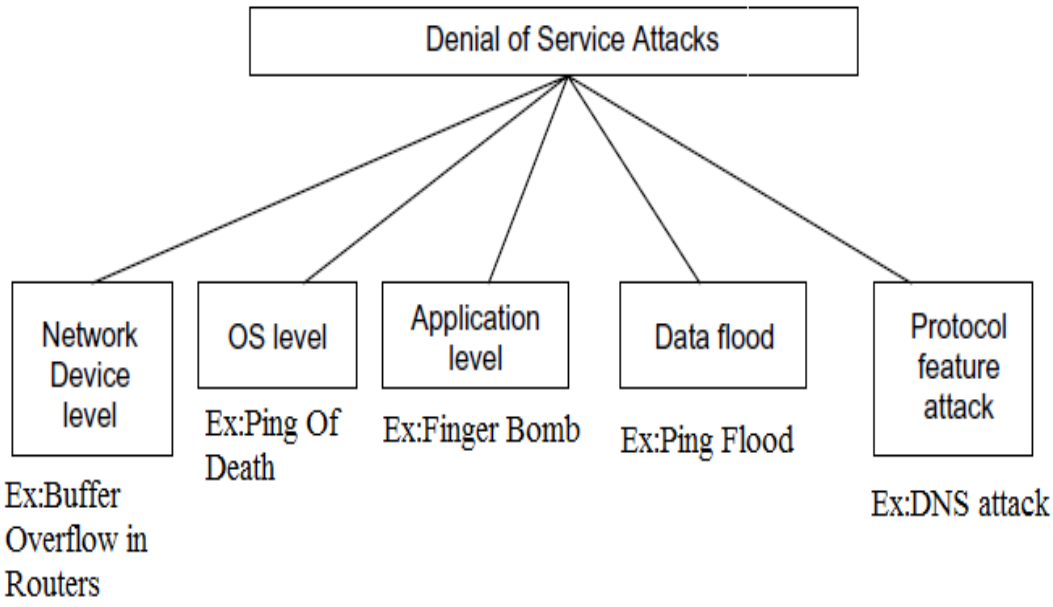

Fig 1.7 Classification of Denial of Service Attacks [24]

The flood attacks based on the TCP/IP protocols stack at the end system fall under data flood attacks. In chapter 2, ICMP based denial of service attacks (ping flood) and windows server 2003 behavior under these kinds of attack has been discussed. Flood based TCP SYN attacks are explained in detail and experimental evaluation of end system protection method called SYN Cache is done in chapter 3. Both these attacks fall under data flood attacks. Chapter 4 deals with the SYN flood protection methods in the network device namely Router that falls under network device level attacks.

1.5 Distributed denial of service Attacks

In a Distributed denial of service attack (DDoS), multiple attack sources are used to launch attack against one victim computer, which increases the resources for the offense while making it complicate to trace back the attacker. The attacker compromises a number of slaves and installs flooding servers on them, later contacting the set of servers to combine their transmission power in an orchestrated flooding attack. The use of a large number of slaves both augments the power of the attack and complicates protecting against it.

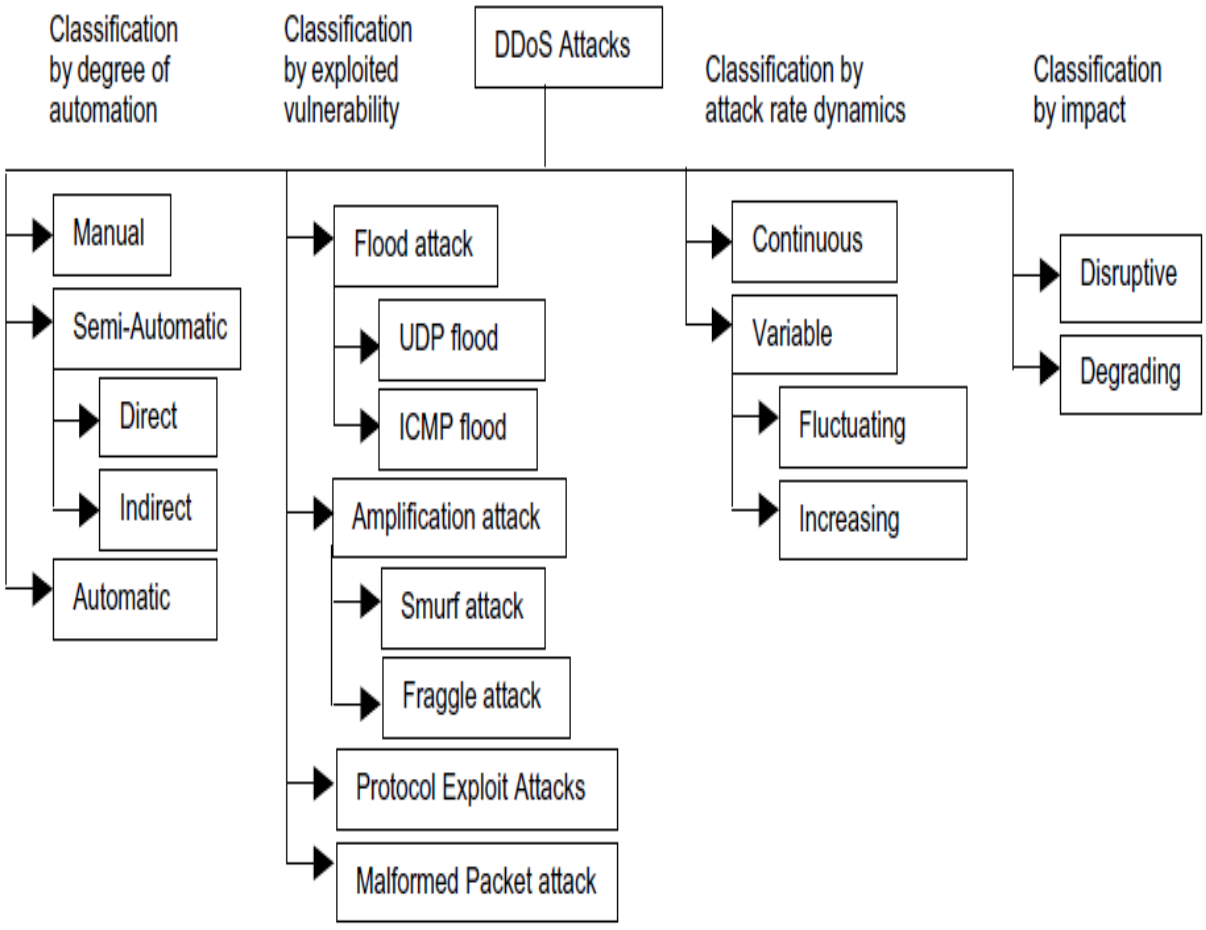

Fig 1.8 Classification DDoS attacks [25]

Many efforts have also been made, in parallel with the evolution of DoS attack, in the field of prevention and detection in networking security. Some of the approaches that have been proposed to defend against DDoS attack include egress filtering [26], ingress [27] filtering, disabling unused services [28], anomaly detection using intrusion prevention system (IPS) [29], trace back and honey pots [30] etc.

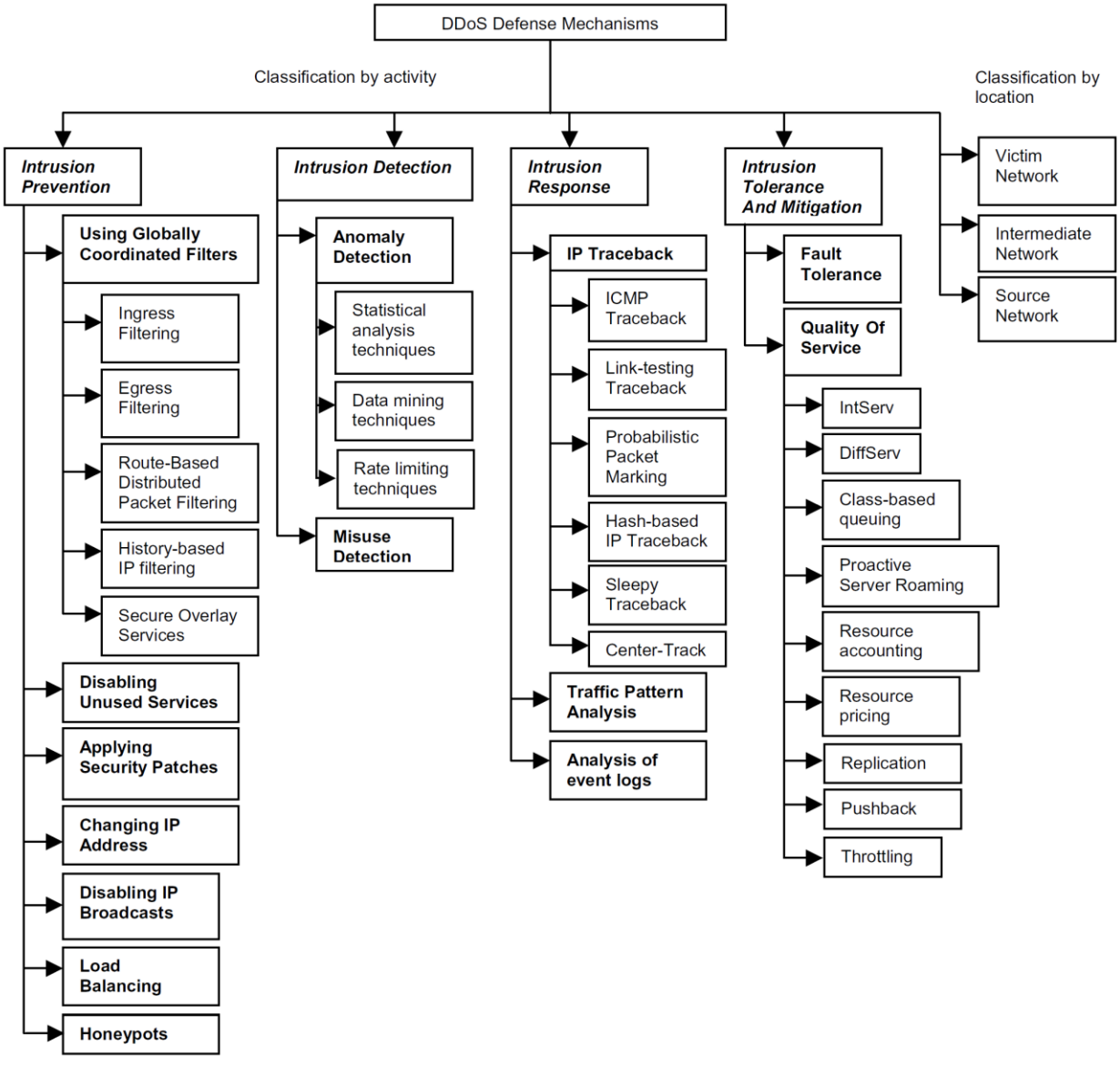

Fig 1.9 Classification of DDoS defense mechanisms [25]

1.6 Introduction to Equipment and Their Specifications

1.6.1 Windows Server 2003

Microsoft Windows server 2003 R2 Enterprise Edition was designed as premier platform for business-critical applications. As claimed by Microsoft this server operating system was designed in a way to increase the reliability, Scalability, Security and Manageability of enterprise applications. In the experimental setup used in the network research lab at UTPA, Windows server 2003 enterprise edition was used as the operating system (OS). Many small businesses to large enterprises are using this edition of operating system to provide services to their clients, is the reason behind using this OS. The server software and hardware specifications are as follows

Microsoft Windows server 2003 R2

Enterprise Edition

Service Pack 2

Intel® Xeon® CPU E5345 @ 2.33 GHz

8GB of RAM

Broadcom BCM5708C NetXtreme II GigE NIC Card

Throughout the experimentation, we used gigabit Ethernet to make sure that packets are not being dropped because of the bandwidth limitation. The Intel processor mentioned above is a very powerful quad core processor. 8GB of RAM available is good enough for the experiments conducted in this thesis. We make sure that the server never ran out of memory at any point of time.

1.6.2 Juniper Gigabit Router

Juniper J4350 is a high-end router, which supports up to one gigabit per second Performance. As claimed by Juniper networks the routing architecture in these routers provides the solid, reliable, high-performance foundation upon which today's real-time, critical networking applications can be delivered. Juniper J4350 router was one of the four best channel product awards won by Juniper networks from business solutions magazine. Along with high performance routing, this router also provides denial of service attack protection mechanisms, Firewall protection, Traffic filtering and traffic shaping as well. The maximum performance and capacity of J4350 router with 1GB of RAM are as follows

1. Firewall Performance is 1.6Gbps with large packets
2. When Firewall and Routing are enabled at the same time, the router can process 225,000 packets per second considering each packet size as 64 Bytes.
3. 128,000 Maximum concurrent sessions can be supported with 1 GB DRAM
4. Router can support 10,000 new sessions per second

Even though the J4350 ROUTER has the above-mentioned performance limitations this router is capable of providing most popular SYN flood protection mechanism namely SYN cookies and SYN Proxy. The maximum number of security policies that we can define is 5,192. There are higher performance routers available from Juniper. However, the one used in the lab is good enough for the experiments that were carried out in the Network Research Lab at UTPA.

CHAPTER II

ICMP DENIAL OF SERVICE

ICMP based distributed denial of service (DDoS) attack is launched via sending large number of request packets by the compromised systems to the victim computer. Ping is a utility, which uses Internet control message protocol (ICMP) echo request/reply messages. Ping is used for diagnostic purpose. Ping attack is a well known ICMP based Denial of service attack. Flooding of ICMP request messages to the victim computer most often with spoofed IP source addresses is called Ping attack. During Ping attack, the victim computer tries to process the directed echo requests, which require significant resources of the victim computer like Processor power, Memory and Bandwidth. Most of the operating systems and servers available in the market released software patches to reduce the adverse effect of ping attack. However, according to our experimental evaluation done in the lab environment these installed mechanisms are not good enough to completely escape the adverse effects of ping attack. It is a common misunderstanding by the end users that as long as they block ICMP messages they are safe from ping attack. But it's not true even though we block ICMP messages by using the server installed firewall Despite the presence of firewall on the server, which blocked all the ICMP messages, it was found that the flood of ping packets under the DDoS attack caused the server to be exhausted even though it dropped all the ping attack packets. In

this paper, we present results of our measurements of ping attack on the performance of windows based server performance get a clear picture of ping attack effects on windows based operating systems and servers performance.

2.1 Introduction

Internet control message protocol [18] (ICMP) is meant for diagnosis purposes which works on top of IP layer. Destination computer respond with an ICMP reply (echo reply) message for every ICMP request message (echo request) send by the source. In this way ICMP request and ICMP reply messages are used by Ping to verify the connection between the source and the destination.

Flooding of echo request messages with spoofed source IP address to the victim causes denial of service, which is called ping based denial of service attack. Ping attack is simple to launch and is good enough to bring the whole Internetwork down by attacking the root DNS servers [4]. The victim computer try to respond with echo reply message for every echo request received which require significant Processing, Storage and Bandwidth.

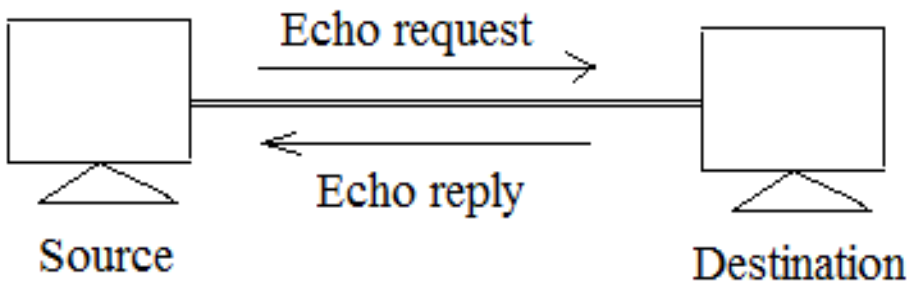

Fig 2.1 ICMP Echo Request/ reply

You can imagine how much processing power the victim computer requires to process ping requests (echo request) received at full bandwidth on a 100Mbps line (1,48,800 echo requests/sec).

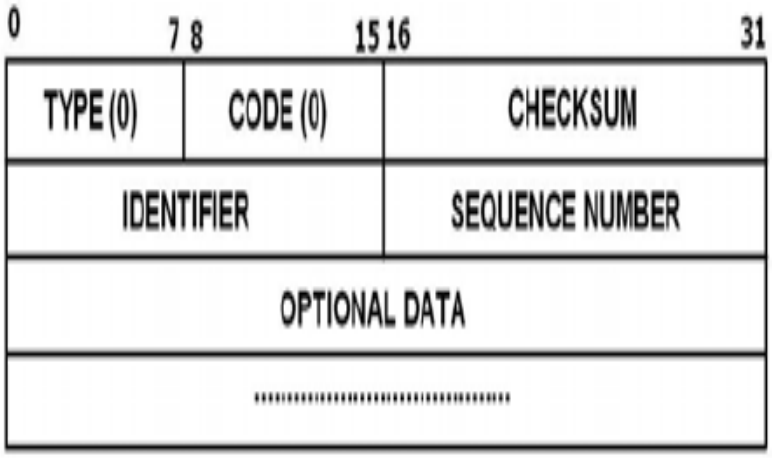

Fig 2.2 ICMP Echo Request/Reply Message Format [31]

Source generate ICMP echo request message by placing type field ‘8’ and code field as ‘0’. To generate ICMP reply message the destination computer replace the type field with ‘0’, alter the source & destination addresses and keep the same Identifier and calculate the Checksum [31]. The identifier and sequence number may be used by the echo sender to aid in matching the replies with the echo requests. Some of the commonly used distributive DDoS detection methods proposed in [32 - 36]. Almost all the available defense methods depend on the some selected nodes in the network to detect the attack. Therefore, the defense of DDoS attack in the network using these methods depends significantly on end systems capability of withstanding to DDoS attacks [37]. In [38] they try to reduce the dependency of detection algorithm on end systems. Different

operating systems implemented different methods to reduce the effect of ping attack. During ping attack on Windows server 2003 enterprise edition with SP2 [39] deploying Intel Xeon 2.4 GHz processor (Quad core) [40] at UTPA. Windows firewall [41] in the server is itself enhancing the resources consumed by the ping attack significantly when ICMP request messages are allowed. Smurf is also another type of ping based DDoS attack [42, 43].

2.2 Experimental setup

The testing of security systems is always a challenging task. The testing of windows server 2003 R2 enterprise edition with sp2 performance was measured by creating a real time real-time environment in networking lab at University of Texas-Pan American (UTPA). Testing the server against ICMP based DDoS requires sending a large number of ICMP ping requests to the server from a large number of different hosts as shown in the figure 2.3.

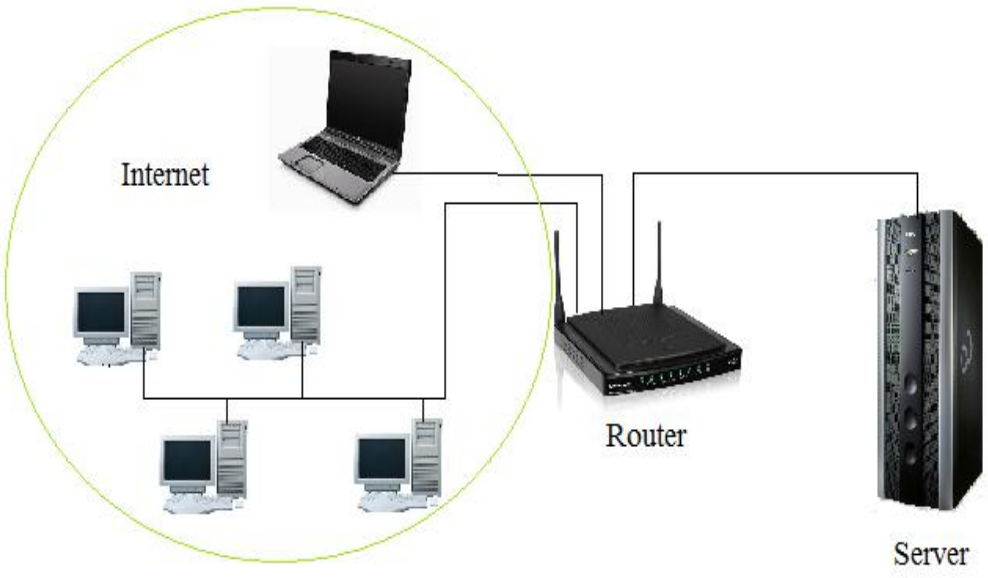

Fig 2.3 Testing Topology

The systems in the ring represent a network of attacking computers. The ping requests are sent starting from 10Mbps to 100Mbps line speed, at an increment of 10Mbps. Each load is sent for 10 minutes and measurements are recorded every second for a total of 10 minutes. Hence, the value shown in the graphs in this paper at any particular load is an average of 600 readings.

2.3 Performance Evaluation

We created a real time network environment at UTPA to test the windows server 2003 enterprise edition with sp2. The server was equipped with powerful Intel Xeon 2.4 GHz and 8 GB of RAM. We launched ping attack on the server under three different circumstances under different firewall configurations. We tried to analyze windows firewall effectiveness in protecting the server against ping based DDoS attacks. We measured the process exhaustion of the server under DDoS attacks under three different configurations.

Case-1:

The performance of the Xeon-processor based Windows server is measured without firewall under ping attack from different IP sources (i.e. Distributed Denial of Service Attack)

In this case, the Windows firewall is not activated (turned OFF). A flood of Ping type ICMP messages are sent to the victim Windows based server. The windows server continues to reply to incoming Ping ICMP messages. The incoming and outgoing ICMP messages are measured for the victim server. The SP2 capabilities built in the Windows operating system appears to detect for the Ping based DDoS attacks by monitoring the rate of incoming ICMP messages. If the rate of incoming ICMP messages exceeds 250

messages per second then all excess ICMP messages are simply dropped without further processing for corresponding response (Fig. 2.4). No reply is sent out from the Windows server for the incoming ICMP messages that exceed 250 messages/second.

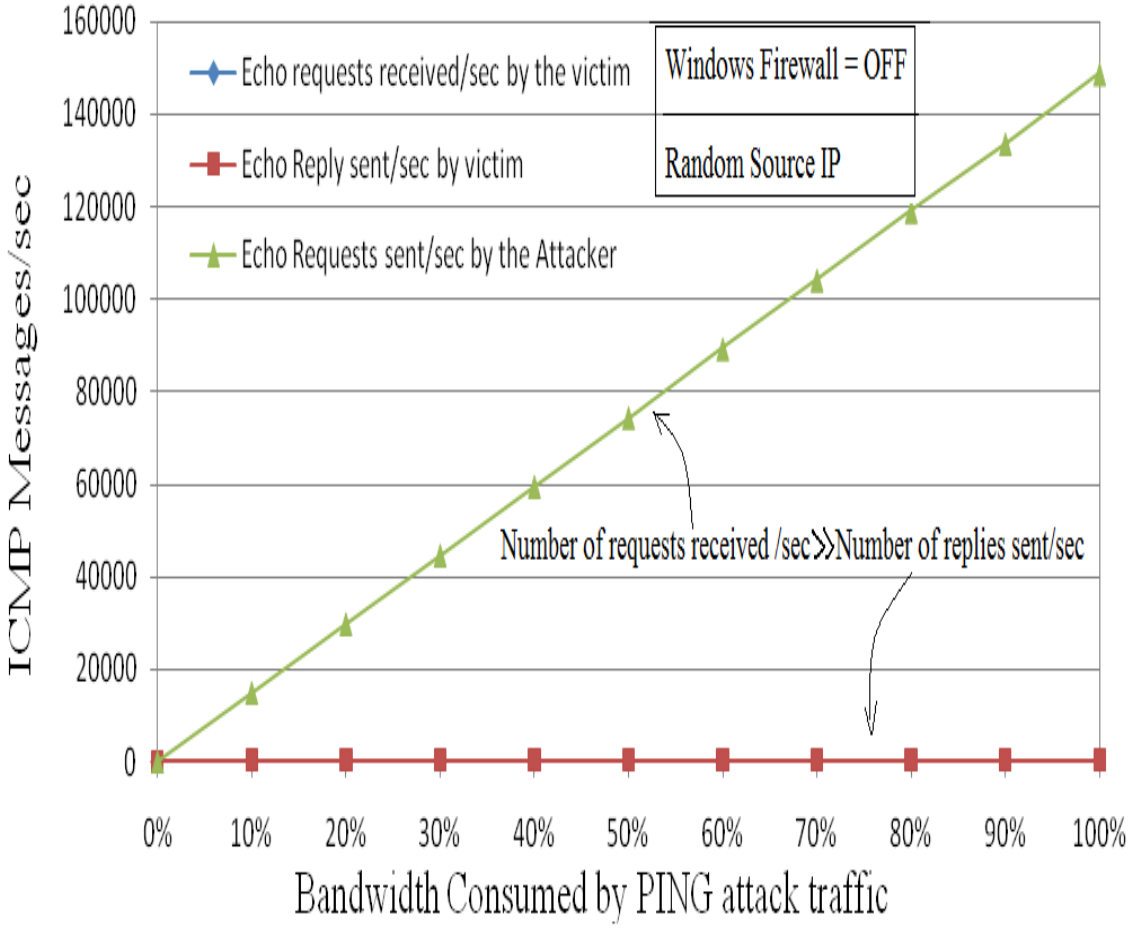

Fig 2.4 Traffic flow during ping attack for increasing traffic loads with the windows firewall OFF on the victim

In Fig. 2.4, the number of received ICMP Ping messages for the victim windows server is measured for different traffic loads starting from 10Mbps to 100Mbps of the attack traffic. It is observed that the received ICMP messages are much more than the number of responded ICMP Ping messages. For each load, irrespective of the number of

Ping ICMP messages received, only 250 messages are replied per second. This is the limit imposed by the SP2 capability that is built in to the Windows operating system. In this case, the firewall is OFF, still the SP2 capability of the Windows server can detect the Ping based DDoS attacks and limit the response to the incoming ICMP messages to only 250 messages per second. Since the firewall is OFF, no further processing is done for each one of the incoming ICMP messages. Firewall being turned OFF, firewall

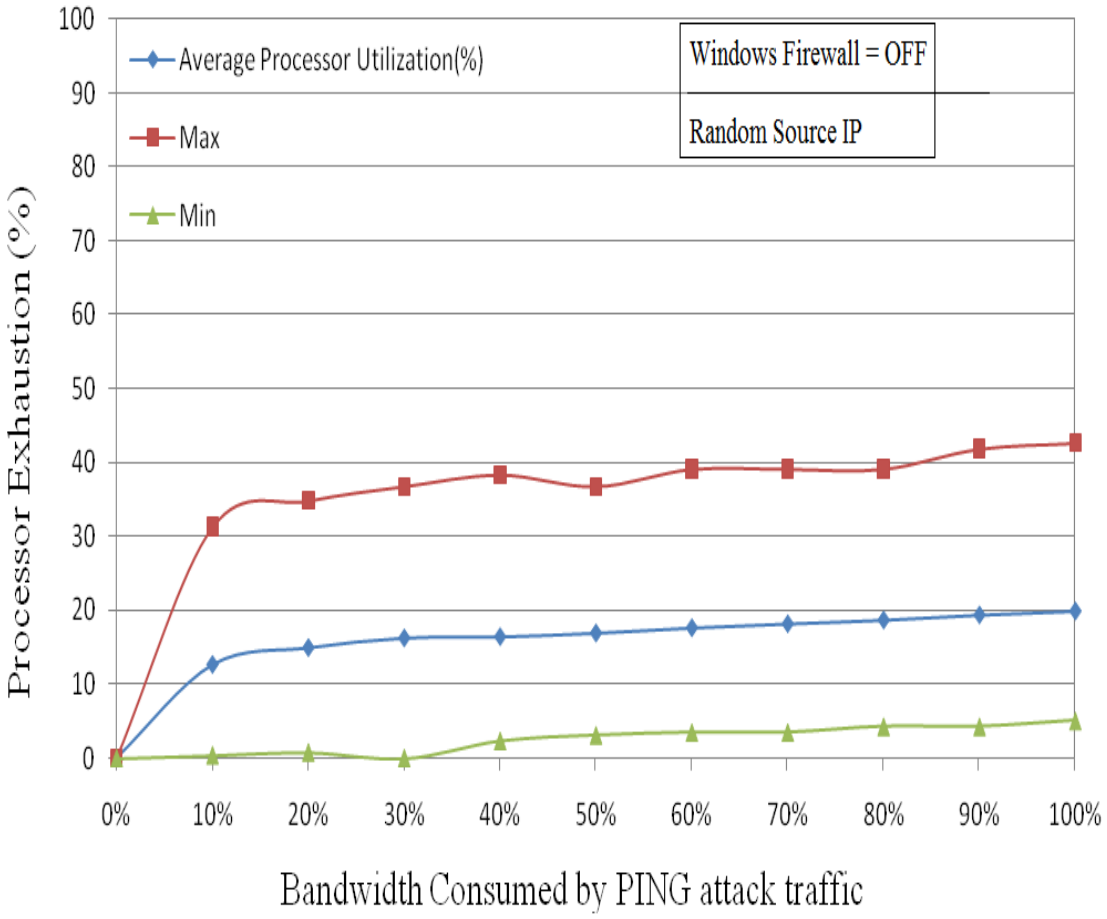

Fig 2.5 Processor Exhaustion of Victim during ping attack for increasing traffic loads with the windows firewall OFF on the victim computer

capabilities are not utilized for each incoming messages, and the average processor exhaustion is limited to only 20% (Fig. 2.5). SP2 operations for the detection and dropping of the excess ICMP messages appear to be the main contributor to the observed 25% of the processor utilization of the Windows server. Under this configuration, the maximum Xeon-processor was found to be around 40% for maximum load of the attack traffic of 100Mbps (Fig. 2.5). In these experiments, no other user applications were running in the background.

Case-2:

The performance of the Xeon-processor based Windows server is measured when the firewall is activated and ICMP packets are allowed.

Windows firewall has an option, which allows the ICMP messages to be allowed in for firewall processing, or disallowed. In this case, the windows server is configured such that the windows firewall is turned ON and the ICMP messaged are allowed in by the firewall for processing. Under this configuration, when the ICMP messages are allowed in by the firewall then the firewall inspections are done for each incoming ICMP messages at the cost of considerable processor resource. In this case, also the ping based DDoS attack is launched from different sources with different source-IP addresses. The SP2 capabilities detect the ping attack by monitoring the rate of incoming ICMP messages and only responds to 250 messages per second (Fig. 2.6).

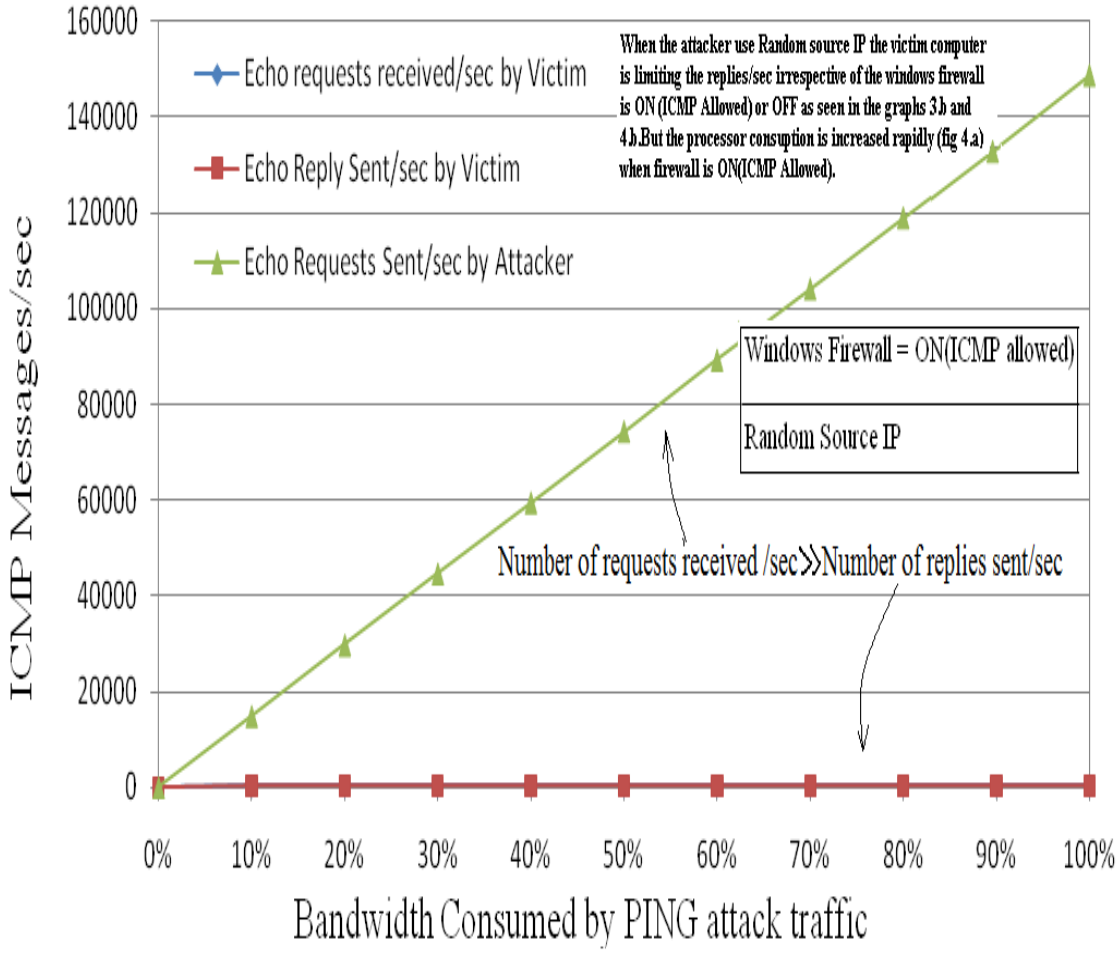

Fig 2.6 Traffic Flow during ping attack for increasing traffic loads with the windows firewall ON and Incoming ICMP requests allowed on the victim computer

Fig. 2.6 shows the number of Ping replies sent out to be much smaller than the number of the received Ping attack packets. Fig. 2.7 shows the corresponding processor exhaustion under DDoS Ping attacks for the given configuration for different attack loads. Since the ICMP messages are allowed in by the firewall in this configuration, the firewall inspection is done for each received Ping-ICMP messages at the cost of server's processor resource. As the attack traffic load is increased, more Ping messages are being processed by the firewall and the corresponding processor resource is measured to be

excessively consumed under such configuration for such attack (Fig. 2.7). For maximum attack load of 100Mbps, the powerful Xeon Quad-processor is measured to completely exhausted to close to 100%. In the lab, the powerful server was observed to be completely frozen under Ping attack with the given configuration.

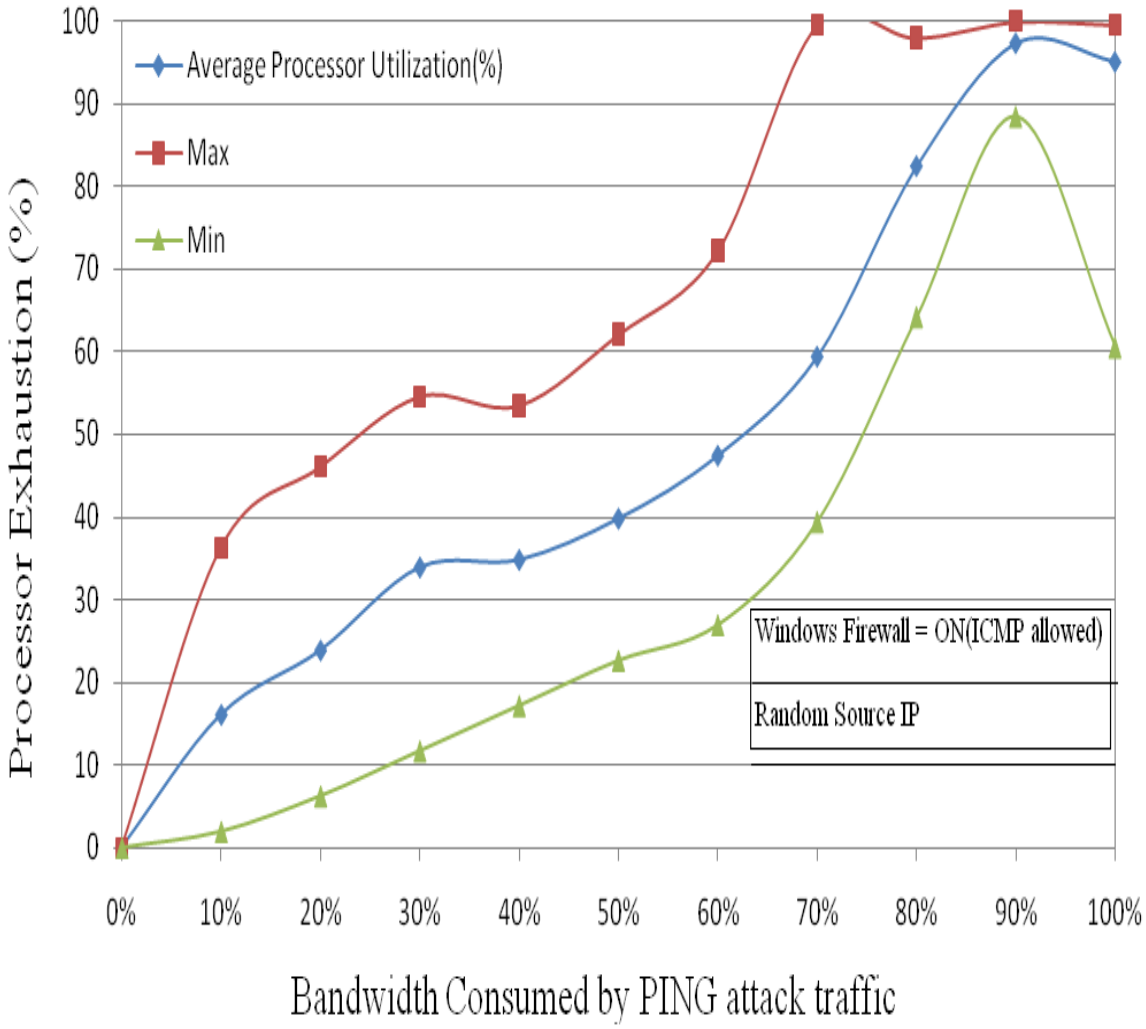

Fig 2.7 Processor Exhaustion of victim during ping attack for increasing traffic loads with the windows firewall ON and Incoming ICMP requests allowed on the victim computer

Case-3:

The performance of the Xeon-processor based Windows server is measured when the firewall is activated, and ICMP packets are not allowed (i.e. dropped) by the server firewall.

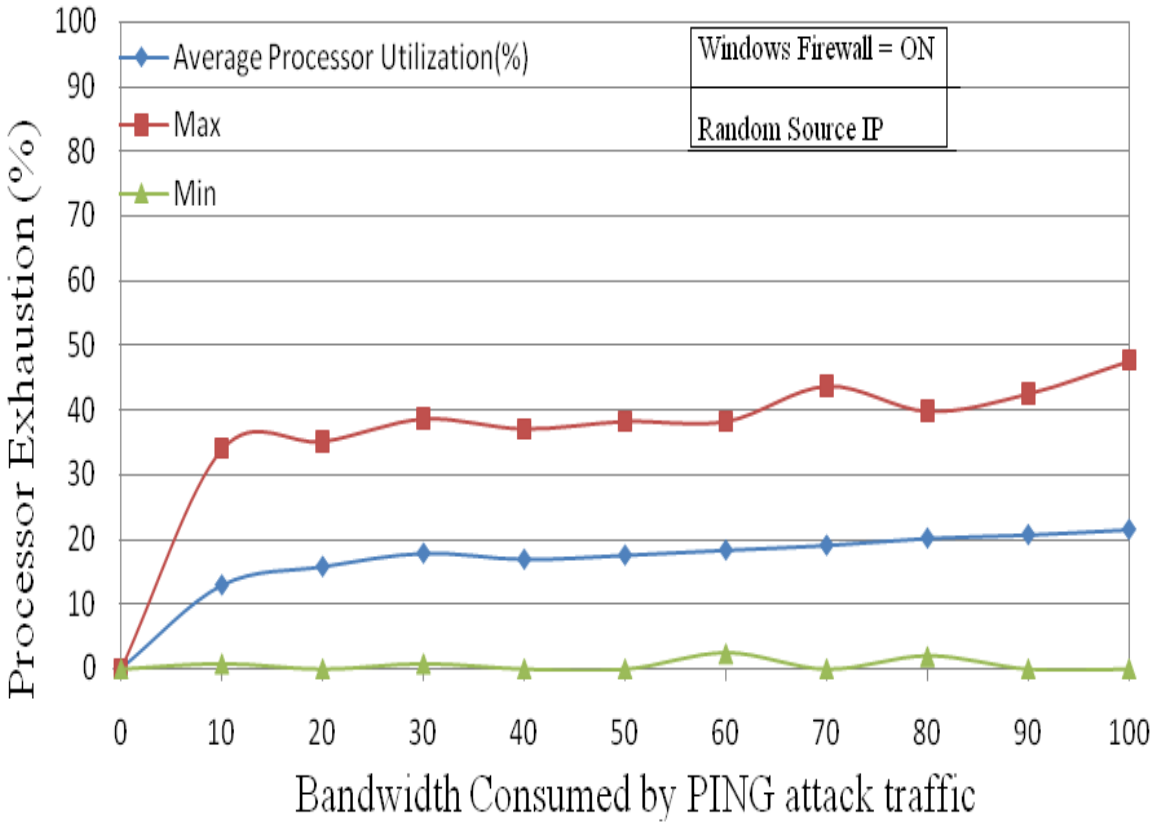

Fig 2.8 Processor Exhaustion of Victim during ping attack for increasing traffic loads with the windows firewall ON and Incoming ICMP requests not allowed on the victim

Under this configuration, the windows firewall was activated (turned ON) while disallowing the incoming ICMP messages. Under this configuration, the firewall inspection was not activated for each one of the incoming ICMP messages. As a result, significant processing resource was saved and it resulted into smaller consumption of the

processing power (Fig. 2.8) compared to the scenario in Case-2 (Fig. 2.7) when the ICMP messages were allowed in by the firewall. These measurements in these three cases provide us the extent of processor exhaustion under Ping based DDoS attacks on server that may be connected to the Internet via wire line or wireless connections. It is observed that just turning ON the firewall for the windows server is not enough to protect it from Ping-based DDoS attacks. It is important that if the firewall is turned ON then the ICMP messages should be explicitly disallowed for it to minimize the processor starvation due to the Ping based DDoS attacks. If not configured correctly, even a simple Ping based DDoS attacks can completely idle a powerful Xeon based Quad processor.

Denial of service attacks are increasing every day in the number of occurrences and sophistication. This is causing denial of legitimate services in the wire line and wireless Internet. Servers are implementing various different ways to ward off the Denial of Service attacks. In this paper, the most popular Windows server is evaluated under real Ping based Denial of Service attacks to understand its capabilities in protecting against such attacks. It is found that the SP2 capability of the Windows server can detect Distributed Denial of Service Attacks by measuring the rate of arrival of ICMP packets from different IP addresses. It is found that if the number of incoming Ping packets from different IP addresses exceeds 250 per second then it starts dropping the excessive packets and does not send any response back for the excess packets. It is found that there are different configurations possible for the server with windows firewall. The windows server firewall has two options available in regards to allowing or disallowing ICMP messages. It is found that even the powerful Xeon-processor based windows server suffers a catastrophic slowdown under a distributed Ping-based DDoS attack, if a

Windows server is configured such that the firewall is activated and ICMP messages are allowed. It is observed that just turning on the firewall in windows server alone does not mean that the windows server is protected from Ping based DDOS attacks. It is also learned from the experiments that it is important that if the windows based servers have their firewalls activated then it must also select the option where the ICMP messages are disallowed. Only under this configuration, the Windows based server manages to survive a Ping based DDoS attacks, despite losing a maximum of 25% of its processing capability in a fast Ethernet environment. If not configured correctly, even a simple Ping based DDoS attacks can completely idle a powerful Xeon based Quad processor.

CHAPTER III

TCP DENIAL OF SERVICE

When TCP/IP protocol suite was initially developed as a part of network research development by the United States Advanced Research Projects Agency (DARPA or ARPA) in 1970's [15], they are unaware of the security attacks. At that time the protocol suite designs was concern with appropriate communication and the scalability of the network. There was no proper framework to defend against security attacks in the initial design of protocol suite. As time progress TCP/IP gained more popularity than any other architecture. There is always been some hacker community who are trying to exploit security breaches of popular TCP/IP architecture. Whenever the hackers exploited the security breaches, the TCP/IP developer community tried to fix it by making some changes to the TCP/IP protocol suite. TCP/IP stack is still evolving to defend against security attacks. For example, recently Microsoft released a critical patch to TCP/IP on 8th September 2009[44]. This patch corresponds to the zero window size of the TCP packet after the three-way handshake is complete and time stamp code execution.

A TCP implementation may permit the LISTEN state to be entered with either all, some, or none of the pair of IP addresses and port numbers specified by the applications. A link can become established with any user whose details are unidentified to the server ahead of time. This type of unbound LISTEN is the target of SYN flooding attacks due to the way it is typically implemented by operating systems [45].

3.1 TCP SYN Flood

Internet today is simply accessing the data and using application services of a remote machine. Most of these applications like HTTP, FTP and E-Mail run on top of TCP layer. The accessibility and performance of application services depends on how well the underlying Transport protocol works. By some means if we make the TCP layer unresponsive, the person who is trying to access these services from a remote machine thought that the services are busy / unavailable. In recent years increase in online shopping and online financial transactions makes unavailability of the web services is simply intolerable.

In this attack, the attacker makes the server's TCP layer unresponsive by sending a large number of open connection requests or TCP SYN packets. This is known as SYN flooding or SYN Bombing, Named after specific bit in TCP header specifications.

The TCP SYN flooding weakness was discovered as early as 1994 by Bill Cheswick and Steven Bellovin. The SYN flooding attack was first publicized in 1996, with the release of a description and exploit tool in Phrack Magazine. By September of 1996, SYN flooding attacks has been observed enormously on the internet around the world. SYN flooding was particularly serious in comparison to other known denial of service attacks at that time and even now. The community quickly developed different techniques for preventing or limiting the impact of SYN flooding attacks. Some of these techniques like SYN Cache protection and SYN Cookies protection have become important pieces of the TCP implementations in certain operating systems, although some significantly diverge from the TCP specification and none of these techniques have yet been standardized or sanctioned by the IETF process [45]. SYN Cache is the most

commonly used SYN flooding prevention method and this method is implemented in almost all popular operating systems like Microsoft's windows operating systems, SunOS, Linux and FreeBSD.

Suppose that an attacker directs a large number of SYN requests rapidly to the server with spoofed source IP addresses. In a traditional TCP 3 way hand shake, the server has to create a new TCB for each new connection request it received and save the incomplete state of the connection and the TCP options like window size, Maximum segment size etc. Since the TCB's are limited for each port of the server, the TCB's gets filled up. In Traditional TCP, the server will send several retransmissions for incomplete connections before the timeout period and eventually gets deleted. Even though TCB's are going to be unallocated after certain timeout period, if the attacker manages to keep flooding the server so that no TCB's are free at any given point of time, the TCP layer becomes unresponsive to the legitimate clients.

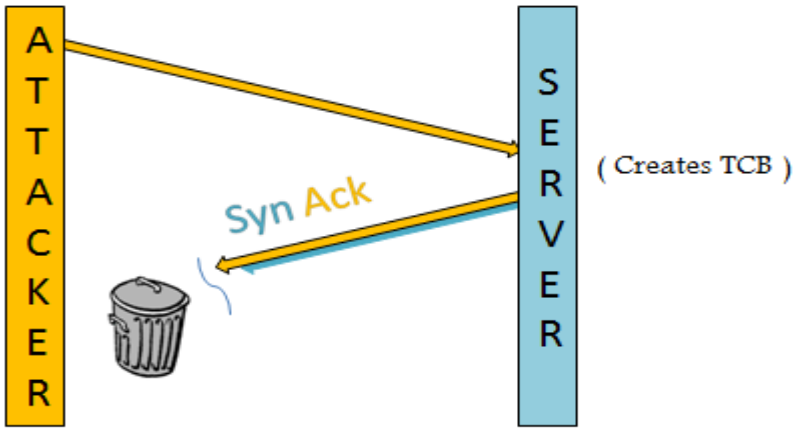

Fig 3.1 TCP SYN Flood

One typical data structure used for communication is the Transmission Control Block (TCB), which is created and maintained during the lifetime of a given connection. The

TCB contains the following information according to RFC 675 (field sizes are notional only and may vary from one implementation to another):

16 bits: Local connection name

48 bits: Local socket

48 bits: Foreign socket

16 bits: Receive window size in octets

32 bits: Receive left window edge (next sequence number expected)

16 bits: Receive packet buffer size of TCB (may be less than window)

16 bits: Send window size in octets

32 bits: Send left window edge (earliest unacknowledged octet)

32 bits: Next packet sequence number

16 bits: Send packet buffer size of TCB (may be less than window)

8 bits: Connection state

The typical TCB size is sum of all fields, which is 280 bits. For each connection, standard transport layer allocates one TCB. Therefore, the total number of connections that can support by the server depends on the number of TCB's available in the server. Some other data structures mentioned in the section 1.1.3 also plays an important role for the communication process between the clients and the server. A TCP synchronize (SYN) attack is a denial-of-service attack that exploits the retransmission and time-out behavior of the Synchronize-Acknowledgement (SYN-ACK) segment during the TCP three-way handshake to create a large number of half-open TCP connections. Depending on the TCP/IP protocol implementation, a large number of half-open TCP connections could do any of the following:

- Use all available memory.
- Use all possible entries in the TCP Transmission Control Block (TCB), an internal table used to track TCP connections. Once the half-open connections use all the entries, further connection attempts are responded with a TCP connection reset.
- Use all available half-open connections. Once all the half-open connections are used, further connection attempts are responded with a TCP connection reset.

3.2 SYN Cache Method of Protection

Research community proposed different techniques to Detect [46-56], Trace back [57, 58] and Defend [59-70] against the TCP SYN flooding attacks. Most of the detection mechanisms proposed was depend on the abnormal traffic flow statistics in the network/Internet and the prevention mechanisms depend on filtering, traffic policing and rate limiting. These mechanisms can be implemented in Internet core, firewalls, routers or end systems. Among the proposed host-based protection methods, the SYN cache method of protection is the most popular solution. In traditional TCP when the server received the first SYN from the client it reserves all the resources like socket, IP control block structure and TCP control block structure, which are required for the communication. The idea of SYN cache mechanism is to lessen the quantity of resource allocation when the first SYN packet is received by the server. The intention is to use a separate small data structure instead of regular large data structures (such as TCB) to store the TCP state and options until the three-way handshake is complete. This prevents the server resources from being consumed by SYN flood attack.

SYN cache implementation replaces the per-socket linear chain of incomplete queued connections with a global hash table, which provides two forms of protection

against running out of resources. There is a limit on the total number of entries in the hash table, which provides an upper bound on the amount of memory that the SYN cache takes up. Host implementing SYN cache uses some randomly generated secret bits by using hashing techniques. The secret bits are hashed along with the IP addresses and TCP ports of a segment. The hash value determined is used to find the location in the global hash table where the incomplete TCB is stored. There is a bucket size limit for each hash value, and when this limit is reached, the oldest entry is dropped. This bucket size limits the amount of time that the machine needs to spend searching for a matching entry, as well as limiting replacement of the cache entries to the subset of the entire cache. One of the major bottlenecks in the original code is the random drop implementation from the linear list, which did not scale. This bottleneck is avoided in the SYN cache, since the queue is split among the hash buckets, which are then treated as FIFO queues instead of using random drop. When a new connection request is received by the server, the server look up in the hash table for the entry, the server creates a new entry if no entry exits for that particular request. Since the hash secret depends on the port number as well if the same client request for the connection again, the server generates a different hash secret. This prevents the attacker from targeting one particular client connection to the server.

3.3 SYN Attack Protection Performance

We measured the performance of SYN cache in the real time traffic circumstances by sending the legitimate client connections and SYN flood attack to the web server at the same time. The legitimate / authentic clients complete the there-way handshake with the server and then send HTTP request for a web page to the server. After receiving the web page, the clients close the connection with server in traditional TCP way of terminating the connection by exchanging FIN packets mentioned in section 1.1.4. On the other hand the attacker's side is made to send a flood of TCP connection requests with spoofed source IP addresses to the web server with no intention to complete the three-way hand shake with the server. The attackers IP source address are fully randomized to overcome any sort of filtering done on the server side.

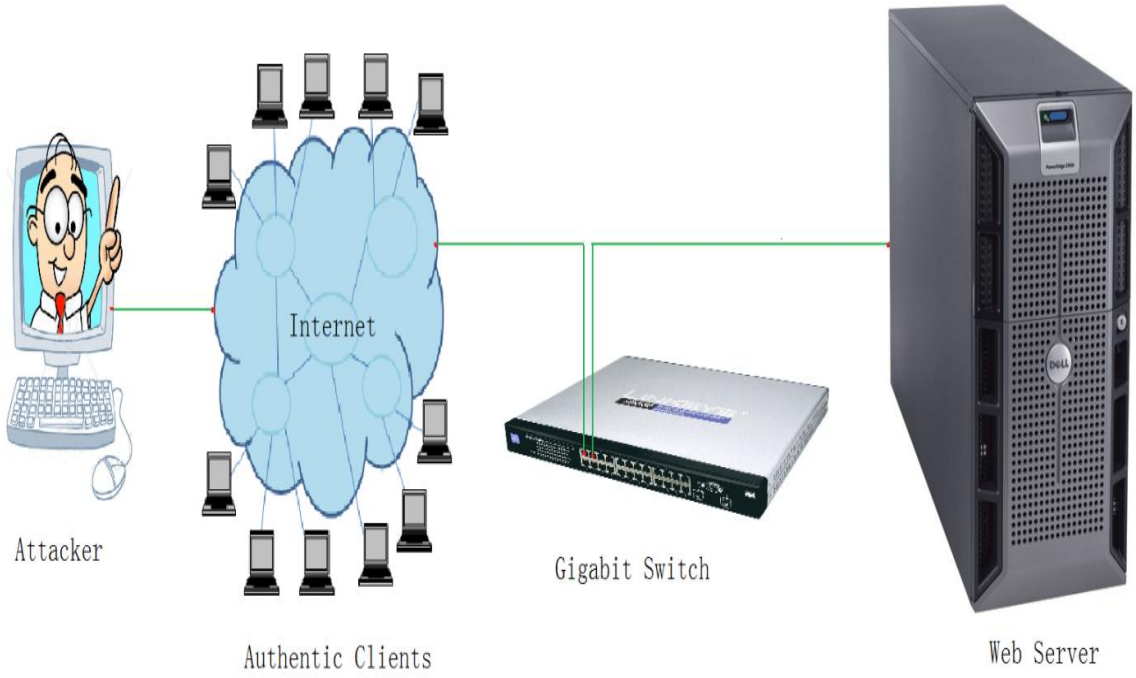

Fig 3.2 Experimental Setup

We measured the number of legitimate client connections that can be established per second with the server under increasing attack loads. The attack load is incremented from low to high intensity in the steps of 0Mbps, 0.25Mbps, 0.5Mbps, 0.75Mbps, 1Mbps, 2Mbps, 3Mbps, 4Mbps, 5Mbps, 6Mbps, 7Mbps, 8Mbps, 9Mbps, 10Mbps, 20Mbps, 30Mbps, 40Mbps, 50Mbps, 60Mbps, 70Mbps, 80Mbps, 90Mbps and 100Mbps in all the following experimental results to find the connection rate behavior at lower and higher intensity of attack traffic. The duration of each attack load is kept for 10 minutes (600 seconds) and the statistical readings are collected for each second. i.e.' 600 reading for each attack load.

The server CPU utilization and Memory status of the server under different loads of SYN attack are shown in fig 3.3 and 3.4.

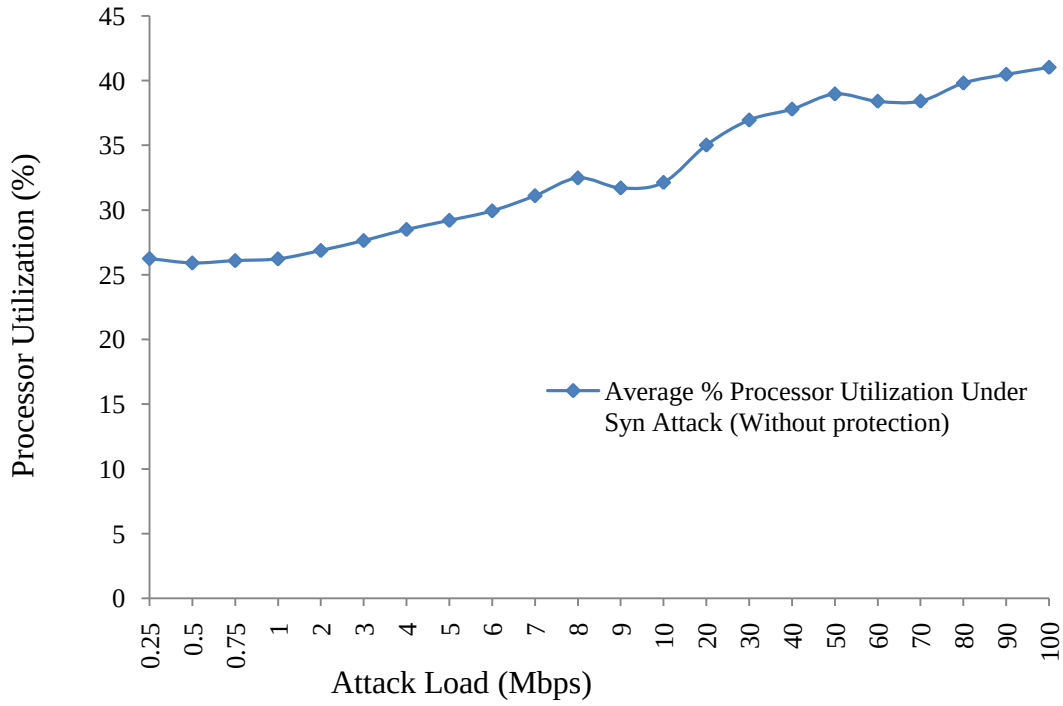

Fig 3.3 Server CPU utilization (without protection) under SYN Attack

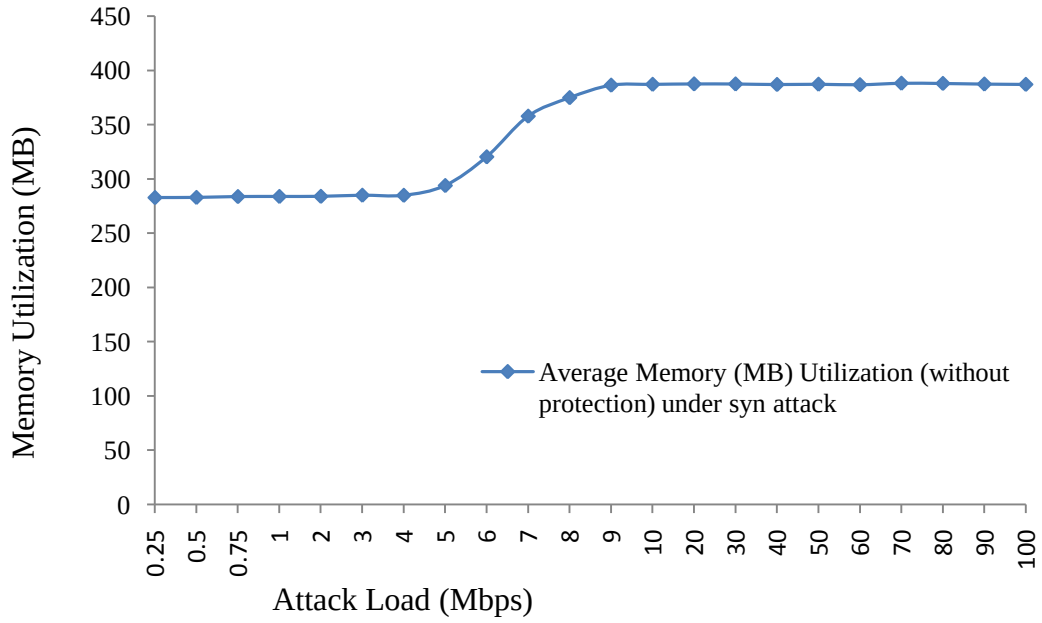

Fig 3.4 Memory Consumption (without protection) under SYN Attack

The powerful quad core CPU utilization of the server is increasing linearly as the attack load increases (nonlinear) when there is no protection. The maximum CPU utilization 41% is reached at 100Mbps of SYN attack load. The memory consumption is just 387MB at 100Mbps attack load which is well below the 8GB RAM installed in the server. From the graphs (fig 3.3 & 3.4), it is observed that the server CPU and Memory are not consumed completely because of the SYN Attack.

The total number of TCP connections in SYN_RECEIVED state when the server is under SYN attack is shown in the figure 3.5. Connections in SYN_RECEIVED state is also referred as half-open TCP connections means incomplete TCP connections The maximum number of half open connections supported by the server at any given instant depends on the backlog size. The registry parameters that will control the backlog size are mentioned in appendix B.

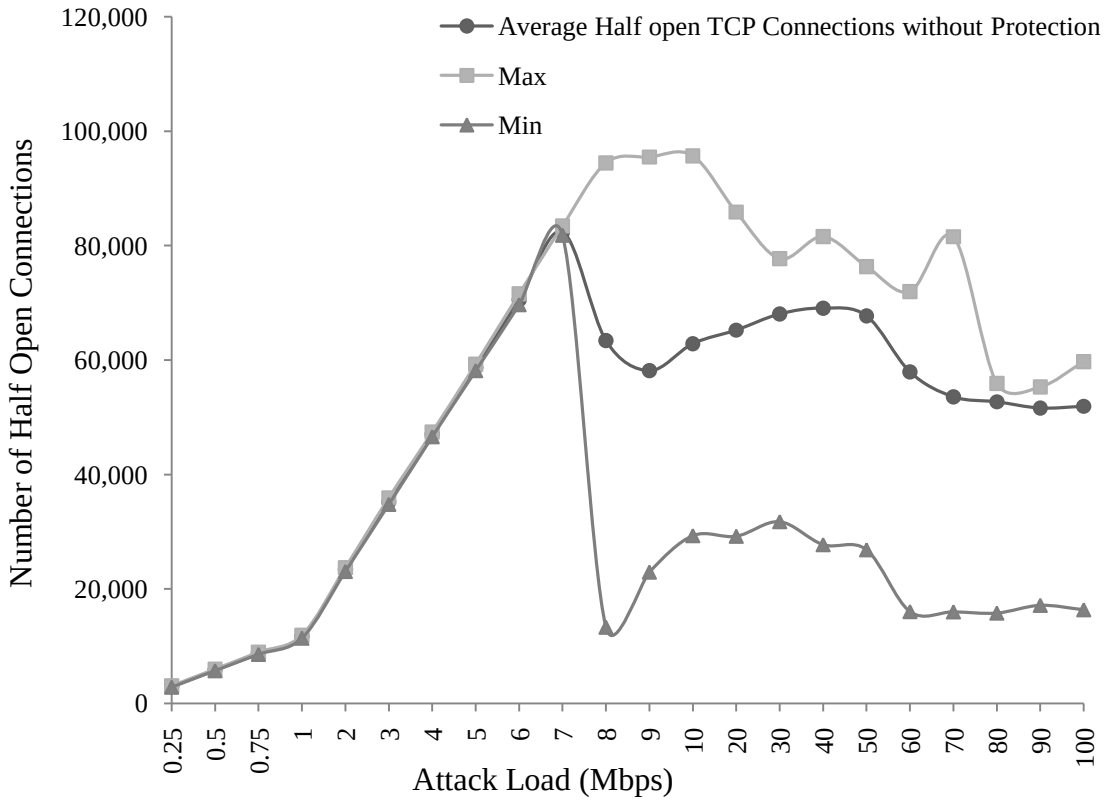

Fig 3.5 Server TCP Connections in SYN_RECEIVED State (without protection) under SYN Attack

TCP half-open connections are increasing linearly at lower loads of SYN attack until 7Mbps. After this point, the number half-open connections are falling at higher attack load. The average half open connections at each attack load shown in fig 3.5 is an average of 200 reading. These reading are manually logged with the help of NETSTAT command.

Netstat -n -p tcp | find/c “SYN_RECEIVED”

It is observed in fig 3.5 that the total number of half-open connections in server is unstable after 7Mbps of SYN attack load.

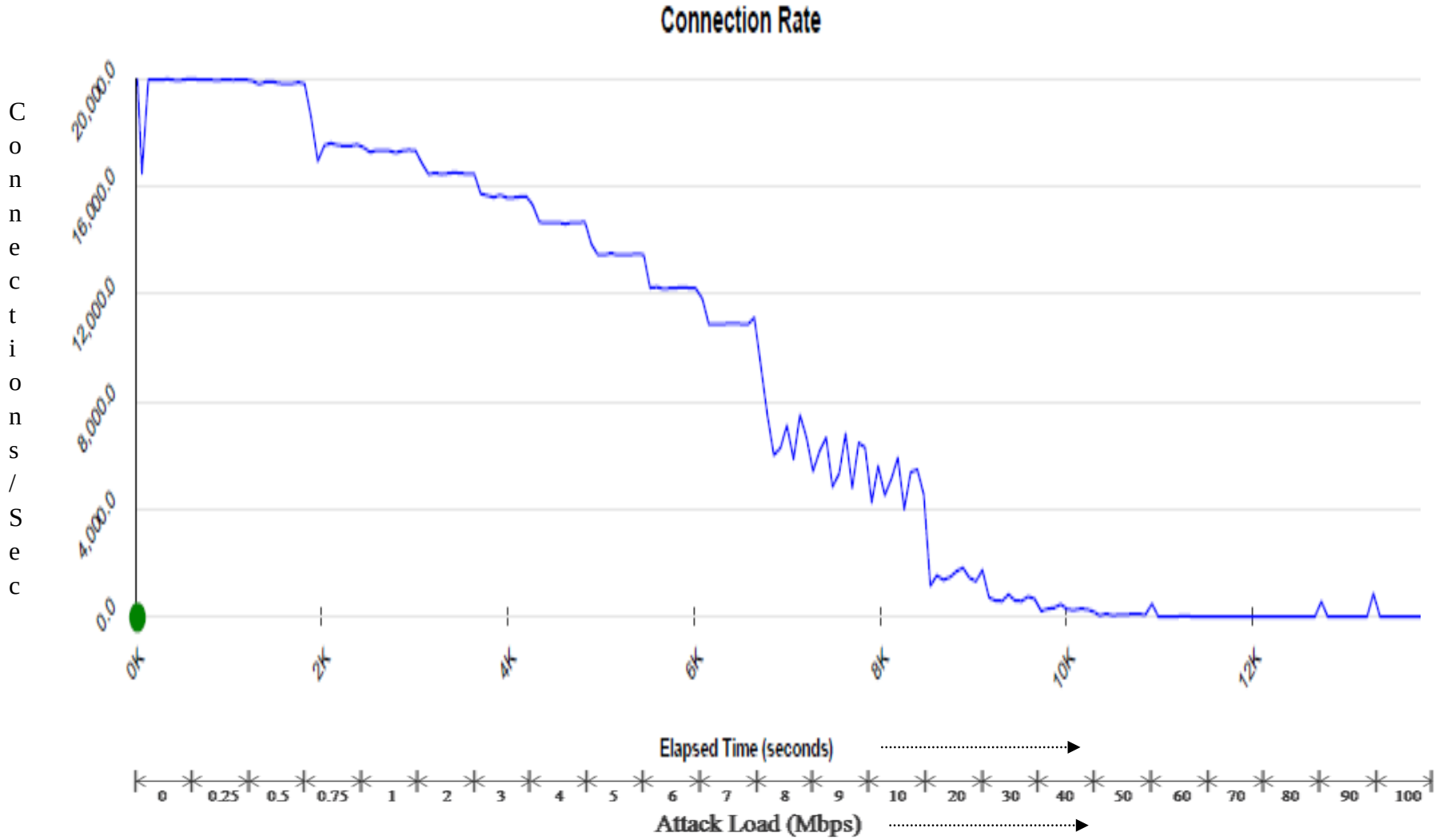

Fig 3.6 Successful Legitimate Client Connections/sec as the Attack Load increases with Time without SYN Attack Protection

Fig 3.6 & 3.7 show the average successful legitimate connections established with the web server and the server is under SYN attack at the same time. The legitimate client connections are decreasing rapidly with increasing SYN attack load. At 0Mbps attack load that means when there is no attack, the legitimate clients are able to form around 20,000 successful connections per second. After 60Mbps of SYN attack load legitimate client connections/sec with the server are almost depleted /well below 100 connections/sec. It is observed that around 5,000 Connections per second are successful when the SYN attack load intensity is 10Mbps.

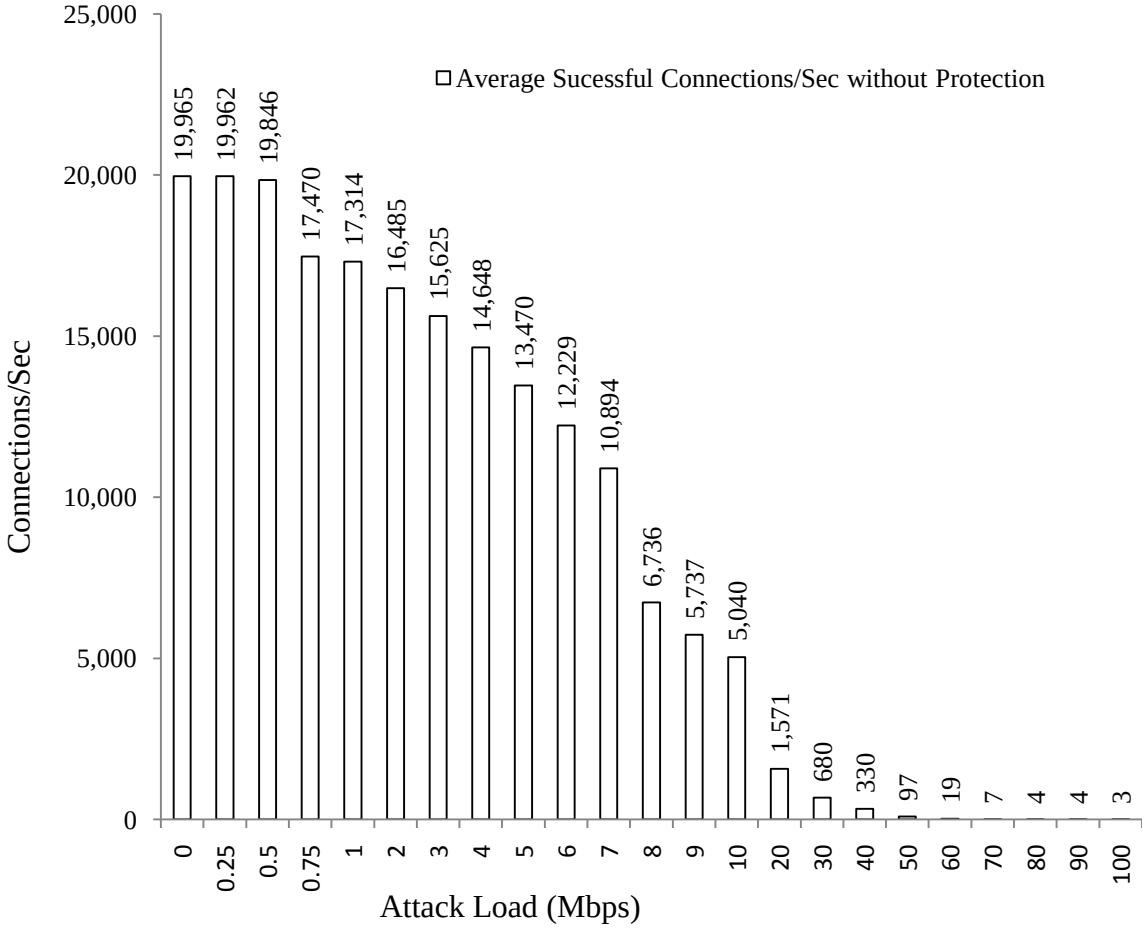

Fig 3.7 Successful legitimate client connections/sec without Protection

To mitigate the impact on a host experiencing a SYN attack, TCP/IP SYN cache protection minimizes the amount of resources devoted to incomplete TCP connections and reduces the amount of time before abandoning the half-open connection. When a SYN attack is detected, TCP/IP in Windows Server 2003 and Windows XP lowers the number of retransmissions of the SYN-ACK segment and does not allocate memory or table entry resources for the connection until the TCP three-way handshake has been completed. Microsoft provided a feature called SynAttackProtect in the server operating system. This feature is available in all versions of windows server 2003 but enabled by default only in some versions of windows server 2003 operating systems. The Microsoft provided definition for this protection as follows [71].

"SYN attack protection involves reducing the amount of retransmissions for the SYN-ACK's, which will reduce the time for which resources have to remain allocated. The allocation of route cache entry resources is delayed until a connection is made and the connection indication to application is delayed until the three-way hand shake is completed."

With the above details regarding the protection mechanism from Microsoft, It is certain that the SynAttackProtect feature provided by Microsoft falls under SYN cache method of protection. The action taken by the SYN attack protection mechanism only occurs if TcpMaxHalfOpen and TcpMaxHalfOpenRetried setting s are exceeded. The three configurable threshold parameters to trigger TCP's SYN attack flooding protection feature are explained below.

1. TcpMaxHalfOpen specifies how many connections the server can maintain in the half-open state before TCP/IP initiates SYN flooding attack protection, by default it is 500 in windows server 2003.
2. TcpMaxHalfOpenRetried specifies how many connections the server can maintain even after a connection request has been retransmitted before TCP/IP initiates SYN flooding attack protection by default it is 400 in windows server 2003.
3. TcpMaxPortsExhausted specifies how many connection requests the server can refuse before TCP/IP initiates SYN flooding attack protection by default it is 100 in windows server 2003.

All the three entries mentioned are used only when SYN flooding protection is enabled on the server, that is, when the value of the SynAttackProtect entry is 1 and the value of the TcpMaxConnectResponseRetransmissions entry is at least 2.

The behavior of TCP/IP protocol stack in the windows server 2003 operating system heavily depends on the registry parameters. A list of all the TCP/IP parameters in the registry and their effect on the behavior of the protocol stack are included in the appendix B. we recognized the research efforts made by Microsoft in deciding these registry key parameters for the stable response of server and its services. Therefore, we kept most of these parameters in the default state or in the state recommended by the Microsoft as mentioned above for the stable response of the server.

The next step is to enable the SYN attack protection feature in windows server 2003 and observe the server behavior under SYN attack. In the remaining part of this chapter, we will observe the server ability to provide services to legitimate clients when

SYN attack protection is enable and compare it with the results we had when the SYN attack protection is not active. The SYN attack protection thresholds mentioned earlier are in the default state/value for all the experiments we conducted in this chapter.

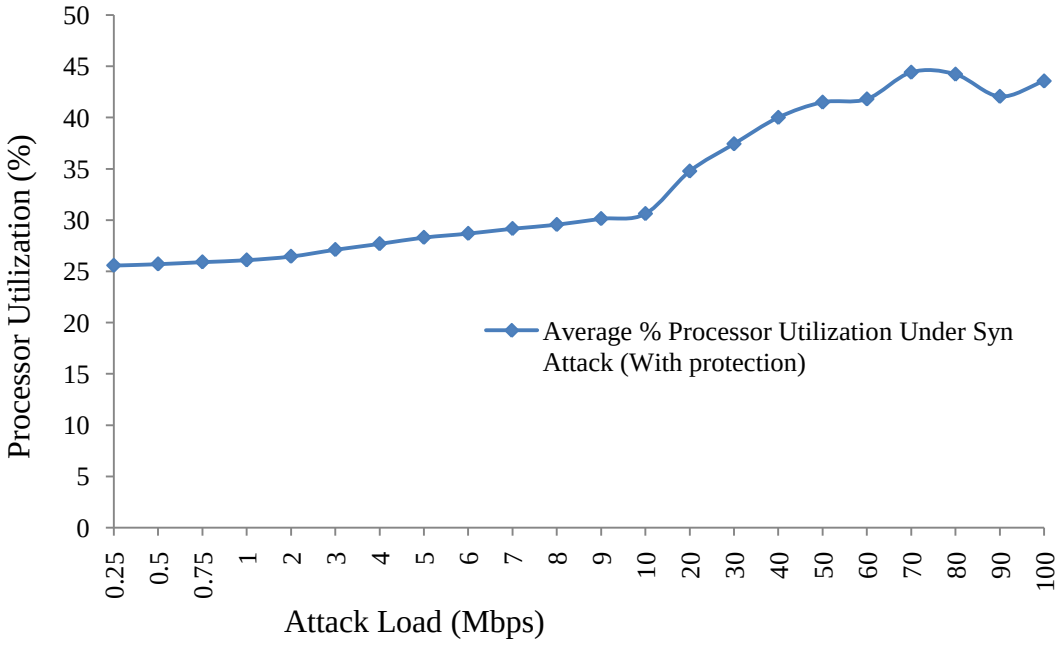

Fig 3.8 Server CPU utilization (with SYN Attack Protection) under SYN Attack

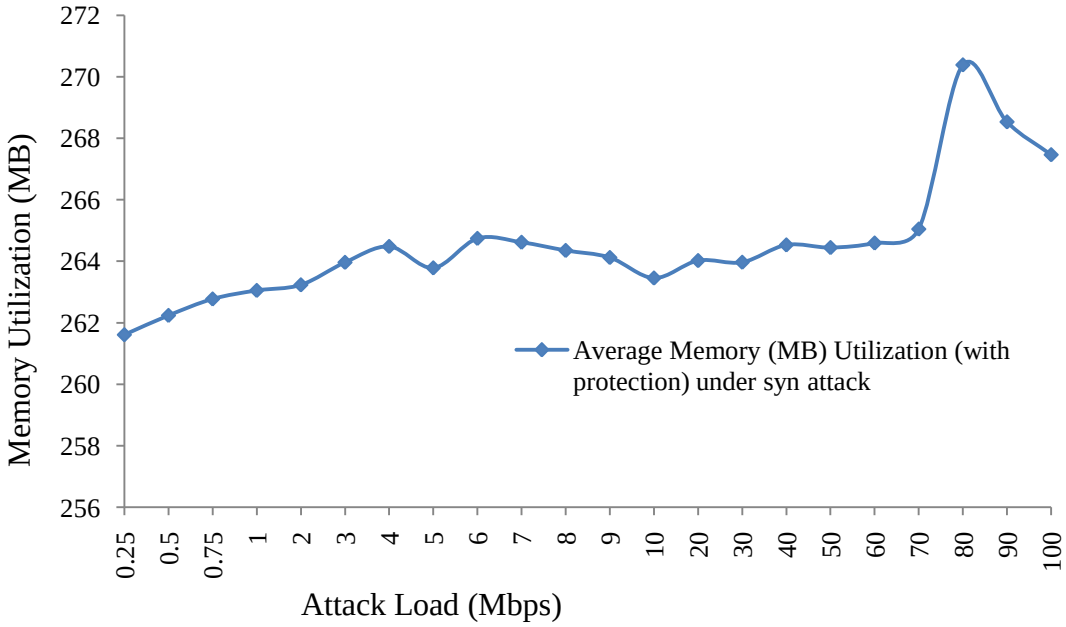

Fig 3.9 Memory Consumption (with SYN Attack Protection) under SYN Attack

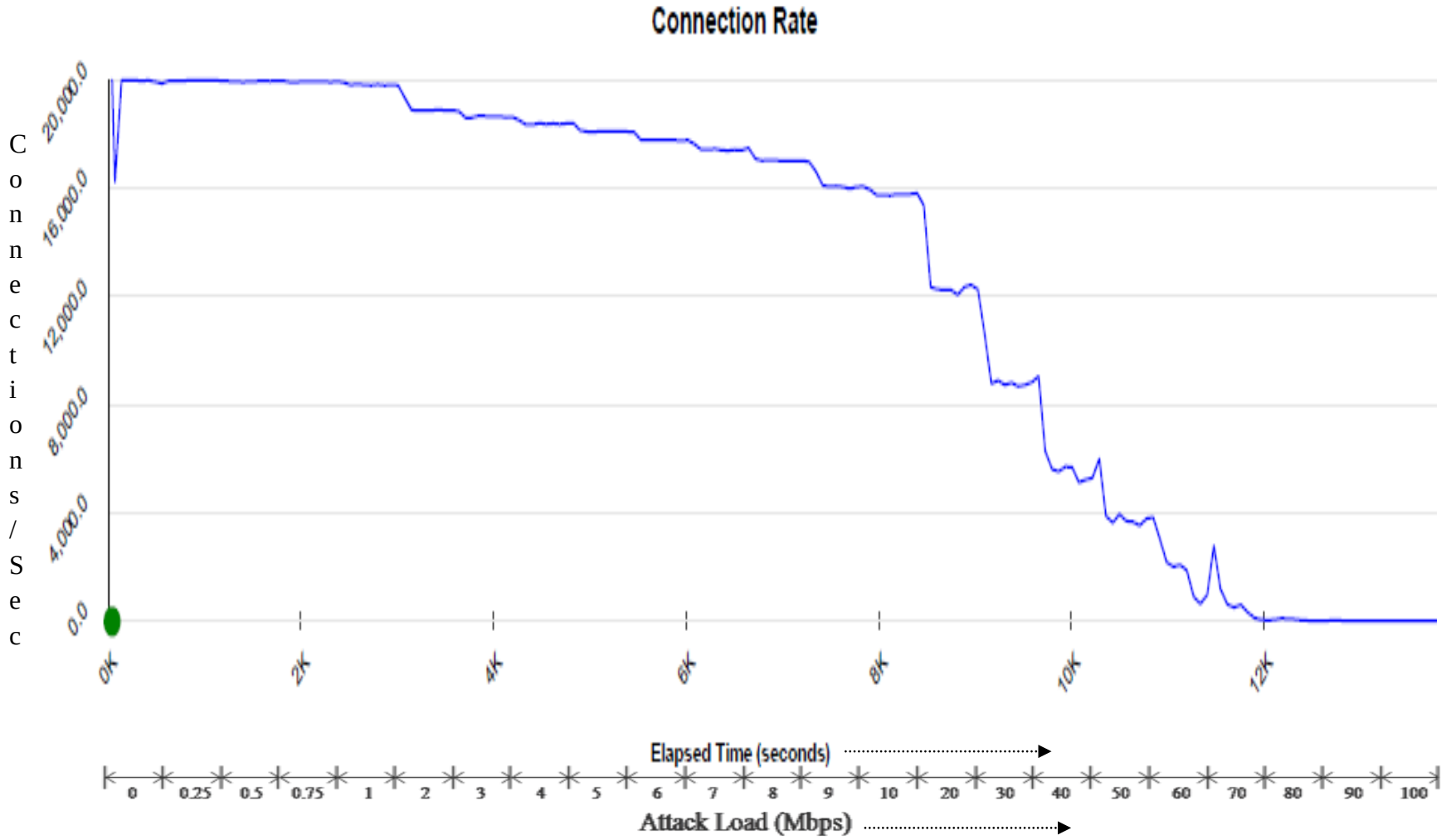

Fig 3.10 Successful Legitimate Client Connections/sec as the Attack Load increases with Time when server SYN attack protection enabled

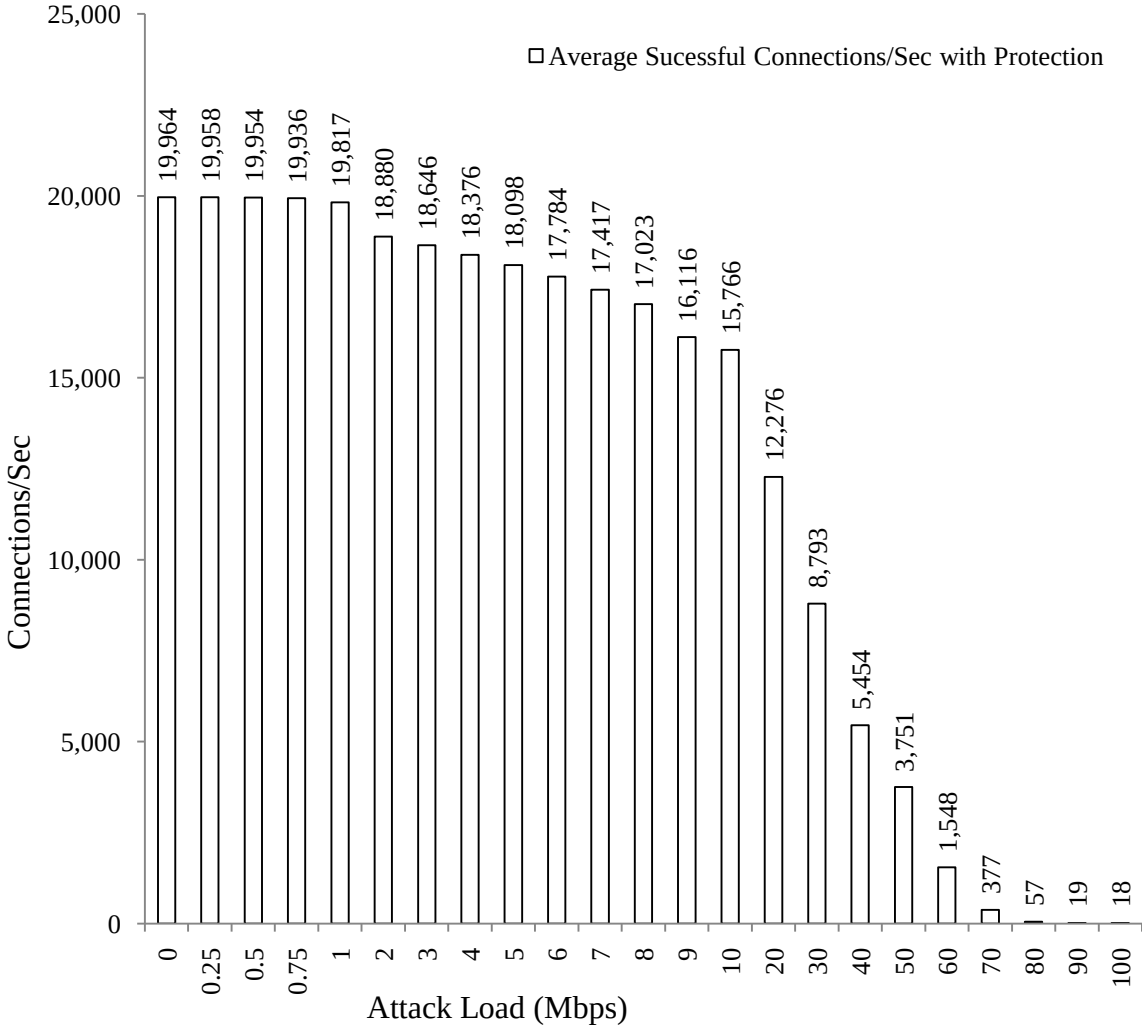

Fig 3.11 Successful legitimate client connections/sec with SYN attack protect (Bar chart)

The network topology created for this testing is same as shown in figure 3.2. The CPU and Memory usage of the server under SYN attack when protection enabled is shown in the figure 3.8 & 3.9 respectively. The CPU utilization is nearly the same with and without protection. The memory consumed by server under SYN attack is significantly reduced when the SYN attack protection enabled. Compared to the memory resources available in the server and the cost of memory today, it is not significant.

The successful legitimate client connections rate vs. attack load when the server SYN attack protection enabled is shown in the Fig 3.10 &3.11. It is observed that even with protection enabled the successful connection rate is decreased as the attack load increases. The legitimate connections are unable to establish and the connection rate is less than 100 connections /sec after 80Mbps attack load. This is an improvement over the previous scenario where the connections/sec fell below 100 at 60 Mbps without SYN protection. It is observed from fig 3.11 that the successful connection rate at 10Mbps of attack load is around 16,000 connections / sec, which is more than two times the successful connection rate we achieved without the SYN flood attack protection. The successful connection rate is improved significantly for a given attack load but at higher attack loads after 60Mbps, the legitimate connections are unable to be established.

Comparison of the results of these two experiments with and without protection gives a clear picture as shown in the Fig 3.12. When we use the TCP SYN attack protection, the authentic new client connection rate to the web server under TCP SYN attack was improved by 226 % at 10Mbps Attack Load. From the results presented in this chapter, it is evident that the legitimate client connection rate improved by the use of SYN attack protection. However, SYN attack protection is not effective at higher loads of SYN attack. But if we increase the number of half open connection limit on the server, the successful connection rate of clients may improve at higher loads as well [73]. The maximum number of the half-open connections can be computed from the bandwidth of the incoming link connected to the server and the timeout used by the server to discard pending requests. This is a kind of brute force solution that wastes a lot of kernel memory and slows down the server.

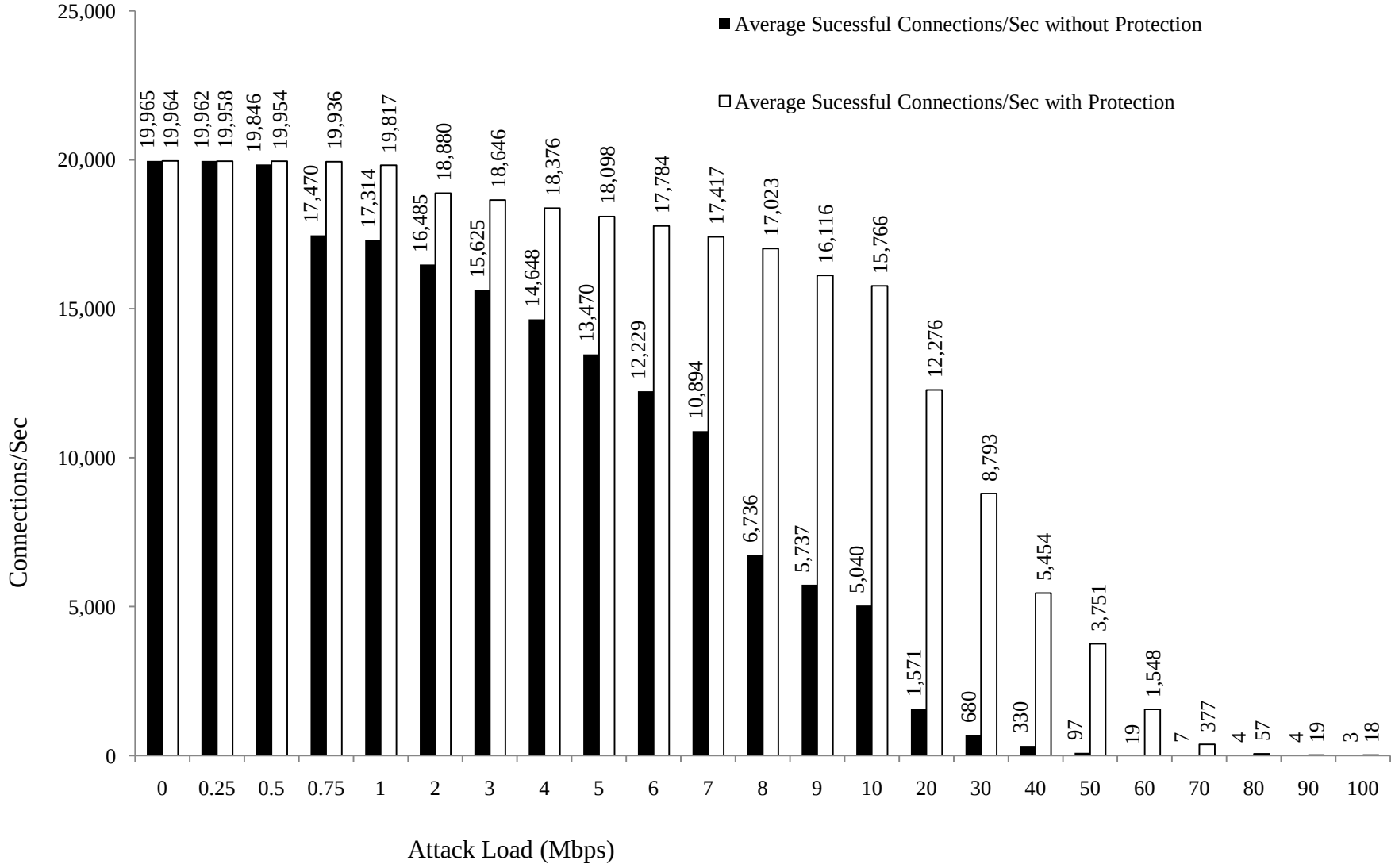

Fig 3.12 Comparison of successful client connections with and without SYN Attack Protection

response time, but it can be effective in public servers serving large communities of clients, since such servers have extensive hardware resources. Even if you increase the half-open connection limit, it is possible that at some higher load attack traffic the hash table fills up/Overflow with forged connection requests. A good thing about the SYN cache method of prevention is that implementation of SYN cache does not require modification of any rules of the traditional TCP. SYN cache method needs implementation only on the server side and it is completely transparent to the clients.

CHAPTER IV

SYN COOKIES AND SYN PROXY

Besides SYN Cache protection method, there are two other popular methods of protection namely SYN cookies and SYN proxy. Servers in general limit the number of TCP half-open connections it can maintain because it has to assign resources like memory and processing power for every SYN request. Attackers are using the TCP half open connection limitation to exploit and cause denial of service to legitimate clients. Attacker will flood the server with SYN requests to occupy the available space for half-open connections so that the real clients have no chance to communicate with the server. We can think of two possible solutions for this problem. One of them is to have Abundant/Unlimited resources at the server so that the server can setup higher or no limits on half open connections and the attacker is never able to fill them up. Off course, this is not possible due to limited hardware capabilities like memory (Random Access Memory) and processor. The other solution is not to assign/reduce the amount of resources for the half-open connections on the server until the server know whether the client is legitimate or not. Reducing the resource allocation for each received SYN packet until the there-way handshake is complete, is the concept behind SYN cache method of protection. The other option is not to dedicate any resources to half-open connections, which can be achieved with SYN cookies using Cryptography. Although the concept

behind cookies protection method is straightforward, its implementation is more complicated. SYN cookies method is a very good protection mechanism to have but there are some disadvantages as well. Implementing SYN cookies technique makes its TCP stack does not support some features of Traditional TCP.

4.1 Introduction to SYN Cookies

SYN cookies go one-step further to SYN cache and do not keep any state information of received SYN packets on the server. Instead, they encode all of the sate information required using cryptographic algorithms into the sequence number transmitted on the SYN-ACK. If the source IP address of the SYN request is not a spoofed one then the server reconstructs the state information required to store in TCB from the final ACK sent by the client to complete the three-way handshake. A TCP SYN cookie was proposed by D. J. Bernstin in September 1996 [72]. There is no need to allocate any buffer space for half-open connections in the server. The challenge in implementing SYN cookies is to make cookies hard to guess on the client side to avoid attacks using ACK segments with forged, valid cookie. One more challenge is to make them obey the properties of the traditional TCP.

SYN cookies should fit in the initial sequence number (ISN) space in the TCP header, which is 32 bits. Therefore, the cookies must follow the rules of ISN as in traditional TCP such as slowly increasing over time. Therefore, cookies entirely cannot be a random value generated by some cryptographic algorithm. The server should be able to check, whether the cookies received in the final ACK is a forged one or not. These are the challenges in implementing cookies and the next paragraph elaborates on the efficient way of creating SYN cookie.

A SYN cookie is included in initial sequence number (ISN) as mentioned but the slowly growing nature of ISN numbers is disobeyed and the generated SYN cookies are completely independent of the algorithm used to generate traditional TCP ISN values. SYN cookies are chosen based on the clients initial sequence number, TCP/IP addresses, Port numbers of clients, a time counter, constant secret values and a 3-bit encoding of the server's fixed MSS values [73]. The actual bits comprising the SYN cookie are chosen to be the bitwise difference (exclusive-or) between the SYN's sequence number and a 32 bit quantity computed so that the top five bits (T) come from a 32-bit counter value modulo 32, where the counter increases every 64 seconds. The next three bits (M) encode a usable MSS closest to the one in the received SYN, and the bottom 24 bits (S) are a server selected secret function of pair of IP addresses, the pair of port numbers, and the 32-bit counter used for the first five bits [74].

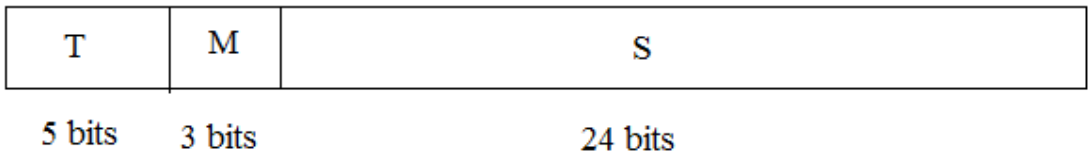

Fig 4.1 SYN Cookie Structure

The SYN cookies implementation proposed by Bernstein is as follows [72]. SYN cookies implementation may vary slightly from one platform to other.

Maintain two (constant) secret keys, sec1 and sec2.

Maintain a (constant) sorted table of 8 common MSS values showed in table 1, msstab [8].

Keep track of a "last overflow time".

Maintain a counter that increases slowly over time and never repeats, such as "number of seconds since 1970, shifted right 6 bits".

When a SYN comes in from (saddr, sport) to (daddr, dport) with ISN x, find the largest i for which msstab[i] <= the incoming MSS. Compute

z = MD5 (sec1, saddr, sport, daddr, dport, sec1)

+ x

+ (counter << 24)

+ (MD5 (sec2, counter, saddr, sport, daddr, dport, sec2) % (1 << 24))

and then

y = (i << 29) + (z % (1 << 29))

SYN Cookie data	0	1	2	3	4	5	6	7
MSS value	64	256	512	536	1024	1440	1460	4312

Table 1: MSS predefined values

Notice it does not set the top five bits from the counter modulo 32, as the previous description did, but instead uses 29 bits from the second MD5 operation and 3 bits for the index into the MSS table.

SYN Cookies drawbacks: SYN cookies method is definitely a dependable concept as of now. However, there are some disadvantages of this method of protection as follows.

1) Since we have only 32 bits available in TCP header for SYN cookie, it is not difficult or impossible to encode all TCP options in SYN cookie. Therefore, we are not able to use all the TCP options, which are designed to improve TCP performance. At present, they only carry a 3-bit encoding of 8 predefined MSS values. All the other options like window scaling, selective acknowledgment, and time stamping for Round-Trip Time Measurement (RTTM) or Protection Against Wrapped Sequence numbers (PAWS) are simply ignored.
2) In SYN cookies implementation we are not storing any of the connection state information on the server, In case if the final ACK is lost in the transmission line,

retransmission is not possible by the server.

3) Hackers may guess the SYN cookie and flood with ACK packets, which allows connections to establish on the server.
4) SYN cookies are incompatible with transactional TCP (T/TCP). T/TCP is a variant of TCP protocol. It is an experimental addition for efficient transaction-oriented (request/response) service. It was developed to fill the gap between TCP and UDP, by Bob Braden in 1994. Detail specifications can be found in RFC 1644.

There is a lot of debate going on in the research community weather to implement the SYN cookie in the server operating system or on the router interface. Which place is better for SYN cookies to implement? Linux OS and some other server operating systems implemented this feature in the kernel and allow administrators to dynamically activate their use. Linux operating system followed a hybrid approach by implemented SYN cookies along with SYN cache. In Linux, SYN cookie protection is only used if the hash bucket is full and kernel has to discard the older SYN requests received [74]. That means SYN cookies are implemented along with SYN cache method of prevention in Linux. Therefore, this helps the server further to provide connections to legitimate clients even at higher attacks of SYN attack load. Microsoft based operating systems have no SYN cookies protection feature and do not support SYN cookies. Some research is going on in developing new protocols, which provide reliable data transfer services to applications by using connection-oriented approach without storing the state of the connection at the server one like SYN cookie [75].

4.2 Performance of SYN Cookies

Juniper Giga bit router J4350 has the protection features, namely SYN cookies and SYN proxy. This router is used for the evaluation of SYN cookies and SYN proxy protection mechanism. The performance evaluation is done almost in the same kind of network environment (Fig 4.2) that we used for the SYN cache method as shown in the figure 4.2. We configured static routing to reduce the processing and maintenance overhead on the router. The clients and attackers roles are the same as mentioned earlier in SYN cache setup in chapter 3.

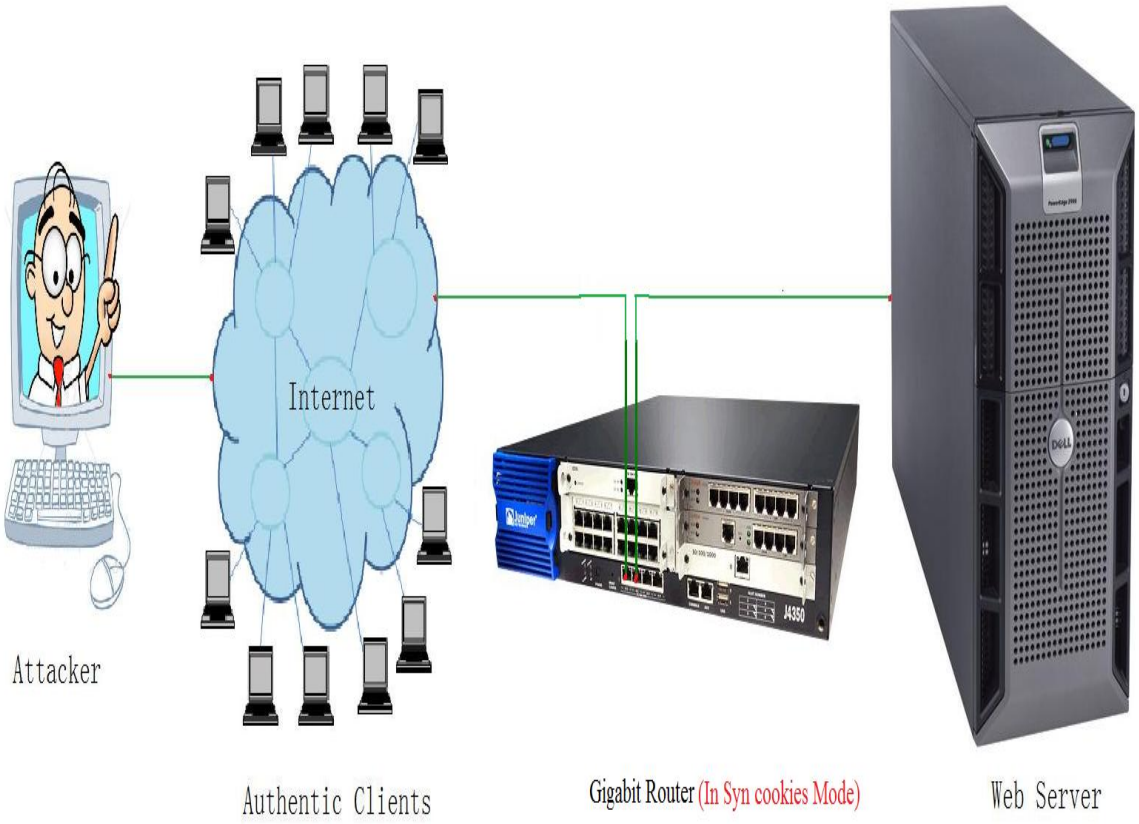

Fig 4.2 Experimental Setup

Some of the limitations of the Juniper Giga bit router J4350 are mentioned in the introduction section 1.6.2. Because of the connection/sec limitations of the router, we decided to test the router new connections switching capability without enabling any SYN flood attack protection mechanisms available at the router or at the server. According to Juniper Networks, this router has the capability to forward 10,000 new sessions per second. In this case, the maximum possible number of connections without any attack is 10,000 unlike in case of working with the server, which supports 20,000 new Connections/sec in chapter 3. For evaluation, we considered three scenarios. In first case, we conducted the experiment without enabling any kind of SYN flood protection at the router or the server. In the second scenario, we enabled the SYN attack protection feature available at the server. The third scenario has SYN cookies enabled at the router and SYN attack protection enabled at the server as well. In all of the three scenarios, the attack traffic is directed to the server along with the regular legitimate client traffic.

Case 1: No protection at the router & No TCP SYN attack protection at the server

In this case, the router has no protection of any kind against SYN flood attack and the SYN attack protection in the server is disabled. The successful legitimate client connections established /sec with the server under SYN attack load is shown in the fig 4.3. Please notice that the attack load variation on x –axis is not linear. The configured attack load step size is varied accordingly to study the server response at both low intensity and high intensity attacks. Since there is no protection enabled in the router, it forwards packets on first come first serve basis and it can only support 10,000 Sessions/sec. Not all the connection traffic including legitimate clients traffic and attack traffic may be able to reach the server due to this limitation of the router. Fig 4.3

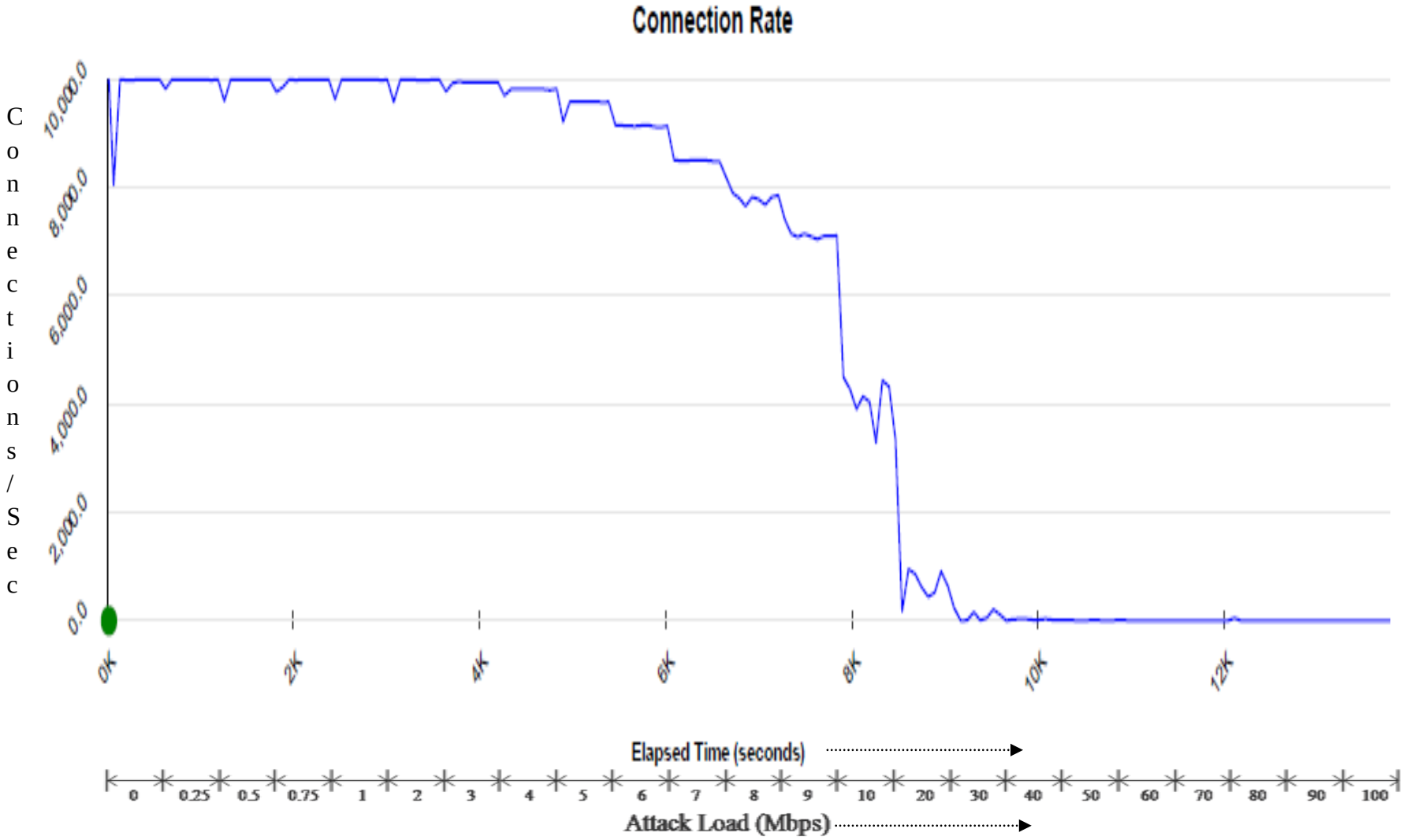

Fig 4.3 Successful client connections without protection at the server as well as at the router under SYN Attack

shows the number of legitimate client connections successful in establishing a connection with the server in this scenario. It is observed from the fig 4.3 that the connection rate is decreasing linearly as the attack load increases at low intensity attack loads. At high intensity of SYN attack, the successful connections are dropping rapidly as the attack load increases. Up to 9Mbps of SYN attack load, the successful connection rate is reasonably good even though the connection rate is decreasing with increase in attack load. However, at higher attack loads more than 9Mbps, the connection rate is dropping rapidly and it is below 100 at 30Mbps of SYN attack load. This result will be used as a base line in other scenarios to compare the effectiveness of protection methods.

Case 2: No protection at the Router & TCP SYN attack protection enabled at the server

Even though we analyzed server SYN attack protection behavior under SYN attack in chapter 3, Because of the router sessions limitation it is interesting to see the SYN attack protection effectiveness in this scenario. The SYN attack protection in the server is enabled in this case and the router still does not have any kind of SYN flood protection. Fig 4.4 shows the successful legitimate client connections in this scenario when the server is under SYN flood attack. In this case, the legitimate client connection rate variation is similar to case 1 except that the number of connections/sec at any particular attack load is slightly increased. The client connections fall below 100 at 40Mbps attack load and the connections depleted to zero after 50Mbps attack load (Table 2). The connection rate is improved a little because of the SYN attack protection in the server when compared to the fist case. It is observed that in this case the router session limitation is playing a significant role than SYN attack protection in the server. This clearly shows that the connection rate in all the experiments conducted in this chapter are

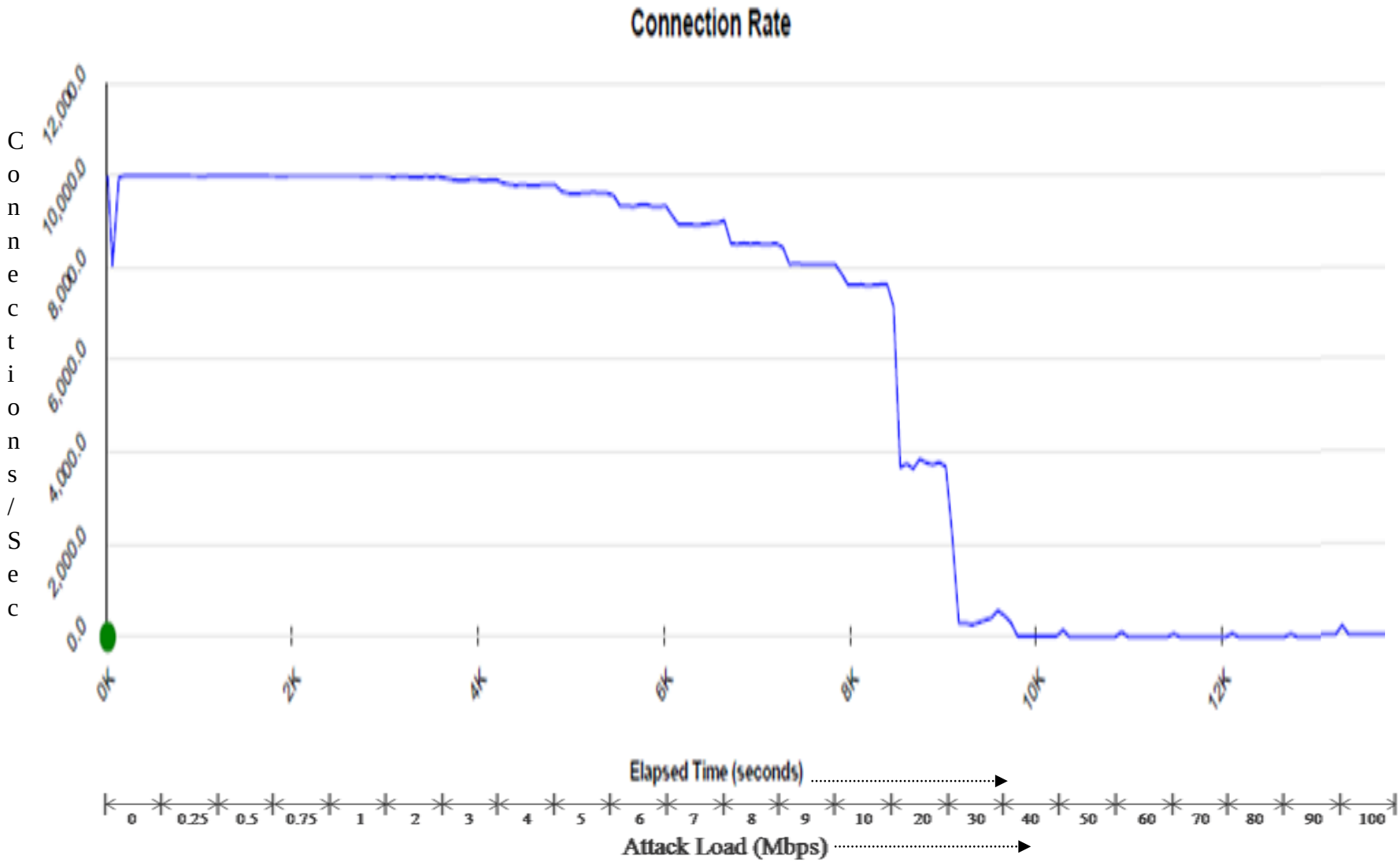

Fig 4.4 Successful client connections with Protection at the Server and no SYN flood protection at the router under SYN Attack

mostly depends on the router and the protection method in the router. We also experimented one more scenario, where the SYN cookies protection in the router is active and without any protection in the server. We observed that the results are very near to the results we got in case 3(where both the protections are active).This again shows that the SYN cookies protection in the router is playing a major role in connection rate behavior than SYN attack protection in the server.

Case 3: TCP SYN Cookies protection at the router and TCP SYN attack protection at the server are active.

Juniper router SYN cookies protection was enabled along with SYN attack protection in the server. Fig 4.5 shows how the SYN cookie protection in juniper router works. JUNOS software with enhanced services can impose a limit on the number of SYN segments permitted to pass through the firewall per second. We can base the attack threshold on the destination address and port (Attack threshold), the destination address only (Destination threshold), or the source address only (Source Threshold).When the number of SYN segments per second exceeds one of these thresholds, JUNOS software with enhanced services starts using the protection mechanism and replying with SYN-ACK segments for incoming connection (SYN) requests.

When SYN cookie is enabled on the router and one of the thresholds is exceeded, the router does the TCP-negotiating for the destination server. Therefore, the router generate SYN cookie for every SYN packet it received and send SYN-ACK back to the client. After receiving the final ACK from the client, the router checks whether the client is legitimate or not by processing the SYN cookies received in the ACK ISN. Then the router send SYN packet for three-way handshake to the server and form a connection

with the server on behalf of the client as shown in fig 4.5 steps 4, 5 and 6. There after the data transfer between the client and the server initiates as shown. The thresholds for SYN flood protection to be activated are defined as follows [76].

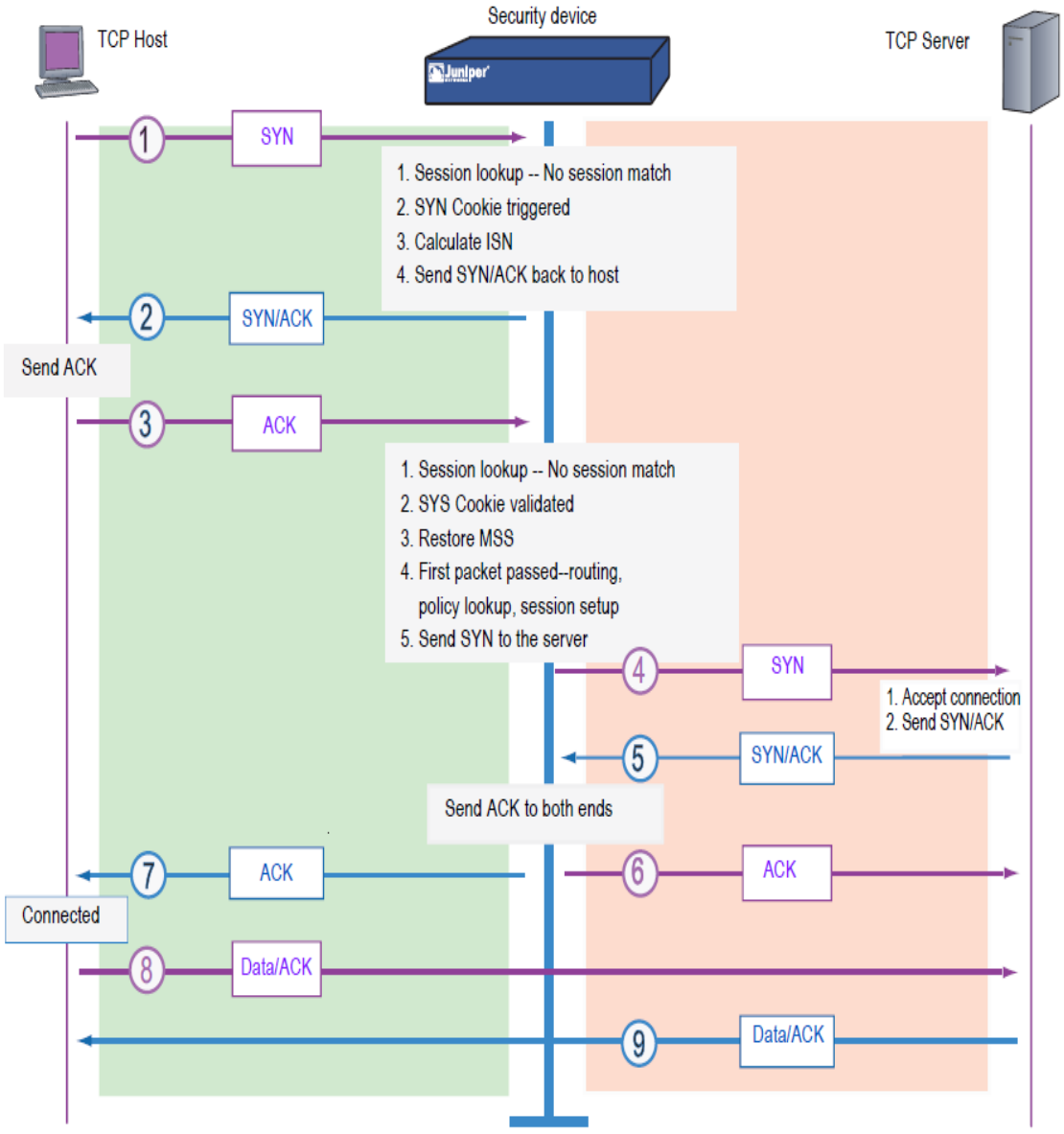

Fig 4.5 Establishing a connection with SYN cookie Active [76]

Attack Threshold: This option allows you to set the number of SYN segments to the same destination address and port number per second required to activate the SYN protection mechanism.

Alarm Threshold: This option allows you to set the half-complete TCP connection requests per second after which JUNOS software with enhanced services enters an alarm in the event log. The value you set for an alarm threshold triggers an alarm when the number of half-completed connection requests to the same destination address and port number per second exceeds that value.

Source threshold: This option allows you to specify the number of SYN segments received per second from a single source IP address—regardless of the destination IP address and port number—before JUNOS software with enhanced services begins dropping connection requests from that source.

Destination threshold: This option allows you to specify the number of SYN segments received per second for a single destination IP address before JUNOS software with enhanced services begins using SYN protection mechanism to connection requests to that destination.

Timeout: This option allows you to set the maximum length of time before a half-completed connection is dropped from the queue.

The threshold values we used for the testing are as fallows

Attack threshold = 200 connections per second

Alarm threshold = 512 connections per second

Source threshold = 4000 packets per second

Destination threshold = 4000 packets per second

Timeout = 20 seconds

Fig 4.6 shows the results when the attack directed to the server and the router is configured to mentioned threshold limits with SYN cookies protection enabled. It is

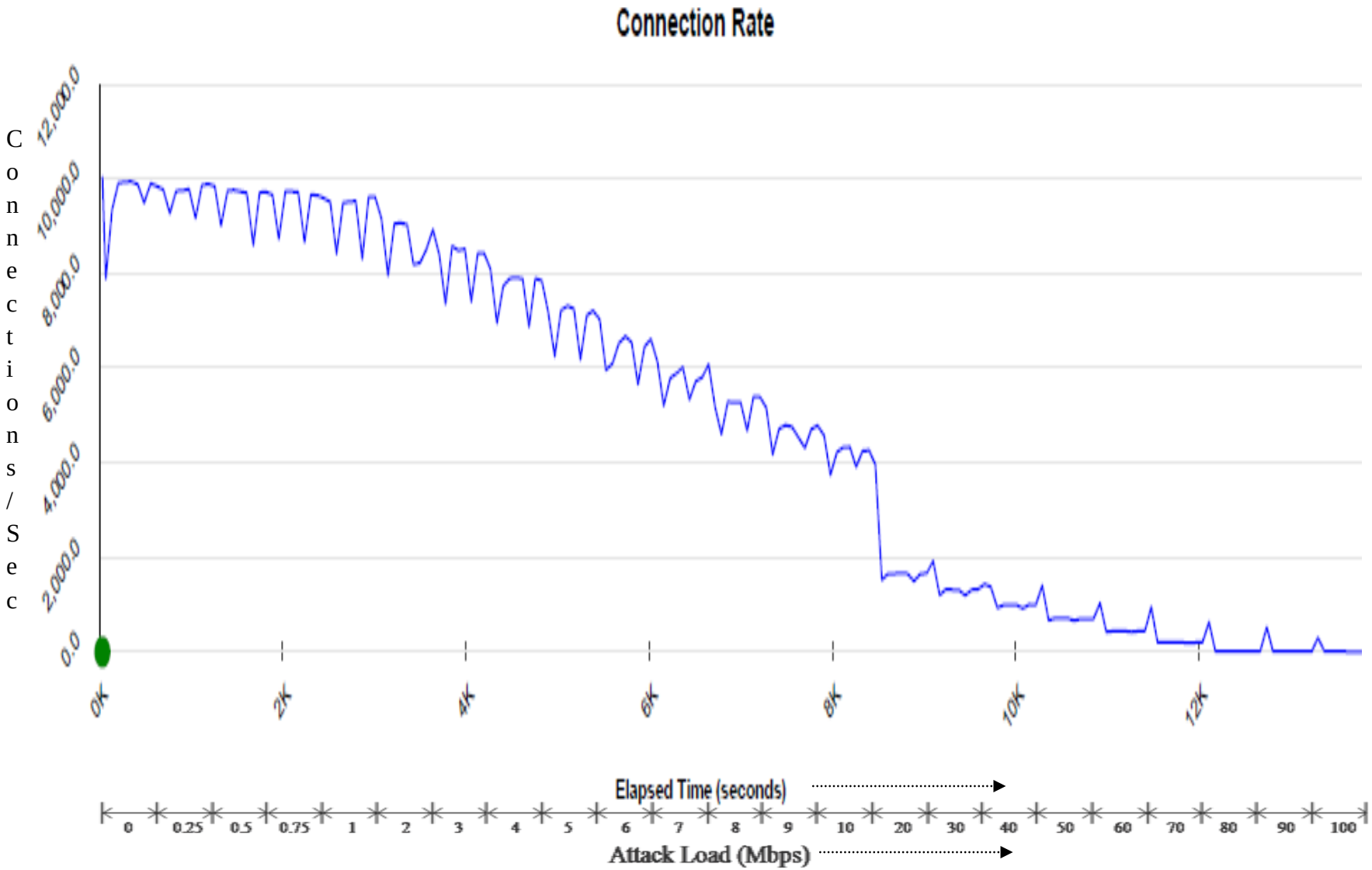

Fig 4.6 Successful client connections per second with SYN cookies protection at the router and SYN attack protect at the server

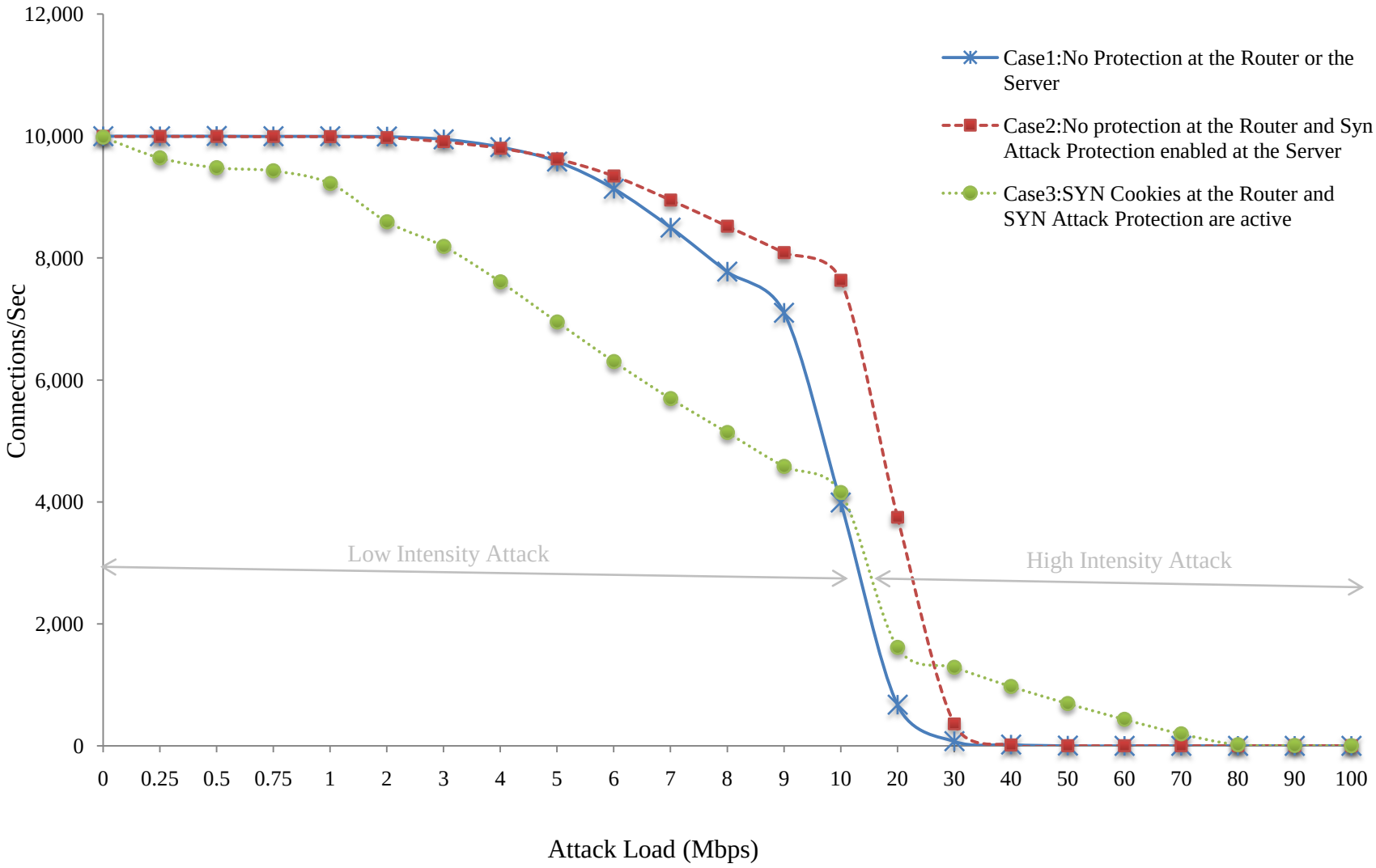

Fig 4.7 SYN cookies performance for low and high intensity TCP SYN attack

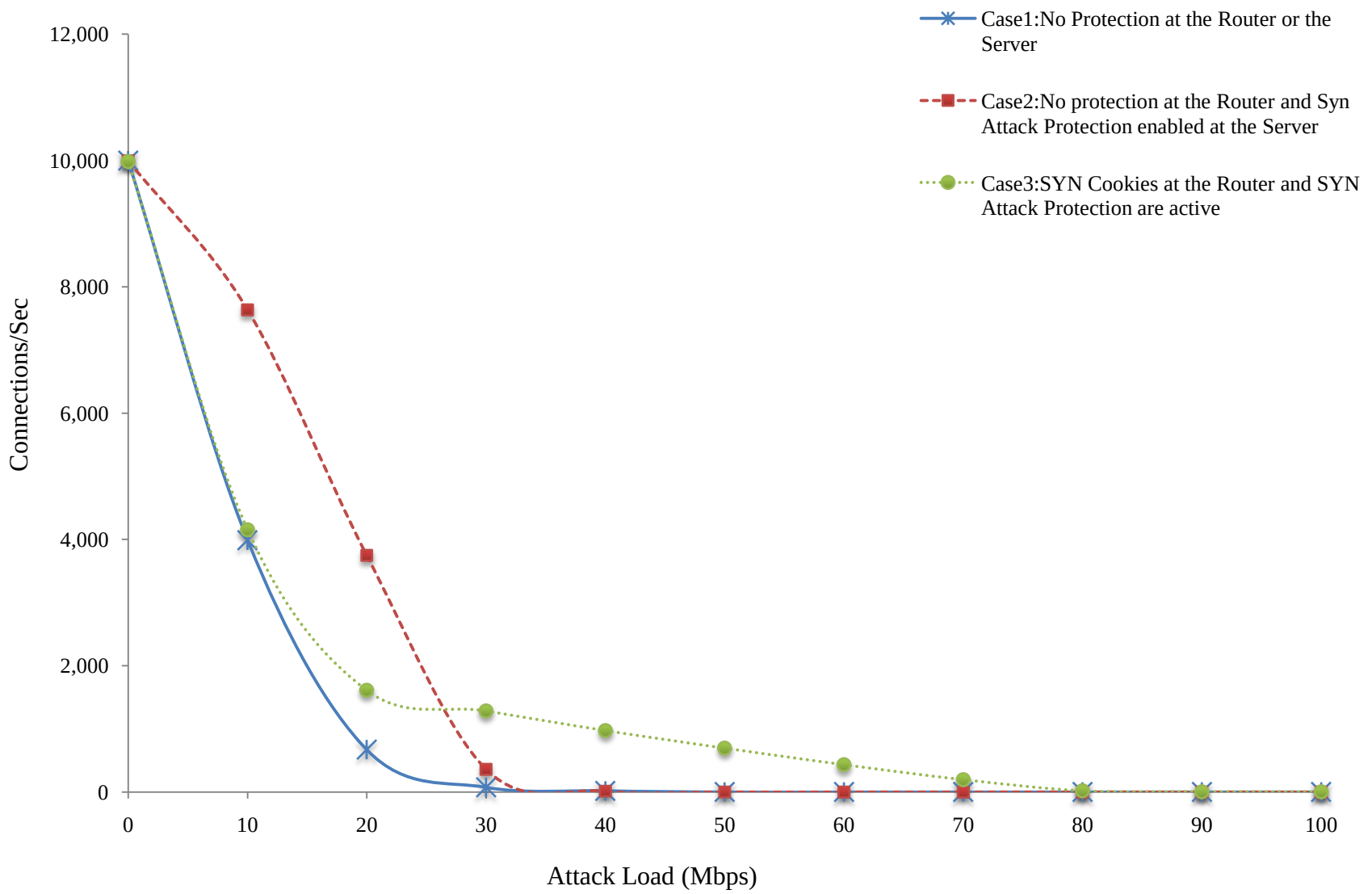

Fig 4.8 SYN cookies performance at equal intervals of attack loads

S.No	Attack Load (Mbps)	Successful Client Connection / Second		
		Case 1	Case 2	Case 3
1	0	9,996	9,995	9,982
2	0.25	9,997	9,994	9,640
3	0.5	9,999	9,993	9,479
4	0.75	9,995	9,991	9,428
5	1	9,995	9,990	9,219
6	2	9,989	9,974	8,591
7	3	9,945	9,906	8,187
8	4	9,817	9,799	7,607
9	5	9,581	9,624	6,949
10	6	9,134	9,344	6,301
11	7	8,498	8,949	5,692
12	8	7,773	8,521	5,135
13	9	7,101	8,086	4,581
14	10	3,988	7,634	4,150
15	20	672	3,747	1,612
16	30	74	359	1,285
17	40	19	14	973
18	50	1	1	694
19	60	0	0	432
20	70	0	0	195
21	80	0	0	18
22	90	0	0	5
23	100	0	0	2

Table 2:SYN Cookies Performace

observed that the successful connections per second are decreasing with the increase of attack load. With SYN cookies protection the connections are able to survive at higher loads of SYN attack compared to the previous two cases. The successful connections per second are depleted to zero at 100Mbps of SYN attack Load.

Fig 4.7 and 4.8 shows the comparison of all the three cases discussed.In case 1 and 2 the router forwards 10,000 SYN packets/sec to the server on first come first serve basis without any filtering. So the router role is the same in case 1 and 2.But the connection rate is sligtly improved in case 2 because of the SYN Attack Protection at the

server when compared to case1.In case 3 the router filters the SYN packets because of SYN cookies protection.In case 3 the router forwards SYN packets to the server until the attack threshold (200 connections/sec) level is reached.After reaching the attach threshold level , the router stops directly forwarding the SYN packets to the server.Instead it will send a SYN cookie back to the client and it forwards the connection request to the server only after receiving the final ACK from the client.Since The attacker will not send the final ACK,The router is never going to forward the connection request sent by the attacker to the server.SYN cookies protection method in the router stops the attack from reaching the server.It is observed from case 3 of the fig 4.7 and 4.8 that the connection rate is reduced at lower intensity of SYN attack when compared to the other two cases.At higher attack intensity the connection rate in case 3 is better than the other two cases.By using SYN cookies protection at the router eventough we lost some connections at the lower SYN attack loads, the connection rate was improved at high intensity of SYN attack loads.

4.3 Introduction to SYN Proxy

SYN Proxy is a very simple concept compared to the other two protection methods we discussed. Instead of storing the state of half open connections in the server and consuming the server resources, we use an intermediate device to deal with half open connections. The connections are forwarded to the server by proxy device only after the clients finish the three-way handshake with the proxy device. New firewalls are coming with this proxy feature inbuilt. After receiving the final ACK of a 3-way handshake from the clients the proxy device form a connection with the server by completing three-way handshake as in the case of SYN cookie shown in Fig 4.5.

4.4 Performance of SYN Proxy

The juniper J4350 router has the capability of SYN proxy protection. Fig 4.9 shows the way juniper router SYN proxy works in case of SYN Flood attack. Initially the proxy device allows the clients to do the three-way handshake directly with the server. After reaching certain attack threshold same thresholds as mentioned earlier in case of SYN cookies in section 4.2 ,SYN proxy kicks in and stops forwarding the SYN packets to the server. The proxy forms a SYN-ACK packet with IP address of server as source address. So the clients are not aware of the proxy device. When the final ACK is received by the proxy device, the three-way handshake is completed between proxy device and the client. The proxy device knows that client is a legitimate one. Following that the proxy completes the three-way handshake with the server for those legitimate clients and the real data transfer between the client and the server is initiated. Unlike SYN cookies, SYN proxy supports all the TCP options and it does not affect the reliability of TCP's Data transfer service. The only drawback of SYN proxy is that it introduces a delay in the

processes of three-way handshake between the client and the server. Since the proxy device is storing the half open connections state it requires significant memory resources.

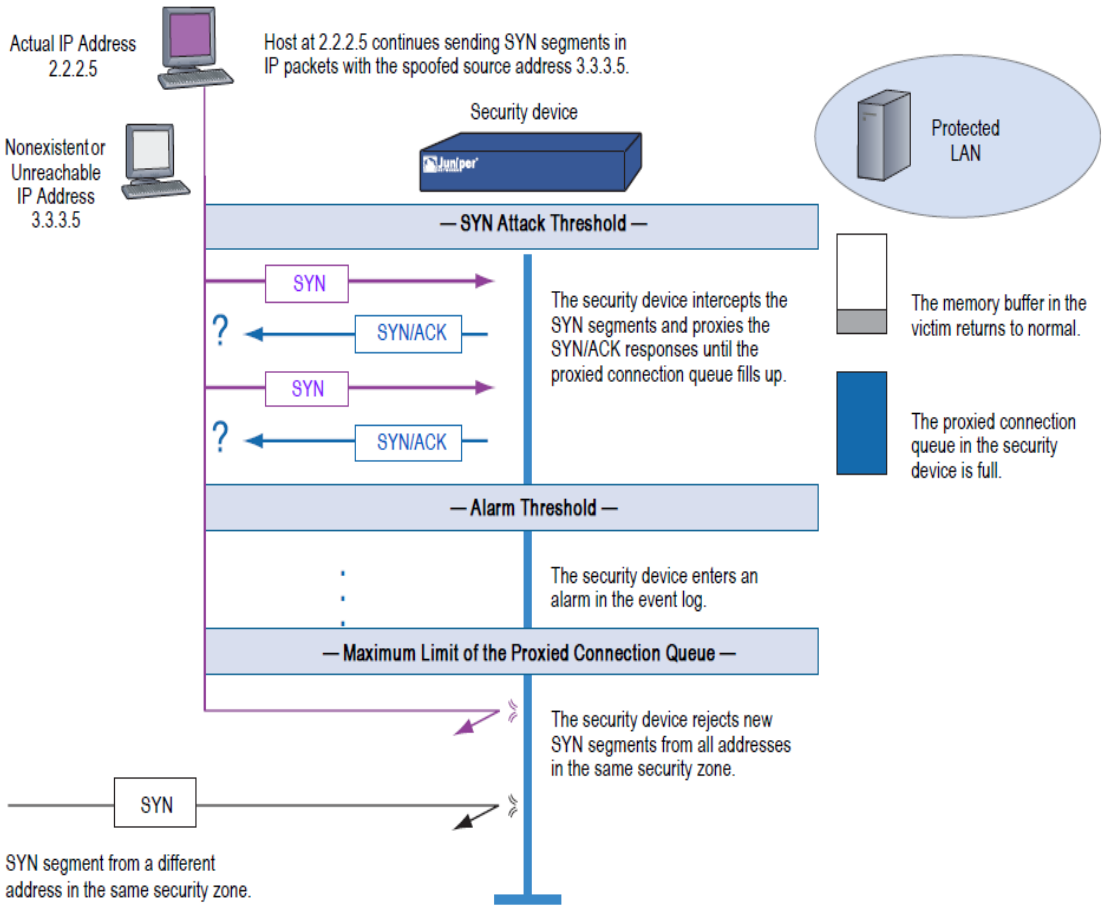

Fig 4.9 SYN Proxy in action [76]

The threshold values we used for the testing are as fallows

Attack threshold = 200 connections per second

Alarm threshold = 512 connections per second

Source threshold = 4000 packets per second

Destination threshold = 4000 packets per second

Timeout = 20 seconds

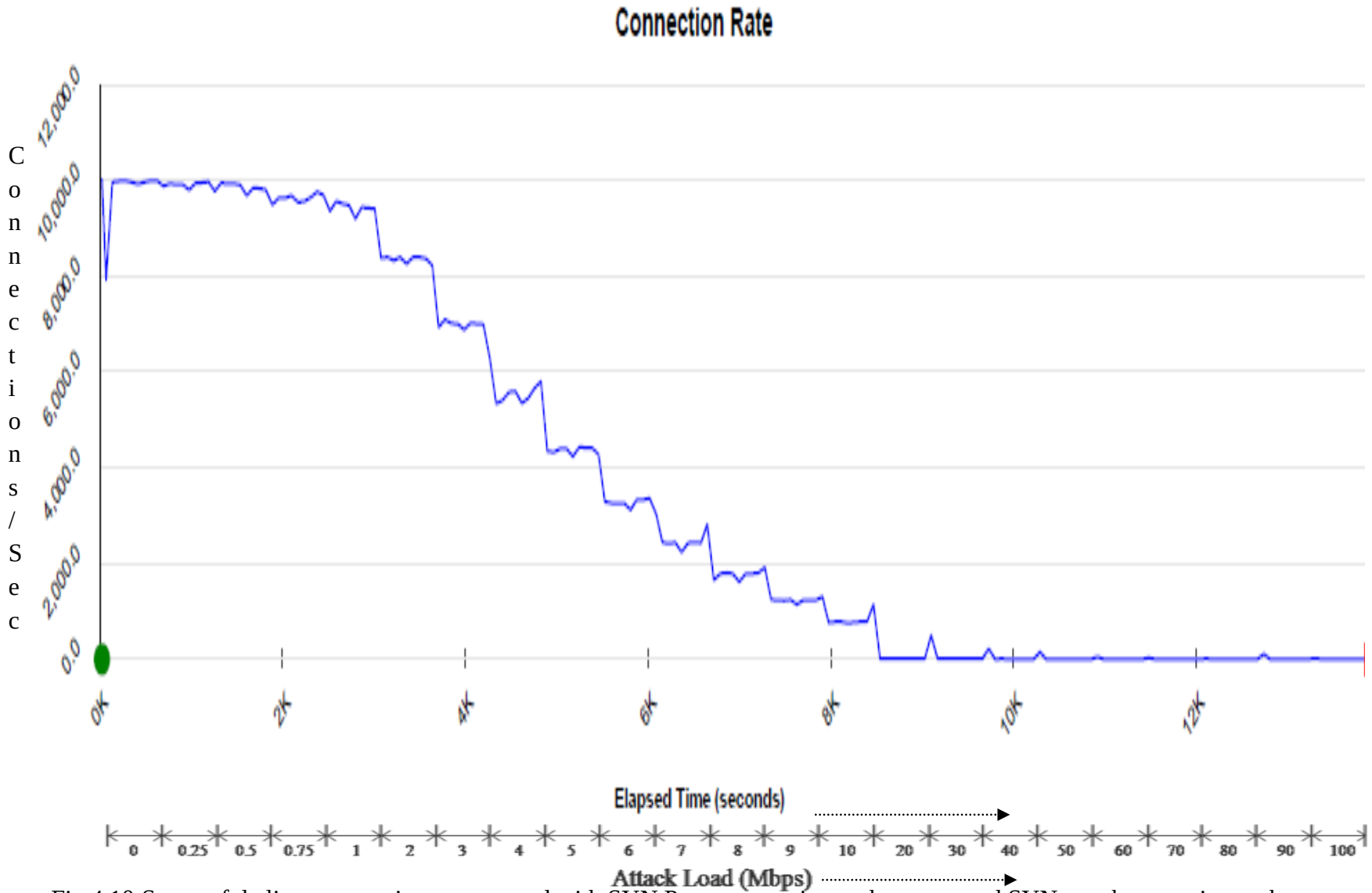

Fig 4.10 Successful client connections per second with SYN Proxy protection at the router and SYN attack protection at the server

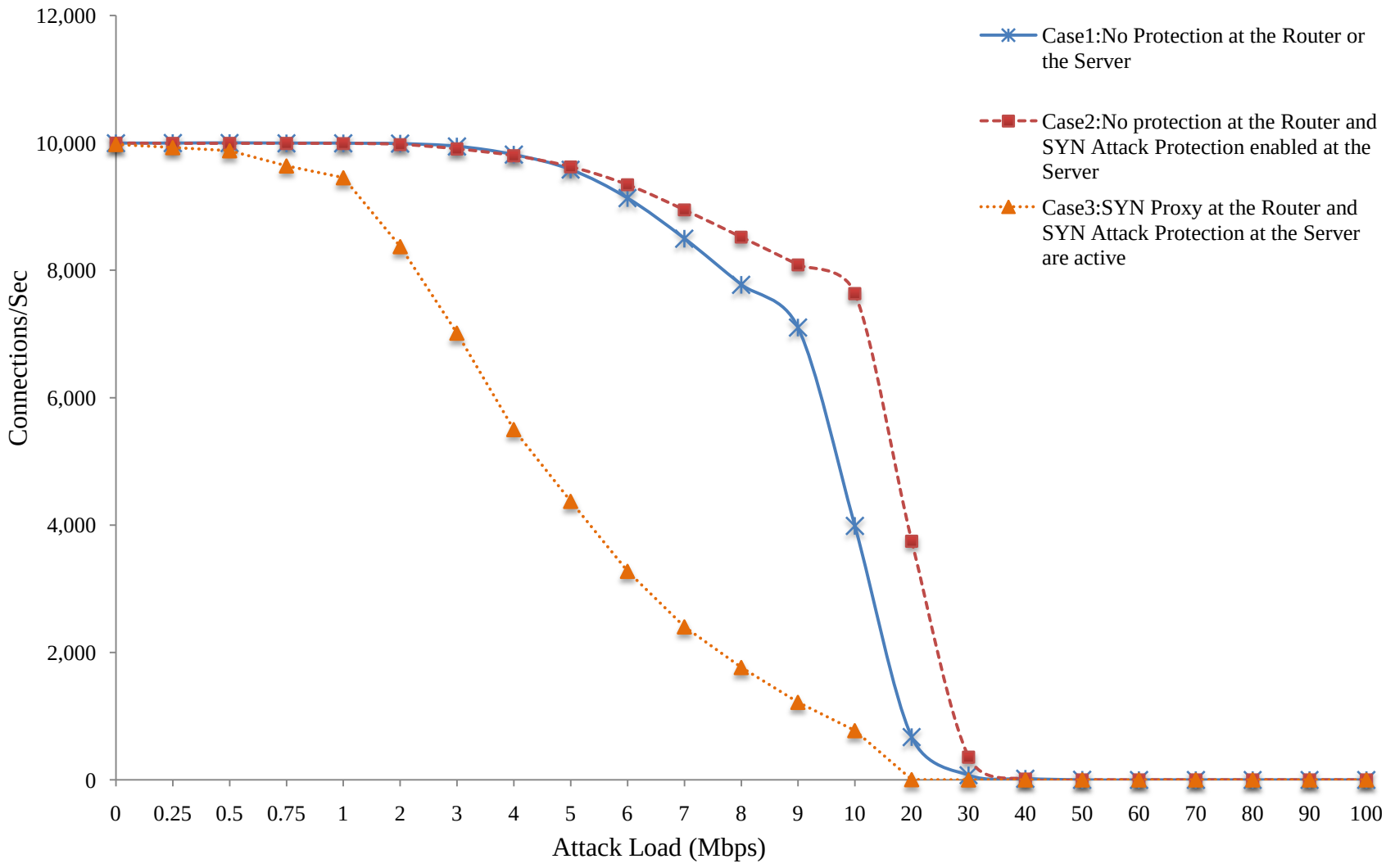

Fig 4.11 SYN Proxy Performance

S.No	Attack Load (Mbps)	Successful Client Connection / Second		
		Case 1	Case 2	Case 3
1	0	9,996	9,995	9,976
2	0.25	9,997	9,994	9,928
3	0.5	9,999	9,993	9,877
4	0.75	9,995	9,991	9,639
5	1	9,995	9,990	9,454
6	2	9,989	9,974	8,371
7	3	9,945	9,906	7,013
8	4	9,817	9,799	5,498
9	5	9,581	9,624	4,374
10	6	9,134	9,344	3,273
11	7	8,498	8,949	2,404
12	8	7,773	8,521	1,765
13	9	7,101	8,086	1,219
14	10	3,988	7,634	772
15	20	672	3,747	6
16	30	74	359	2
17	40	19	14	1
18	50	1	1	1
19	60	0	0	1
20	70	0	0	0
21	80	0	0	0
22	90	0	0	0
23	100	0	0	0

Table 3:SYN Proxy Performance

When the alarm threshold is reached the proxy device notify about the attack in the event log. Since the resources are limited not only in server but also in the proxy device. When the memory in the proxy device is full, the new SYN request caused oldest SYN request in the queue to be -dropped. If we direct SYN requests to proxy device so that the half-open connection queue is occupied with forged SYN packets at certain attack load, the legitimate clients are unable to communicate with the server.

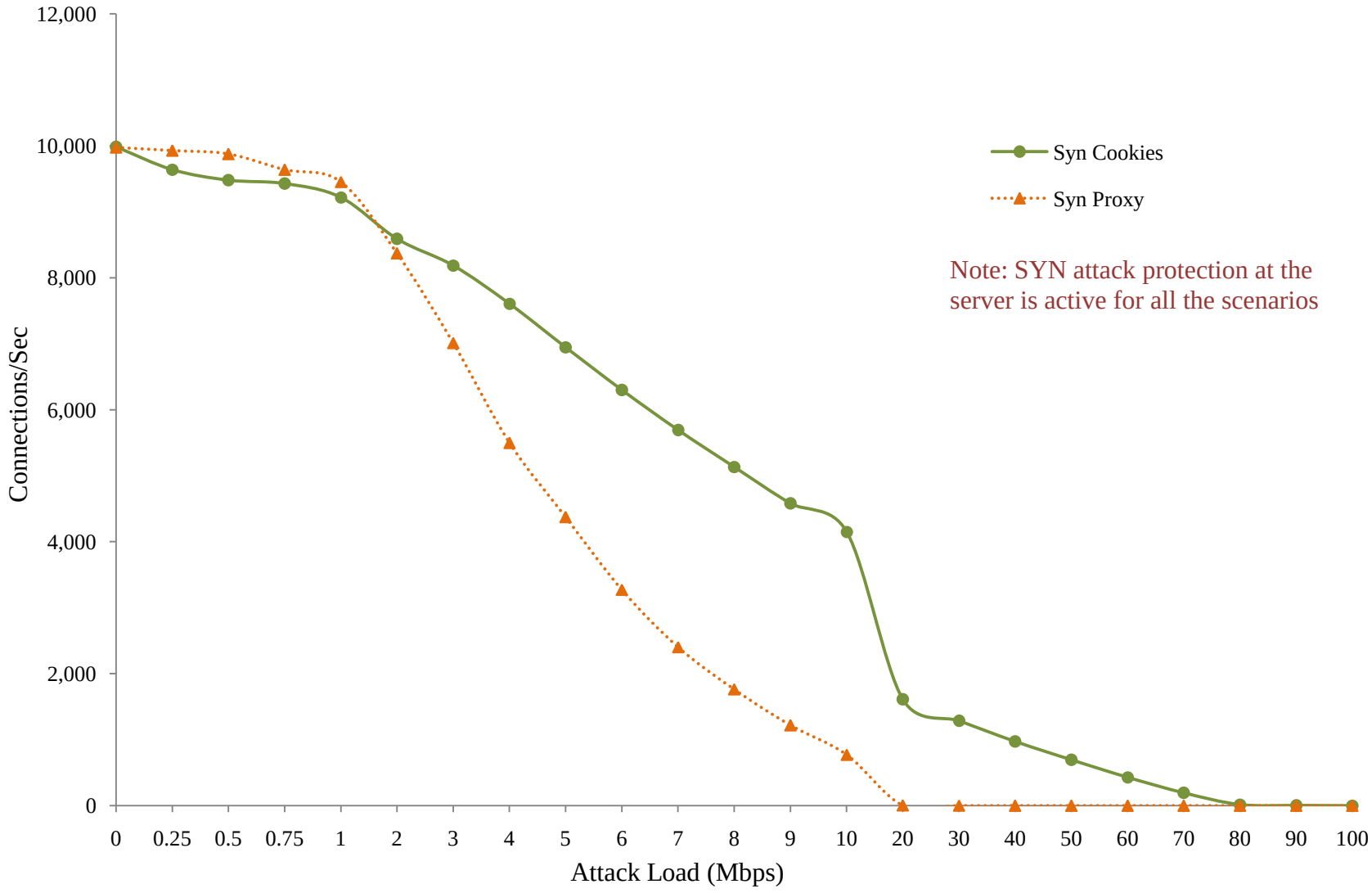

Fig 4.12 Comparison of SYN Proxy with SYN Cookies

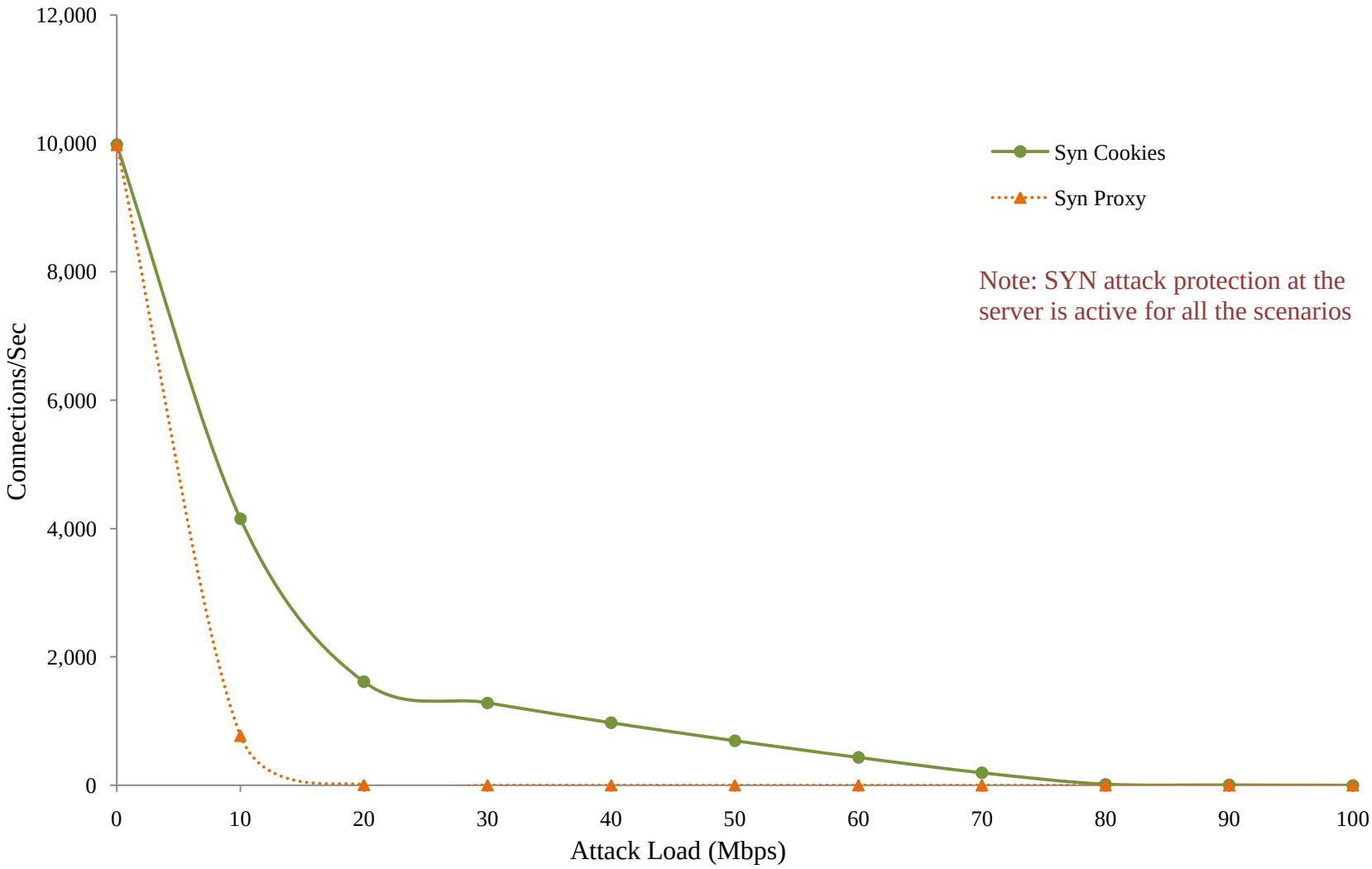

Fig 4.13 Comparison of SYN proxy with SYN cookies at equal intervals of attack loads

Fig 4.11 shows the successful legitimate client connections per second for each attack load from 0 – 100Mbps when SYN proxy is enabled. It is observed that the legitimate connections are depleted to zero at 20Mbps of attack load. SYN proxy method of protection is found to be not effective at higher loads of attack traffic.

Performance of SYN proxy comparison with the first two configurations mentioned in SYN cookie performance is shown in the fig 4.12 and 4.13. It is observed that the good connection rate is decreased under SYN attack when the SYN proxy protection method is enabled in the router. The SYN proxy protection mechanism is only effective in stopping the attack traffic from reaching the server and prevents possible intrusion. The SYN proxy method protects the resources of the server from SYN flood attack but the connection rate under attack is reduced significantly. When we compare SYN proxy with SYN cookies in fig 4.13, the number of good connection rate is significantly lower in case of SYN proxy. SYN proxy seems to require more router hardware capability than SYN cookies to protect the clients from denial of service at any particular attack load.

4.5 Attack on Edge Router connected to Server

The result shown in fig 4.14 is from an interesting scenario. Recently attackers are using more sophisticated techniques to attack web servers. In recent days, servers are usually equipped with hardware and software capabilities to defend itself from security attacks, and network administrators use additional devices like intrusion prevention system, router's firewall and load balancers to guard the high profile servers. In general, servers are positioned behind network security devices. It is possible for an attacker to deny services to regular clients by directing the attack to network devices like routers behind which the server is usually located. In order to understand the effect of attacking a router itself instead of the server connected to the router, we considered following configurations -

1. No protection at the Router
2. With SYN cookies protection at the Router
3. With SYN proxy protection at the Router

In these experiments, the SYN attack protection feature of the server is always enabled for all the attack scenarios mentioned in this section.

Fig 4.14 shows the successful connections when there is no SYN flood protection in the router and the SYN attack is directed to the router. It is observed from fig 4.15 that the client connection rate drops more quickly (Fig 4.4) when compared with the situation where the attack is directed to server. Fig 4.16 shows the results when the SYN attack directed to the router instead of the server and SYN cookies protection is enabled in the router. The successful connections per second are almost the same (Fig 4.17) as when we direct the SYN attack to server. That means even if you use some IPS system between

the server and the router for protection we can eliminate the IPS protection by directing the attack to the router instead of server.

Fig 4.18 shows the SYN proxy performance when the attack load is directed to the Proxy device itself instead of the server. It is observed that this attack scenario is causing the worse performance than any other case mentioned in this thesis (Fig 4.20). The successful connection rate dropped to zero at 5Mbps of SYN attack traffic.

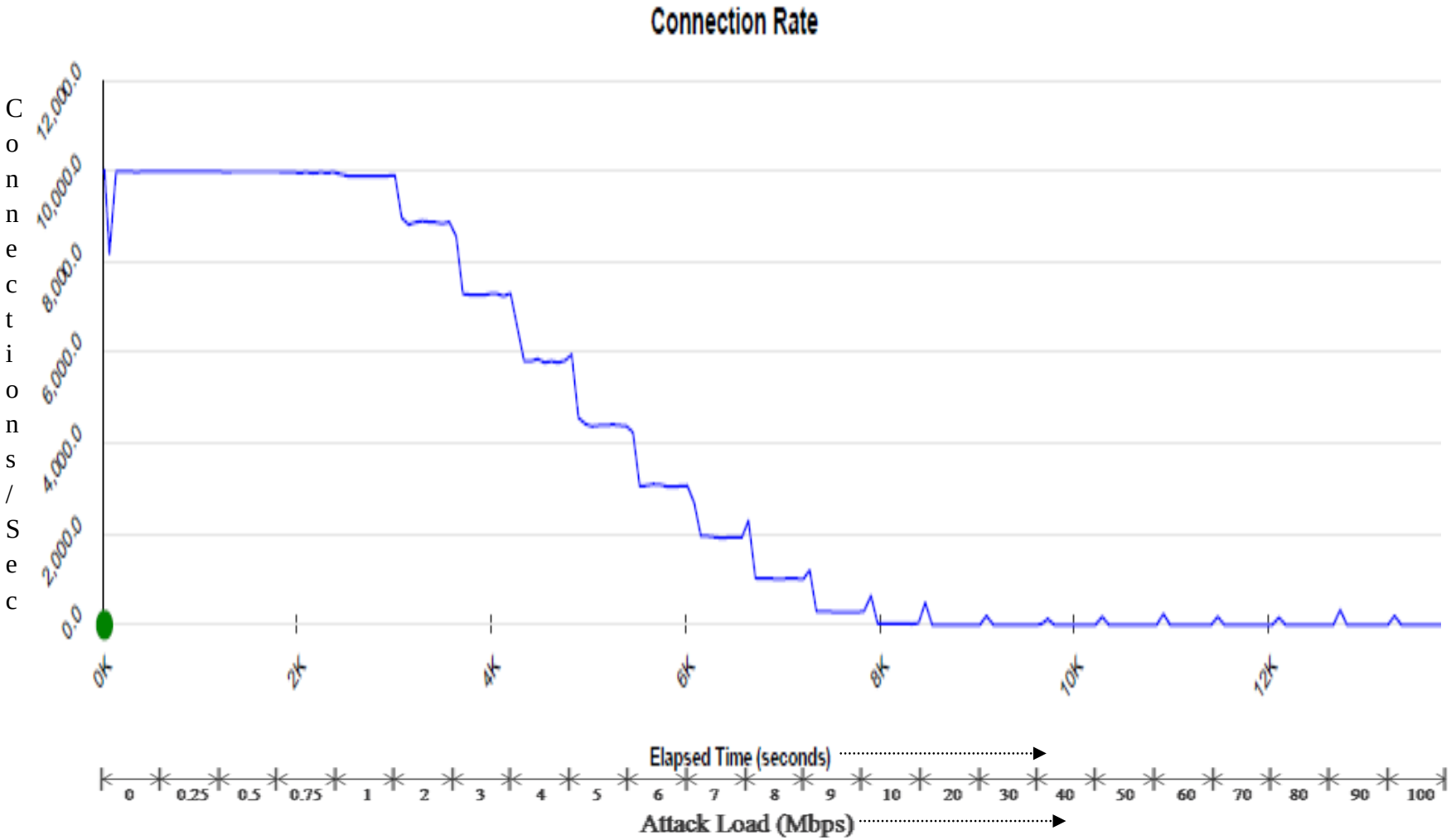

Fig 4.14 Successful Client connections/sec without any SYN flood protection at the router under SYN Attack (Attack directed to Router)

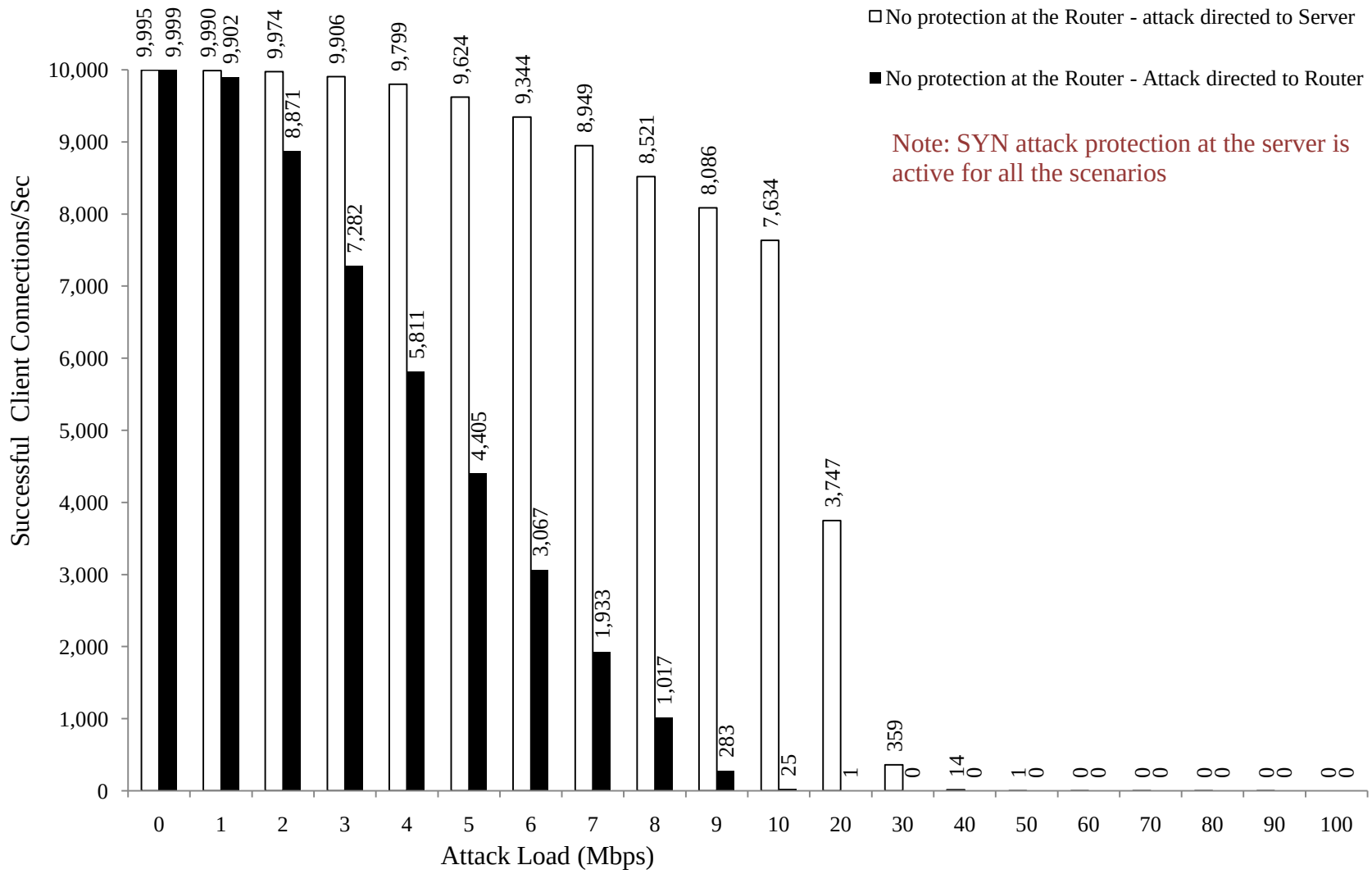

Fig 4.15 Attack directed to Server vs. Attack directed to Router when no protection enabled in the Router

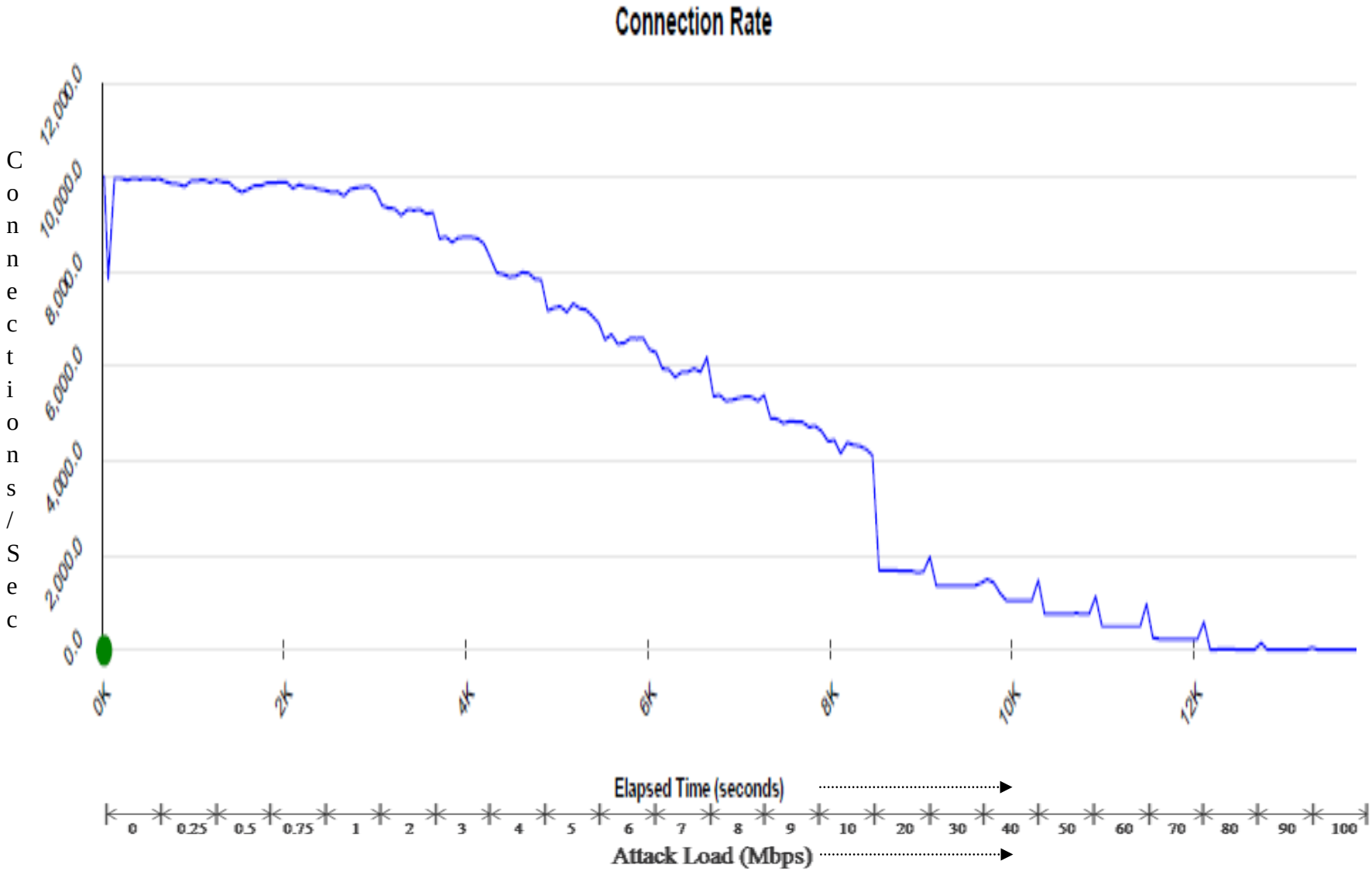

Fig 4.16 Successful client connections per second with SYN cookies protection at the router(Attack Directed to Router)

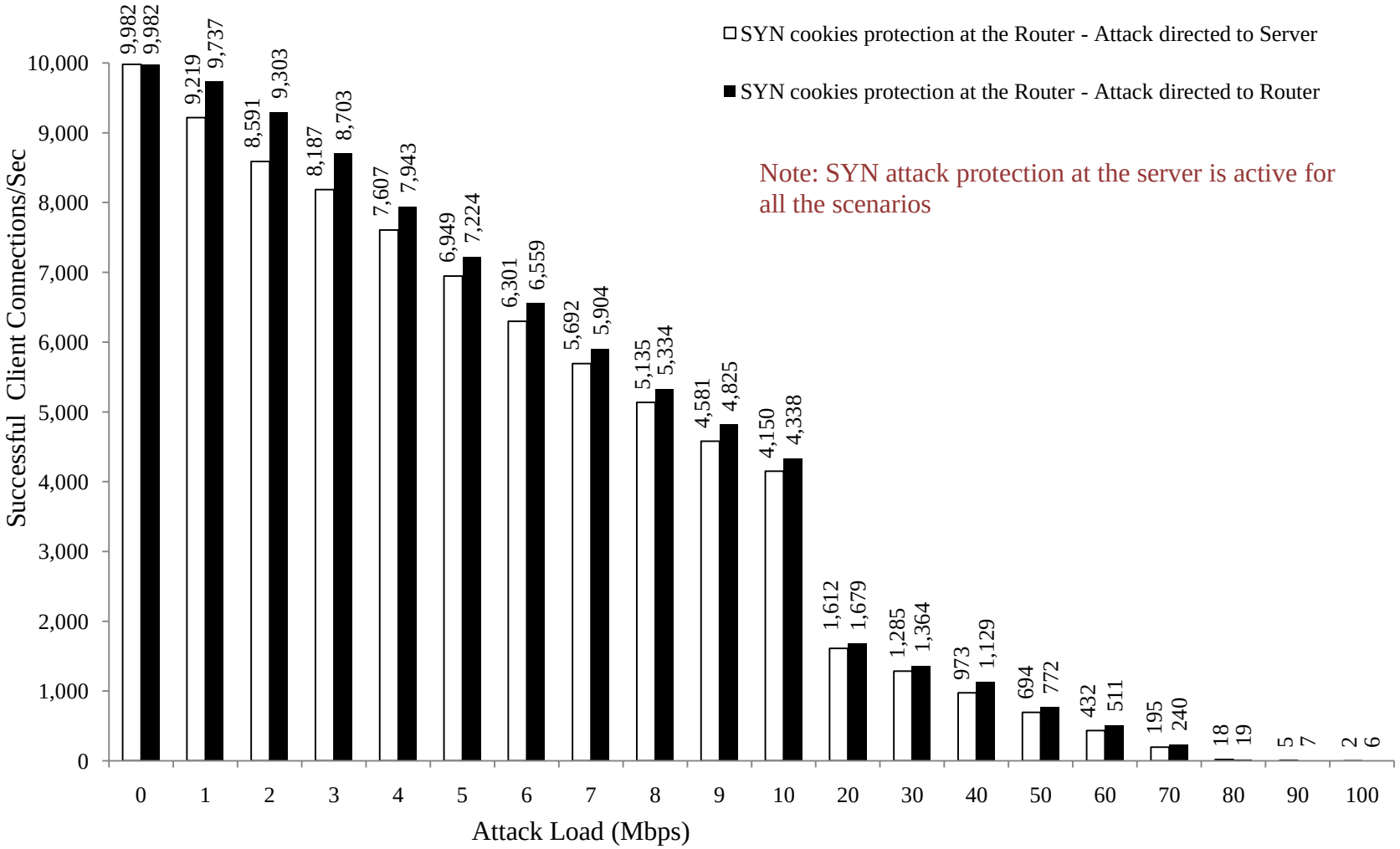

Fig 4.17 Attack directed to Server vs. Attack directed to Router when SYN cookies protection enabled at the Router

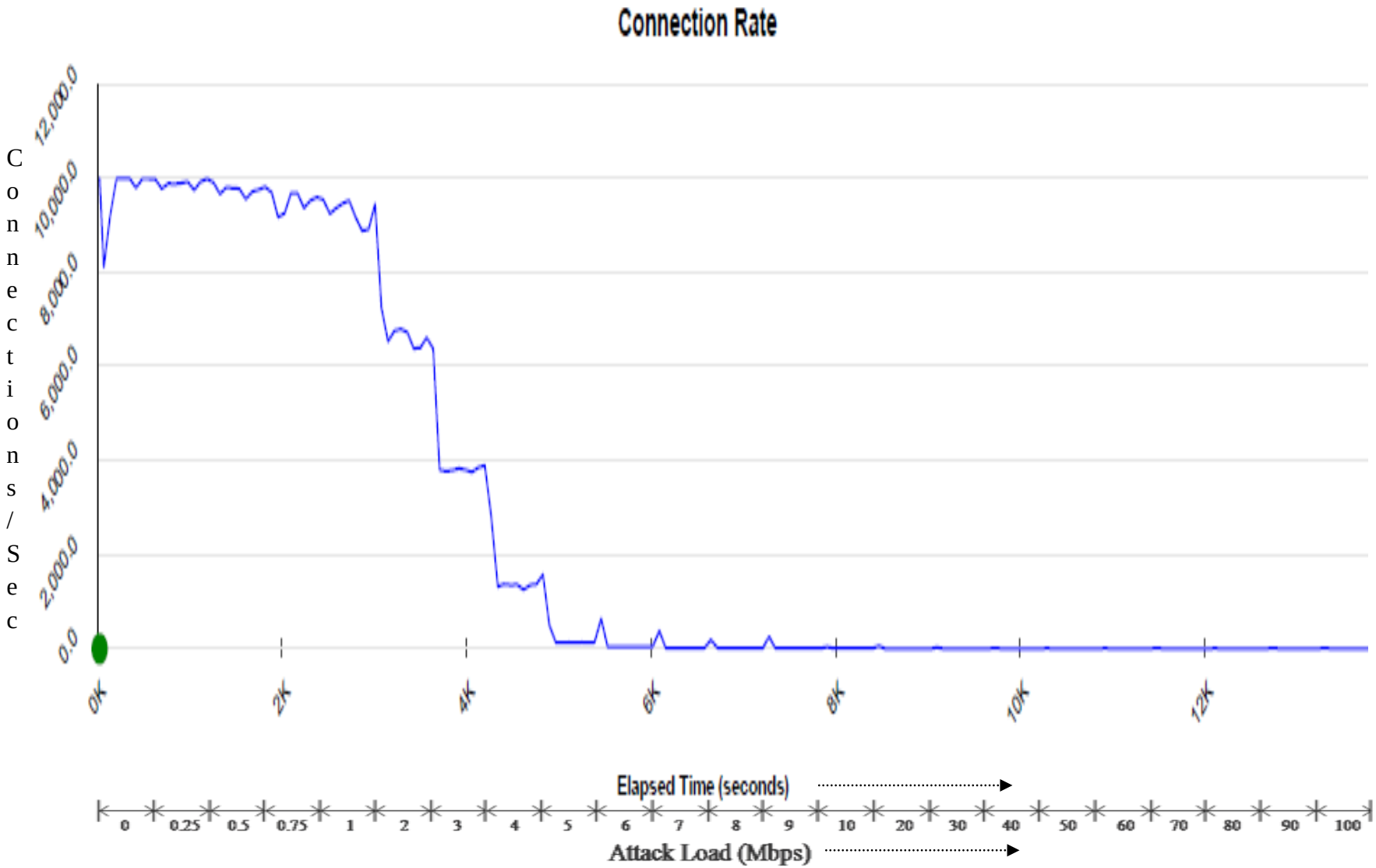

Fig 4.18 Successful client connections per second with SYN Proxy protection at the router (Attack Directed to router)

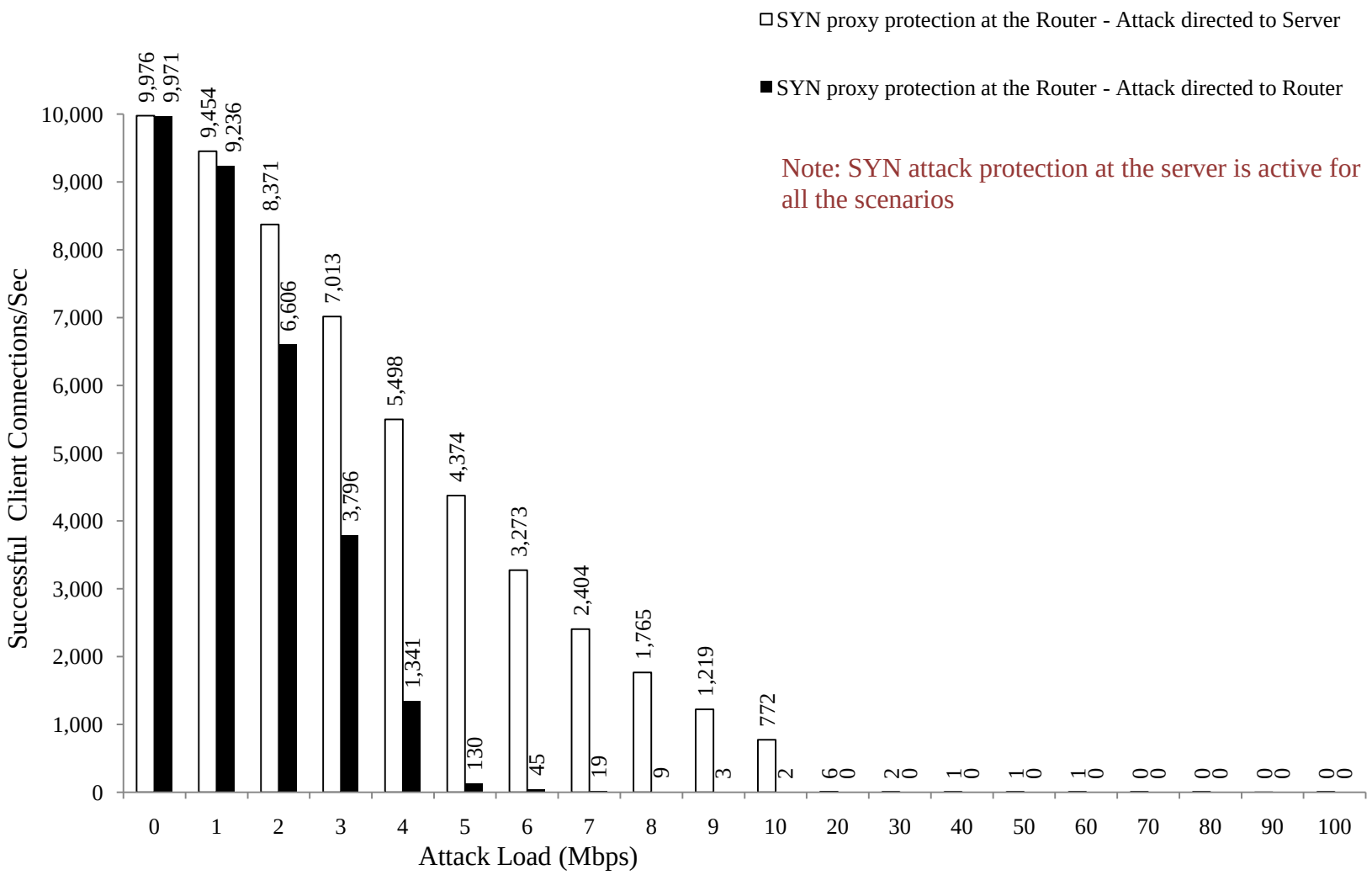

Fig 4.19 Attack directed to Server vs. Attack directed to Router when SYN proxy protection enabled at the Router

4.6 Comparison of all Scenarios and Protection Methods

Comparing all the scenarios discussed so far in one place provides a clear picture of effectiveness of different methods of protection as shown in fig 4.20 and 4.21. The worst case among all of them is when attack directed to router with SYN proxy protection enabled. Performance evaluation results shows SYN cookie is the best method of protection at higher attack intensity. The client connections successful rate with SYN cookies protection at the router is almost the same weather the attack is directed to the router or the server. It is also observed that the average number of successful client connections behavior with attack load is almost the same in both the cases with attack directed to server when SYN proxy protection enabled and Attack directed to router without any protection in the router. So depending on the intensity of the SYN attack and the business needs such as number of clients to support, we can choose the best suitable protection mechanism from the graph shown in fig 4.20 and 4.21.

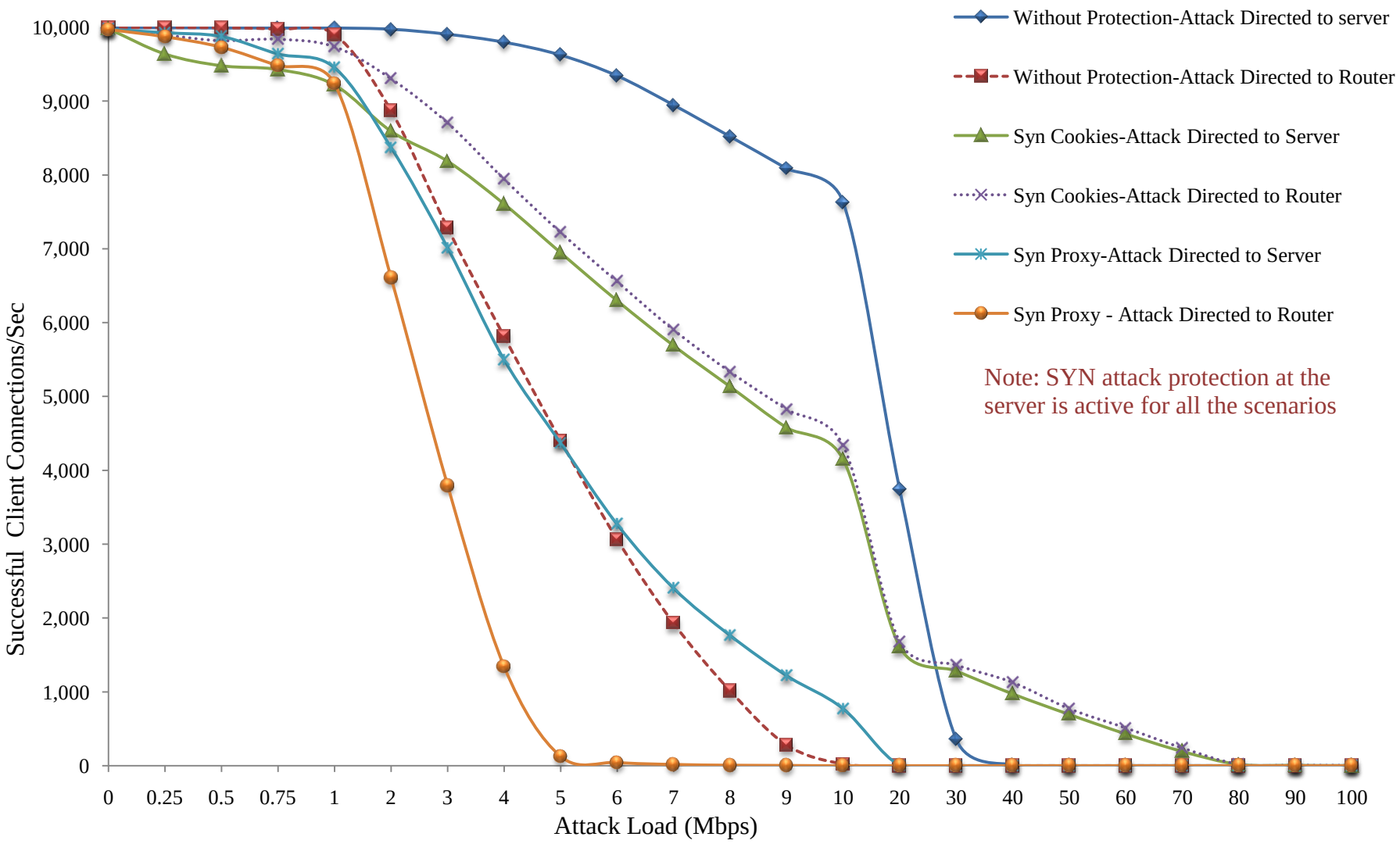

Fig 4.20 Comparison of average number of successful connections in different scenarios with protection at the Server

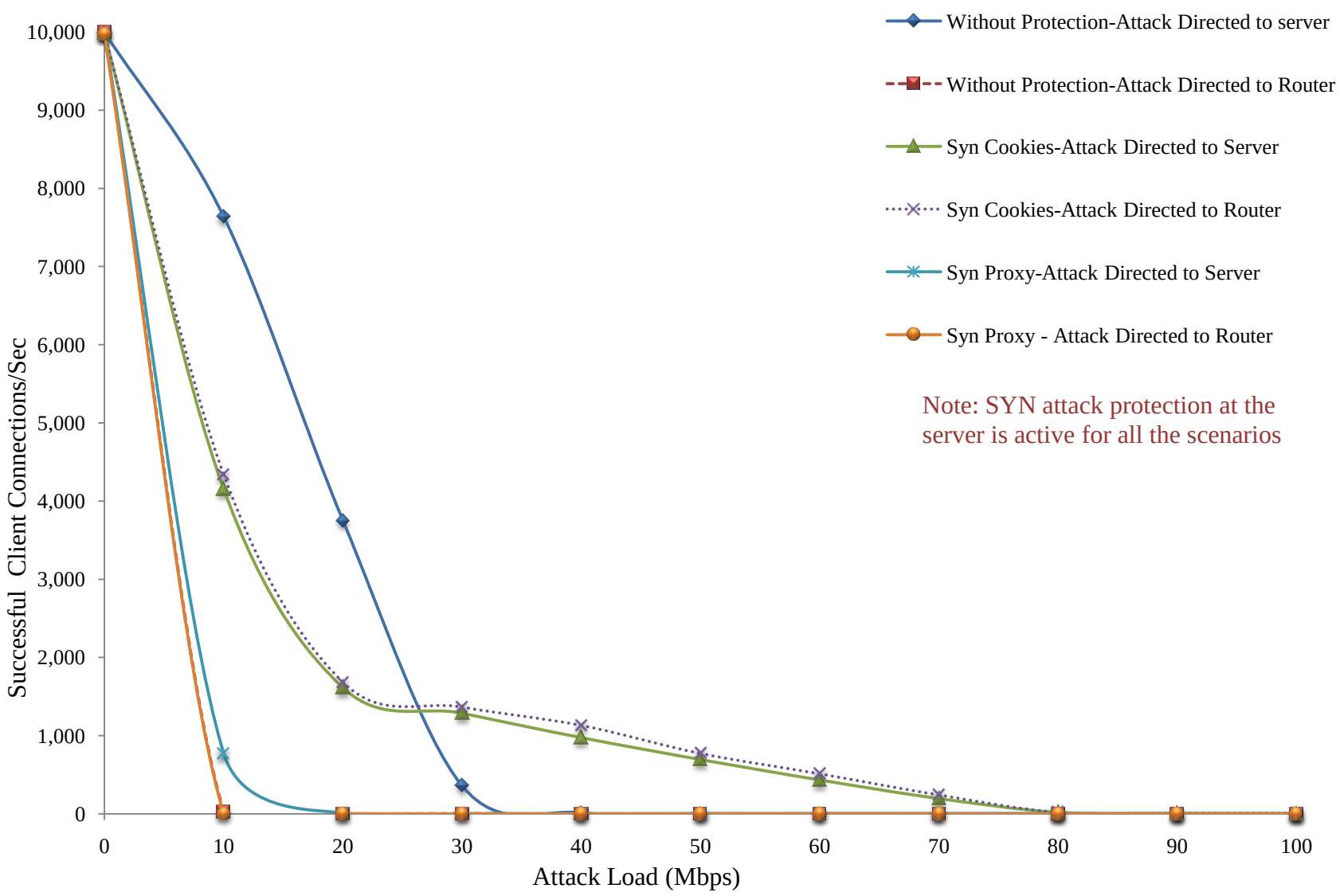

Fig 4.21 Comparison of successful connection rate under equal interval of attack loads with SYN attack protection at the Server

S No.	Attack Load (Mbps)	Successful Client Connections/Sec (SYN attack protection in the server is active)					
		No Flood Protection in the Router		Flood Protection Enabled in the Router			
				SYN Cookies		SYN Proxy	
		Attack Directed to Server	Attack Directed to Router	Attack Directed to Server	Attack Directed to Router	Attack Directed to Server	Attack Directed to Router
1	0	9,995	9,999	9,982	9,982	9,976	9,971
2	0.25	9,994	9,997	9,640	9,897	9,928	9,872
3	0.5	9,993	9,994	9,479	9,820	9,877	9,729
4	0.75	9,991	9,975	9,428	9,840	9,639	9,481
5	1	9,990	9,902	9,219	9,737	9,454	9,236
6	2	9,974	8,871	8,591	9,303	8,371	6,606
7	3	9,906	7,282	8,187	8,703	7,013	3,796
8	4	9,799	5,811	7,607	7,943	5,498	1,341
9	5	9,624	4,405	6,949	7,224	4,374	130
10	6	9,344	3,067	6,301	6,559	3,273	45
11	7	8,949	1,933	5,692	5,904	2,404	19
12	8	8,521	1,017	5,135	5,334	1,765	9
13	9	8,086	283	4,581	4,825	1,219	3
14	10	7,634	25	4,150	4,338	772	2
15	20	3,747	1	1,612	1,679	6	0
16	30	359	0	1,285	1,364	2	0
17	40	14	0	973	1,129	1	0
18	50	1	0	694	772	1	0
19	60	0	0	432	511	1	0
20	70	0	0	195	240	0	0
21	80	0	0	18	19	0	0
22	90	0	0	5	7	0	0
23	100	0	0	2	6	0	0

Table 4: Successful Client Connections Vs Attack Load

CHAPTER V

CONCLUSIONS AND FUTURE WORK

Denial of service attacks are increasing in the number of occurrences and sophistication. In this thesis, the popular Windows server 2003 was evaluated under Ping based and TCP based Denial of Service attacks to evaluate its capabilities in protecting against such attacks. It was found that the Windows 2003-SP2 capability of the Windows server could detect Ping-based Distributed Denial of Service Attacks by measuring the rate of arrival of ICMP packets from different IP addresses. It was found that if the number of incoming ping request packets exceeded 250 per second then it started dropping the excessive packets and didn't send any response (ICMP echo replies) back for the excessive packets. It was found that even the powerful quad-core Xeon-processor based windows server could suffer a catastrophic slowdown under a distributed Ping-based DDoS attack, if the Windows server was not configured correctly, especially when ICMP packets were allowed by the Windows firewall. When ICMP packets were blocked, the windows server 2003 managed to survive the same ping based DDoS attacks, despite losing a maximum of 25% of its processing capability in the fast Ethernet environment. Since it is not always practical to block ICMP messages for diagnostic reasons, it is possible for the server to be attacked by Ping-based DDoS floods, resulting in failure or slow down of even a powerful Xeon based Quad core processor.

TCP flood based denial of service attacks are being used more often by the attackers in recent years to cause denial of Internet services to legitimate clients. In this thesis, we conducted experiments to evaluate different protection mechanisms commonly used by the servers and routers to prevent TCP SYN attacks. Protection methods that were evaluated in this thesis were SYN Cache, SYN Cookies and SYN proxy protection methods.

It was observed that with SYN attack protection (a type of SYN cache protection) feature when enabled in windows server 2003, the legitimate client connections were improved significantly at especially for low intensity of TCP-SYN flood attack. The client-connections were found to fall rapidly at higher loads of SYN attack even when SYN attack protection was enabled in the server. For attack load in the range 0 to 30 Mbps SYN attack protection at windows server 2003 is good enough for protecting against TCP SYN flood attacks. However, TCP SYN flood attacks usually involve higher attack loads (greater than 30 Mbps).

In order to investigate protection methods against higher intensity of TCP SYN attacks, two protection mechanisms, namely SYN cookies and proxy methods of protection were evaluated. These protection mechanisms were available in the Juniper router J4350 that was connected to the Windows 2003 server under test. The SYN cookies as well as the SYN proxy protection methods were designed to stop the attack from reaching the server and prevent resource consumption and possible intrusions at the server. The major advantage of SYN cookies method of protection over SYN cache is that unlike SYN cache method of protection, SYN cookie method has better performance under higher loads of SYN attack traffic. SYN proxy method of protection is also capable

of protecting the server from possible intrusions, however it is only effective against very low intensity SYN attack load (< 10Mbps). It is the least effective method of protection against high intensity TCP SYN attack when compared to the other two methods of protection. Based on our experiments in this thesis, the best protection against a low intensity TCP SYN based DDoS attack (< 30Mbps) is offered by the Window's 2003 built-in SYN attack protection mechanism, and for high intensity TCP SYN attack (> 30Mbps), the SYN cookie method of protection is found to be the most effective mechanism. For the given network used in this thesis, with the Juniper router and attached Windows server 2003, the best recommendation to maximize the protection against the TCP SYN attack, is to enable TCP SYN cookie protection at the router, and also enable the built-in TCP SYN attack protection at the Windows Server 2003 at the same time. Furthermore, the threshold values in the router for TCP SYN cookie protection can be adjusted such that the SYN cookie protection becomes active only under high intensity attacks, and for the low intensity of TCP SYN attack, only the server's built-in protection is utilized. None of these protection methods are standardized by IETF as of yet, however, we hope that continued testing and evaluation along the line of work presented in this thesis will be helpful in standardization efforts.

For future work, it will be useful to study the effects of TCP SYN flood attack on pre-established TCP client connections and its effect on the connections' throughput. One may also consider, studying the detailed effect of different threshold values on the performance of different protection mechanisms; and extending the performance evaluation to different platforms.

REFERENCES

[1] The World Fact Book, Central Intelligence agency (CIA), 2008, ISSN 1553-8133. https://www.cia.gov/library/publications/the-world-factbook/geos/us.html

[2] "Ascending the Managed Services Value Chain," Cisco Visual Networking Index Forecast, 2007-2012, June 2008. http://www.cisco.com/en/US/solutions/collateral/ns341/ns524/ns546/white_paper_c11-554084.html

[3] M. Mellia, I. Stoics, and H. Zhang, "TCP Model for Short Lived Flows," in Proc. IEEE Communication Letters, Feb. 2002. http://www.tlc-networks.polito.it/mellia/papers/letteraTCP.pdf

[4] Lee Garber," Denial-of-Service Attacks Rip the Internet," Computer, v.33 n.4, p.12-17, April 2000.

[5] D. Moore, G. Voelker and S. Savage, "Inferring Internet Denial of Service Activity," Proceedings of USENIX Security Symposium 2001, August 2001.

[6] H. Wang, D. Zhang, K. G. Shin, "Change-Point Monitoring for the Detection of DoS Attacks," Dependable and Secure Computing, IEEE Transactions on Volume 1, Issue 4, Dec 2004, Page(s):193 – 208.

[7] Y. Ohsita, S. Ata, M. Murata, "Detecting Distributed Denial-of-Service Attacks by analyzing TCP SYN packets statistically," Global Telecommunications Conference, GLOBECOM '04, IEEE Volume 4, 29 Nov. - 3 Dec. 2004, Page(s): 2043 - 2049 Vol.4.

[8] Tian, H.-T., Huang, L.-S., Lei, Y.-F., and Chen, G.-L."A new scheme for IP trace back under DOS attack," Proc. 4th Int. Conf. on Parallel and Distributed Computing, Applications and Technologies, (PDCAT'2003), 27–29 Aug. 2003, pp. 189–193.

[9] Rizvi, B., and Fernandez – Gaucherand, "Effectiveness of advanced and authenticated packet marking scheme for trace back of denial of service attacks," Proc. Int. Conf. on Information Technology: Coding and Computing (ITCC 2004) vol. 2, pp. 111–115.

[10] Rocky K. C. Chang, "Defending against Flooding-Based Distributed Denial-of-Service Attacks: A Tutorial," IEEE Communications Magazine, Oct., 2002.

[11] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," ACM SIGCOMM Computer Communications Review, 34(2):39—54, April 2004.

[12] F.A. El-Moussa, N. Linge, and M. Hope, "Active router approach to defeating denial-of-service attacks in networks," Communications, IET Volume 1, Issue 1, February 2007 Page(s):55 – 63.

[13] Yau, D.K.Y., Lui, J.C.S., Feng, L., and Yeung, Y, "Defending against distributed denial of service attacks with max–min fair server-centric router throttles," IEEE/ACM Trans. Netw., 2005, 13, (1), pp. 29–42.

[14] Xu. Y, "Statistically countering denial of service attacks," Proc. IEEE Int. Conf. on Communications (ICC 2005), 16–20 May 2005, vol. 2, pp. 844–849.

[15] "Transmission Control Protocol," RFC 793, Information Science Institute, September 1981, University of southern California.

[16] Christoph L. Schuba , Ivan V. Krsul , Markus G. Kuhn , Eugene H. spafford , Aurobindo Sundaram ,and Diego Zamboni, "Analysis of a Denial of Service Attack on TCP," Proceedings of the 1997 IEEE Symposium on Security and Privacy, p.208, May 04-07, 1997.

[17] W. Richard Stevens," TCP/IP illustrated volume 1: The Protocols," ISBN 0201633469, October 2002, Pages 229-260.

[18] J.Postel, "Internet Control Message Protocol," DARPA Internet program protocol specifications, RFC 792, September 1981.

[19] T.Berners-Lee, R.Fielding, and H.Frystyk," Hypertext Transfer Protocol—HTTP/1.0, "RFC1945, May 1996.

[20] Nail Mansfield," Practical TCP/IP: Designing, Using, and Troubleshooting TCP/IP Networks on LINUX and WINDOWS," ISBN: 0201750783, Addison-Wesley Publication 2003, Pages 27-82.

[21] M.Williams,"Ebay,amazon,buy.com hit by attacks, 02/09/00.IDG News Service," http://www.nwfusion.com/news/2000/0209attack.html

[22] “Cyber Biltz Hits U.S., Korea: Simple Attack on Government, Business Exposes Vulnerability, “The Wall Street Journal, July 2009.
http://online.wsj.com/article/SB124701806176209691.html

[23] Shigang Chen, Yibei Ling, Randy Chow and Ye Xia” AID: A global anti-DOS service,” Computer Networks 51, May 2007.

[24] D.Karig, and R. Lee,” Remote Denial of Service Attacks and counter measures,” Department of Electrical Engineering, Princeton University, Technical Report CE-L 2001 - 002, October 2001.

[25] Christos Douligeris, and Aikaterini Mitrokotsa,” DDoS attacks and Defense Mechanisms: Classification and state-of-art, “Department of informatics, University of Piraeus, 13 October 2003.

[26] Dennis Distler,” Performing Egress Filtering,” SANS institute infosec reading room, August 2008.

[27] P.Ferguson, and D.Senie,” Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing,” RFC 2827, 2001.

[28] Xianjun Geng and Andrew B.Whinston,” Defeating Distributed Denial of Service Attacks,” IEEE IT Professional, 2000.

[29] R.R.Talpade, G. Kim, and S.Khurana,” NOMAD: Traffic-based network monitoring framework for anomaly detection,” Proceedings of the fourth IEEE symposium on computers and communications, 1999, pp. 442-452.

[30] Nathalie Weiler,” Honeypots for Distributed Denial of Service Attacks,” Computer Engineering and networks Laboratory, Swiss Federal Institute of Technology, Proceedings of the eleventh IEEE international workshops on enabling technologies, 2002.

[31] Dr. Sanjeev Kumar,” Ping Attack –how bad is it?,” IEEE Computers and Security, November 2005

[32] J.Xu and W.Lee,” Sustaining availability of web services under distributed denial of service attacks,” IEEE transactions on computers, Vol. 52, Feb., 2003.

[33] Rocky K.C.Chang,” Defending against flood based distributed denial of service attack –A tutorial,” IEEE communication Magazine, Oct., 2002.

[34] D. Sterne, K Djahandari, B.Wilson, B.Babson, D. Schnackenberg, H.Holliday and T.Reid,” Autonomic response to distributed denial-of-service attacks,” RAID, Davis, California, 134-149, October 2001.

[35] Vyas Sekar, Nick G. Duffield, Oliver Spatscheck, Jacobus E.van der Merwe and Hui Zhang," LADS: Large-scale automated DDOS detection system," USENIX06, 171-184, 2006.

[36] Y.Chen, K.wang, and W.Ku," Collaborative Detection of DDOS Attacks over Multiple Networks," IEEE Transactions on parallel and distributed systems, 2007.

[37] Neumann Peter G "Inside Risks: Denial of Service Attacks," Communications of ACM, Volume 43, Issue 4, April 2000.

[38] Zong-Lin Li, Guang-Min Hu and Dan Yang, "Global Abnormal correlation analysis for DDoS detection," Key Lab of Broadband Optical Fiber Transmission and Communication network, University of Electronic Science And Technology of China, January 2008.

[39] Microsoft corp.," Microsoft technical support for windows server 2003 R2" http://technet.microsoft.com/en-us/library/cc756026.aspx 9

[40] Intel Inc., "Intel Xeon® Quad core Processor" http://www.intel.com/support/processors/xeon/

[41] Microsoft corp., "How Windows firewall works" http://technet.microsoft.com/en-us/library/cc755604.aspx

[42] Dr. S Kumar, "Can Microsoft's service pack 2 (SP2) security software prevents Smurf attacks?" IEEE computer society, Sep 2006.

[43] Dr. S Kumar," Smurf based denial of service (DDOS) attack amplification in Internet, "IEEE computer society, ICIMP 2007.

[44] Microsoft corp.," Vulnerabilities in Windows TCP/IP Could Allow Remote Code Execution (967723)," Microsoft Security Bulletin MS09-048-Critical, 8th September 2009. http://www.microsoft.com/technet/security/Bulletin/MS09-048.mspx

[45] W.Eddy," TCP SYN Flooding Attacks and Common Mitigations," RFC 4987, The IETF trust, August 2007. http://tools.ietf.org/html/rfc4987

[46] Ramana Rao Kompella, Sumeet Singh, and G. Varghese, "On Scalable Attack Detection in the Network," University of California, San Diego, IMC'04, October 25–27, 2004, Taormina, Sicily, Italy.

[47] S. Shin, K. Kim, and J. Jang, "D-SAT: Detecting SYN flooding Attack by Two-stage statistical approach," Applications and the Internet, Proceedings, the 2005 Symposium on 31 Jan. - 4 Feb. 2005, Page(s):430 – 436.

[48] B. Lim, and Md. S. Uddin, "Statistical-based SYN-flooding Detection Using Programmable Network Processor," Information Technology and Applications, 2005. ICITA 2005, Third International Conference on Volume 2, 4-7 July 2005, Page(s): 465 – 470, vol.2.

[49] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN Flooding Attacks," INFOCOM 2002, Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings. IEEE Volume 3, 23-27 June 2002, Page(s):1530 – 1539.

[50] Y. Ohsita, S. Ata, and M. Murata, "Detecting Distributed Denial-of-Service Attacks by analyzing TCP SYN packets statistically," Global Telecommunications Conference, 2004, GLOBECOM '04, IEEE Volume 4, 29 Nov. - 3 Dec. 2004, Page(s): 2043 – 2049, Vol.4.

[51] D. M. Divakaran, H. A. Murthy, and T. A. Gonsalves, "Detection of SYN Flooding Attacks Using Linear Prediction Analysis," Networks, 2006, ICON '06, 14th IEEE International Conference on Volume 1, Sept. 2006, Page(s): 1 – 6.

[52] V. A. Siris, F. Papagalou, "Application of Anomaly Detection Algorithms for Detecting SYN Flooding Attacks," Global Telecommunications Conference, 2004, GLOBECOM '04, IEEE Volume 4, 29 Nov. - 3 Dec. 2004, Page(s): 2050 – 2054, Vol.4.

[53] R. A. Shaikh, A. A. Iqbal, and K. Samad, "Review Over Anomaly Detection Algorithms For Detecting SYN Flooding Attacks," Engineering Sciences and Technology, 2005, SCONEST 2005, Student Conference on 27-27 Aug. 2005, Page(s): 1 – 5.

[54] B. Xiao, W. Chen, Y. He, and E. H. M. Sha, "An Active Detecting Method against SYN Flooding Attack," Parallel and Distributed Systems, 2005, Proceedings. 11th International Conference on Volume 1, 20-22 July 2005, Page(s): 709 – 715, Vol. 1.

[55] J. Haggerty, T. Berry, Q. Shi and M. Merabti, "DiDDeM: A System for Early Detection of TCP SYN Flood Attacks," Global Telecommunications Conference, 2004, GLOBECOM '04, IEEE Volume 4, 29 Nov. - 3 Dec. 2004 Page(s): 2037 – 2042, Vol.4.

[56] Seung-won Shin, Ki-young Kim, and Jong-soo Jang, “Analysis of SYN Traffic: An Empirical Study,” Technical Document in ETRI, 2004.

[57] H. Wang, D. Zhang, and K. G. Shin, “SYN-dog: Sniffing SYN Flooding Sources,” Distributed Computing Systems, 2002, Proceedings. 22nd International Conference on 2-5 July 2002, Page(s): 421 – 428.

[58] Minho Sung, Jun Xu, ”IP Trace back - Based Intelligent Packet Filtering: A Novel Technique for Defending against Internet DDoS Attacks,” Proceedings of the 10th IEEE International Conference on Network Protocols, p. 302-311, November 12-15, 2002.

[59] Christoph L. Schuba , Ivan V. Krsul , Markus G. Kuhn , Eugene H. spafford , Aurobindo Sundaram , and Diego Zamboni, “Analysis of a Denial of Service Attack on TCP,” Proceedings of the 1997 IEEE Symposium on Security and Privacy, p.208, May 04-07, 1997.

[60] W. Chen, and D. Yeung, “Defending Against TCP SYN Flooding Attacks under Different Types of IP Spoofing,” Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, 2006, ICN/ICONS/MCL 2006, International Conference on 23-29 April 2006, Page(s): 38 – 38.

[61] U. K.Tupakula, V. Varadharajan, and A. K. Gajam, “Counteracting TCP SYN DDoS Attacks using Automated Model,” Global Telecommunications Conference, 2004, GLOBECOM '04, IEEE Volume 4, 29 Nov. - 3 Dec. 2004, Page(s): 2240 – 2244, Vol.4.

[62] Y. W. Chen, “Study on the Prevention of SYN Flooding by Using Traffic Policing,” Network Operations and Management Symposium, 2000, NOMS 2000, IEEE/IFIP 10-14 April 2000, Page(s): 593 – 604.

[63] L. Ricciulli, P. Lincoln and P. Kakkar, "TCP SYN Flooding Defense," in Proceedings of Communication Networks and Distributed Systems Modeling and Simulation (CNDS '99), 1999.

[64] B. AI-Dwmiri, and G. Manimaran “Intentional Dropping: A Novel Scheme for SYN Flooding Mitigation,” INFOCOM 2006, 25th IEEE International Conference on Computer Communications, Proceedings April 2006, Pages: 1 – 5.

[65] Q. Xiaofeng, H. Jihong, and C. Ming “A Mechanism to Defend SYN Flooding Attack based on Network Measurement System,” Information Technology: Research and Education, ITRE 2004, 2nd International Conference on 28 June-1 July 2004, Page(s): 208 – 212.

[66] Y. Ohsita, S. Ata, and M. Murata, "Deployable Overlay Network for Defense against Distributed SYN Flood Attacks," Computer Communications and Networks, ICCCN 2005, Proceedings. 14th International Conference on 17-19 Oct. 2005, Page(s): 407 – 412.

[67] S. Cai, Y. Liu, and W. Gong," Client-Controlled Slow TCP and Denial of Service," 43rd IEEE Conference on Decision and Control, December 14-17, 2004, Atlantis, Paradise Island, Bahamas.

[68] Frank Kargl, Joern Maier and Michael Weber," Protecting Web Servers from Distributed Denial of Service Attacks," May 2001, ACM.

[69] Haidar Safa, Mohamad Chouman, Hassan Artail and Marcel Karam," A collaborative defense mechanism against SYN flooding attacks in IP networks," December 2007, Journal of Network and Computer Applications 31(2008), pages: 509 - 534.

[70] Yogesh Prem Swami, and Hannes Tschofenig," Protecting mobile devices from TCP flooding attacks," Proceedings of first ACM/IEEE international workshop on Mobility in the evolving Internet architecture, December 01-01, 2006, San Francisco, California.

[71] Microsoft Corporation, "Microsoft Windows Server 2003 TCP/IP Implementation Details, Published: June 2003, Updated: March 2006.

[72] D. J. Berbnstein," SYN Cookies". http://cr.yp.to/syncookies.html

[73] Andre Zuquete," Improving the functionality of SYN cookies," 6th IFIP Communications and Multimedia Security Conference, September 2002.

[74] Lemon Jonathan," Resisting SYN flood attacks with SYN cache," February 2002, Proceedings of the BSDCon Conference on File and Storage Technologies. http://people.freebsd.org/~jlemon/papers/syncache.pdf

[75] Alan Shieh, Andrew C.Myers, and Emin G. Sirer," A Stateless Approach to Connection-Oriented Protocols," September 2008, ACM Transactions on Computer Systems, Vol. 26, No. 3, Article 8.

[76] "Attack Detection and Defense Mechanisms," Concepts and examples ScreenOS reference Guide, Volume 4, Juniper Networks, Release 6.1.0, Rev.02.

[77] Raja Sekhar Gade, Hari Krishna Vellalacheruvu and Dr.Sanjeev Kumar, "Performance of Windows XP, Windows Vista and Apple's Leopard under a DDoS attack," International Conference of Digital Society, 2010.

[78] Hari Krishna Vellalacheruvu, Raja Sekhar Gade, Sirisha Surisetty and Dr. Sanjeev Kumar, "Evaluation of TCP – SYN Cache Protection Security Attacks," Poster Presentation, HESTECH science symposium, Sep. 2009.

[79] Raja Sekhar Gade, Hari Krishna Vellalacheruvu, Sirisha Surisetty and Dr. Sanjeev Kumar, "Impact of Land Attack compared for Windows XP, Vista, and Apple Leopard," Poster Presentation, HESTECH science symposium poster presentation, Sep. 2009.

[80] Sirisha Surisetty, Hari Krishna Vellalacheruvu, Raja Sekhar Gade and Dr. Sanjeev Kumar, "Is McAFee Firewall really protects your System," Poster Presentation, HESTECH science symposium poster presentation, Sep. 2009.

APPENDIX A

APPENDIX A

JUNIPER ROUTER CONFIGURATION

```
version 9.2R1.10;
system {
    host-name Gigrouter;
    root-authentication {
        encrypted-password "$1$O/BT2JRu$hj6iZTmhlFvGYCL9ulrUv0";
    }
    login {
        user hari {
            uid 2001;
            class super-user;
            authentication {
                encrypted-password "$1$h2rWZS/j$.BslcKvZknTSdl8iJg2SU.";
            }
        }
    }
    services {
        ssh;
        telnet;
        xnm-clear-text;
        web-management {
            http {
                interface [ ge-0/0/1.0 ge-0/0/2.0 ge-0/0/3.0 ];
            }
        }
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any any;
            authorization info;
        }
        file interactive-commands {
```

```
            interactive-commands any;
        }
    license {
        autoupdate {
            url https://ae1.juniper.net/junos/key_retrieval;
        }
    }
}
interfaces {
    ge-0/0/0 {
        description network192.168.0.0;
        mtu 9192;
        gratuitous-arp-reply;
        no-gratuitous-arp-request;
        gigether-options {
            auto-negotiation;
            ethernet-switch-profile {
                mac-learn-enable;
            }
        }
        unit 0 {
            proxy-arp;
            family inet {
                accounting {
                    source-class-usage {
                        input;
                        output;
                    }
                    destination-class-usage;
                }
                address 40.0.0.1/8;
            }
        }
    }
    ge-0/0/1 {
        description network172.16.0.0;
        mtu 9192;
        gratuitous-arp-reply;
        no-gratuitous-arp-request;
        gigether-options {
            auto-negotiation;
            ethernet-switch-profile {
                mac-learn-enable;
            }
        }
```

```
    unit 0 {
        family inet {
            accounting {
                source-class-usage {
                    input;
                    output;
                }
                destination-class-usage;
            }
        }
    }
}
ge-0/0/2 {
    description network30.0.0.0;
    mtu 9192;
    gratuitous-arp-reply;
    no-gratuitous-arp-request;
    gigether-options {
        auto-negotiation;
        ethernet-switch-profile {
            mac-learn-enable;
        }
    }
    unit 0 {
        family inet {
            accounting {
                source-class-usage {
                    input;
                    output;
                }
                destination-class-usage;
            }
            address 30.0.0.1/8;
        }
    }
}
ge-0/0/3 {
    description server;
    mtu 9192;
    gratuitous-arp-reply;
    no-gratuitous-arp-request;
    gigether-options {
        auto-negotiation;
        ethernet-switch-profile {
            mac-learn-enable;
        }
```

```
        }
        unit 0 {
            family inet {
                accounting {
                    source-class-usage {
                        input;
                        output;
                    }
                    destination-class-usage;
                }
                address 20.0.0.1/8;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 127.0.0.1/32;
            }
        }
    }
}
routing-options {
    static {
        route 20.0.0.0/8 next-hop 20.0.0.1;
        route 40.0.0.0/8 next-hop 40.0.0.1;
        route 0.0.0.0/0 next-hop 40.0.0.2;
    }
}
policy-options {
    prefix-list nrl {
        20.0.0.0/8;
        30.0.0.0/8;
        40.0.0.0/8;
        50.0.0.0/8;
        172.16.0.0/16;
        192.168.0.0/16;
    }
}
security {
    screen {
        ids-option untrust-screen {
            alarm-without-drop;
            tcp {
                syn-flood;
            }
```

```
    }
}
zones {
    security-zone trust {
        address-book {
            address nrl 192.168.0.0/16;
            address nrl1 172.16.0.0/16;
            address nrl2 30.0.0.0/8;
            address nrl3 20.0.0.0/8;
            address nrl4 40.0.0.0/8;
            address nrl7 50.0.0.0/8;
            address-set nrl444 {
                address nrl1;
                address nrl;
                address nrl2;
                address nrl3;
                address nrl4;
                address nrl7;
            }
        }
        host-inbound-traffic {
            system-services {
                all;
            }
            protocols {
                all;
            }
        }
        interfaces {
            ge-0/0/1.0;
            ge-0/0/2.0 {
                host-inbound-traffic {
                    system-services {
                        http;
                        https;
                        ssh;
                        telnet;
                        dhcp;
                    }
                }
            }
            ge-0/0/3.0 {
                host-inbound-traffic {
                    system-services {
                        http;
                        https;
```

```
                ssh;
                telnet;
                dhcp;
            }
        }
    }
  }
  security-zone untrust {
    screen untrust-screen;
    interfaces {
      ge-0/0/0.0;
    }
  }
}
policies {
  from-zone trust to-zone trust {
    policy default-permit {
      match {
        source-address any;
        destination-address any;
        application any;
      }
      then {
        permit;
      }
    }
  }
  from-zone trust to-zone untrust {
    policy default-permit {
      match {
        source-address any;
        destination-address any;
        application any;
      }
      then {
        permit;
      }
    }
  }
  from-zone untrust to-zone trust {
    policy default-deny {
      match {
        source-address any;
        destination-address any;
        application any;
```

```
            }
            then {
                permit;
            }
        }
    }
    default-policy {
        permit-all;
    }
  }
  flow {
    syn-flood-protection-mode syn-cookie;
    tcp-session {
        no-syn-check;
        no-sequence-check;
    }
  }
}
applications {
  application ping protocol icmp;
}
```

APPENDIX B

APPENDIX B

WINDOWS SERVER 2003 TCP/IP MOST EFFECTIVE REGISTRY PARAMETERS

1. MaxFreeTcbs
   Key: Tcpip\Parameters
   Value Type: REG_DWORD number
   Valid Range: 0–0xFFFFFFFF
   Default: The following default values are used (note that small is defined as a computer with less than 19 MB of RAM, medium is 19–63 MB of RAM, and large is 64 MB or more of RAM. Although this code still exists, nearly all computers are large now).
   For Windows Server 2003:
      Small system - 500
      Medium system - 1000
      Large system - 2000
   For Windows XP:
      Small system - 250
      Medium system - 500
      Large system - 1000
   Description: This parameter controls the number of cached (pre-allocated) Transport Control Blocks (TCBs) that are available. A TCB is a data structure that is maintained for each TCP connection.

2. MaxHashTableSize
   Key: Tcpip\Parameters
   Value Type: REG_DWORD number (must be a power of 2)
   Valid Range: 0x40–0x10000 (64-65536 decimal)
   Default: 512
   Description: This value should be set to a power of 2 (for example, 512, 1024, 2048, and so on.) If this value is not a power of 2, the system configures the hash table to the next power of 2 value (for example, a setting of 513 is rounded up to 1024.) This value controls how fast the system can find a TCB and should be increased if MaxFreeTcbs is increased from the default.

3. NumTcbTablePartitions
   Key: Tcpip\Parameters
   Value Type: REG_DWORD—number of TCB table partitions

Valid Range: 1–0xFFFF
Default: 4
Description: This parameter controls the number of TCB table partitions. The TCB table can be portioned to improve scalability on multi-processor systems by reducing contention on the TCB table. This value should not be modified without a careful performance study. A suggested maximum value is (number of CPUs)

4. SynAttackProtect
   Key: Tcpip\Parameters
   Value Type: REG_DWORD
   Valid Range: 0, 1
   0 (no SYN attack protection)
   1 (reduced retransmission retries and delayed RCE [route cache entry] creation if the TcpMaxHalfOpen and TcpMaxHalfOpenRetried settings are satisfied and a delayed indication to Winsock is made.)
   Default: 1 - enabled for Windows Server 2003 Service Pack 1, 0 -disabled for Windows Server 2003 with no service packs installed
   Recommendation: 1
   Description: SYN attack protection involves reducing the amount of retransmissions for the SYN-ACKS, which will reduce the time for which resources have to remain allocated. The allocation of route cache entry resources is delayed until a connection is made and the connection indication to AFD is delayed until the three-way handshake is completed. Note that the actions taken by the protection mechanism only occur if TcpMaxHalfOpen and TcpMaxHalfOpenRetried settings are exceeded.

5. TcpInitialRTT
   Key: Tcpip\Parameters\Interfaces\interfaceGUID
   Value Type: REG_DWORD—number
   Valid Range: 0–0xFFFF
   Default: 3
   Description: This parameter controls the initial time-out in seconds used for a TCP connection request and initial data retransmission on a per-interface basis. Use caution when tuning with this parameter because exponential backoff is used. Setting this value to larger than 3 results in much longer time-outs to nonexistent addresses.

6. TcpNumConnections
   Key: Tcpip\Parameters
   Value Type: REG_DWORD—number
   Valid Range: 0–0xFFFFFE
   Default: 0xFFFFFE
   Description: This parameter limits the maximum number of connections that TCP can have open simultaneously.

7. TcpMaxConnectResponseRetransmissions
   Key: Tcpip\Parameters
   Value Type: REG_DWORD—number
   Valid Range: 0–255
   Default: 2
   Description: This parameter controls the number of times that a SYN-ACK is retransmitted in response to a connection request if the SYN is not acknowledged. If this value is greater than or equal to 2, the stack employs SYN attack protection internally. If this value is less than 2, the stack does not read the registry values at all for SYN attack protection. See also SynAttackProtect, TCPMaxPortsExhausted, TCPMaxHalfOpen, and TCPMaxHalfOpenRetried.

8. TcpMaxSendFree
   Key: Tcpip\Parameters
   Value Type: REG_DWORD—number
   Valid Range: 0–0xFFFF
   Default: 5000
   Description: This parameter controls the size limit of the TCP header table. On machines with large amounts of RAM increasing this setting can improve responsiveness during a SYN attack.

9. TcpAckFrequency
   Key: Tcpip\Parameters\Interfaces\interfaceGUID
   Value Type: REG_DWORD—number
   Valid Range: 0–255
   Default: 2
   Description: Specifies the number of ACKs that will be outstanding before the delayed ACK timer is ignored. Microsoft does not recommend changing this value from the default without careful study of the environment.

10. TcpDelAckTicks
    Key: Tcpip\Parameters\Interfaces\interfaceGUID
    Value Type: REG_DWORD—number
    Valid Range: 2–6
    Default: 2
    Description: Specifies the number of 100-millisecond intervals to use for the delayed-ACK timer on a per-interface basis. By default, the delayed-ACK timer is 200 milliseconds. If you set this value to 0 or 1, the delayed-ACK time is 200 milliseconds. Microsoft does not recommend changing this value from the default without careful study of the environment.

BIOGRAPHICAL SKETCH

Hari Krishna Vellalacheruvu was born in Kommineni Vari Palem, Prakasam, Andhra Pradesh, India on July 20, 1984. The youngest son of Anjamma Vellalacheruvu and Koteswara Rao Vellalacheruvu pursued his Bachelor of Technology degree in 2006 at V. R. Siddhartha Engineering College, Vijayawada, India. He enrolled in the Master of Science in Electrical Engineering program with an emphasis in the area of computer network security at San Jose State University (SJSU) and later transferred to University of Texas Pan American (UTPA).

Hari worked as a research assistant with Dr. Sanjeev Kumar and Dr. Harlow at Network Research Lab (NRL), UTPA where he contributed in ongoing network security research projects. He also served as a teaching assistant in the departments of Computer Engineering, Computer Science and Electrical Engineering.

Present Address: 1809 W Schunior St, Apt 610

Edinburg, TX – 78541

Email : vellala2000@yahoo.co.in