Denial of Service attack in IPv6 networks and counter measurements

Mahmood Nazari; Lokesh Galla

Computer Network Engineering, 15 credits

Halmstad 2016-01-17
Abstract

This thesis describes and expresses the different IPv6 based cyber-attacks which could result in the Denial of Service (DoS) on the IPv6 network. IPv6 is the next generation internet protocol and the demand of its benefits is implacable. Therefore, we tried to answer to the following questions:

How effective is DoS in IPv6 networks?
How to protect IPv6 networks from DoS?

The effect of implementing new changes in network often leads to new challenges and security breaches that make IP based networks challenging to monitor and defense. Better understanding of types of network and traffic requirements will assist in building robust network.

The project done for this thesis is based on investigating the strength of some possible methods of launching the DoS on future solely IPv6 networks with open source tools. Moreover, it is based to signify how differently some network devices respond to this type of attack either locally or remotely in respect of the CPU utilization and the bandwidth usage. Packet analyzer is used to capture and analyze these attacks. The DoS attacks in this project include the protocols IPv6, ICMPv6 and TCP with two different category methods and variety of different IPv6 extension headers and packet formats.

While the extension header mechanism is an important part of the IPv6 architecture, by taking advantage of it including the hop-by-hop option, router alert, AH and specially packets with big size extension headers which cause slow processing, we achieved the goal to strangle the nodes (specially routers) with the high CPU utilization in the project test environment. We also tested the efficacy of IPv6 fragmentation with 36 different options and their results are represented on 3 different areas of the network.

This thesis has concentrated on different kinds of attacks that have low impact on the local area devices such as the default gateway router but very high impact on the targets’ devices (remotely) with different autonomous system number that an attacker would not have any administrative control on. Some of the attacks expressed in this thesis need future work and analysis since they are out of scope for this work and therefore we just illustrate the result of these tests.

The results of this thesis can be used for judgmental evaluation of the IPv6 security against DoS attack.
Contents

1 Introduction ............................................................................................................................................. 8
  1.1 Motivation and Objectives .................................................................................................................. 8
  1.2 Project goal ......................................................................................................................................... 9
  1.3 Method .............................................................................................................................................. 9
2 Background ............................................................................................................................................ 11
3 Related work and Literature Review ..................................................................................................... 13
  3.1 Related work ...................................................................................................................................... 13
    3.1.1 IPv6 Security Analysis ............................................................................................................... 13
    3.1.2 Handling of Overlapping IPv6 Fragments .................................................................................. 14
  3.2 IPv6 Features .................................................................................................................................... 14
    3.2.1 IPv6 packet Fragmentation: ........................................................................................................ 14
    3.2.2 Extension Header Option .......................................................................................................... 14
  3.3 Router Input Queue for IPv6 Packets .................................................................................................. 16
  3.4 IPV6 Packets with Special Need for Processing .................................................................................. 17
  3.5 DoS Attacks ...................................................................................................................................... 18
4 Methodology and Experiments .............................................................................................................. 20
  4.1 Tools .................................................................................................................................................. 20
  4.2 Network Topology .............................................................................................................................. 21
    Scenario 1: ......................................................................................................................................... 22
    Scenario 2: ......................................................................................................................................... 23
  4.3 Methods .......................................................................................................................................... 23
    Method 1: DoS with IPv6 Extension Header Mechanism .......................................................................... 24
    Method 2: DoS with IPv6 Fragmentation Mechanism .......................................................................... 26
  4.4 Experiments ..................................................................................................................................... 48
    DoS with IPv6 Extension Header Mechanism .......................................................................................... 48
    DoS with IPv6 Fragmentation Mechanism .............................................................................................. 49
5 Results and Analysis ............................................................................................................................ 51
  5.1 Denial with IPv6 Extension Header Mechanism .................................................................................. 51
    5.1.1 Denial6 attack on the remote area target (PC3): ........................................................................... 51
    Analysis: ............................................................................................................................................ 62
    5.1.2 “Denial6” attack on a Router as a firewall. ................................................................................. 66
    Analysis: ............................................................................................................................................ 67
  5.2 DoS with IPv6 “fragmentation mechanism” ....................................................................................... 67
    5.2.1 Fragmentation attack on Local area: ......................................................................................... 68
    Analysis: ............................................................................................................................................ 69
    5.2.2 Fragment on a router .................................................................................................................. 70
    Analysis: ............................................................................................................................................ 73
    5.2.3 Fragment on a Router as a Firewall ........................................................................................... 75
    Analysis: ............................................................................................................................................ 77
6 Conclusions .......................................................................................................................................... 79
References ................................................................................................................................................ 83
Appendix 1 .......................................................................................................................... 86

Appendix 2 .......................................................................................................................... 88

ISP1 Router Configuration:................................................................................................. 88
ISP2 Router Configuration:................................................................................................. 89
Chapter 1

1 Introduction

IPv6 is next generation internet protocol. It is developed by the Internet Engineering Task Force (IETF) to provide better performance and also to provide new services in comparison with IPv4. It is intended to serve for insufficient IP address space for the present Internet growth. IPv6 uses a 128 bit address, permitting $3.4 \times 10^{38}$ IP addresses compared to IPv4 which uses 32 bit address scheme which is providing nearly $4.3 \times 10^9$ addresses.

IPv6 contributes a lot more possibilities and improvements compared to IP based networks. It improves considerable Quality of Services (QoS), end to end communications, routing speed and simplicity in routing tables. Besides these improvements, network security is a central issue that remains very significant due to reason of the security threats. Dismissing of the Network Address Translation (NAT) has forced IPv6 nodes with complete transparency.

1.1 Motivation and Objectives

Even though IPv6 simplifies and improves security compared to IPv4, it arises several significant security challenges. Some of these specific changes could cause potential security issues in the IPv6 networks. For example, routing headers, site-scope multicast addresses, Stateless Address Auto Configuration (SLAAC), Neighbor Discovery (ND), Duplicate Address Detection (DAD), header extension, ICMPv6 multicast, bogus error packets in ICMPv6 error Messages, dynamic DNS, excessive hop-by-hop options, packet fragmentations, bogus router advertisements. Moreover, the mechanism of the extension header itself which is added into the IPv6 protocol could be the main root for DoS attacks.

IPsec is embedded in IPv6 while it is not enabled by default. Deployment and management of the IPsec in networks is difficult. Therefore this feature is used less frequently by the administrators of the networks. Additional IPv6’s features have their own security implications that are not yet fully understood.

In addition, some of the IPv4 network security issues such as application-layer attacks, rouge devices, and Packet flooding could also affect IPv6 networks. Respectively, IPv6 networks have to face new, unanticipated security issues as hacking community actively targeting IPv6 networks [1]. As the internet and the number of its devices are developing excessively and the messages being transferred over it become more critical for the companies, the need of providing the users as well as the companies for operative online services become compulsory [2]. The dramatically growth of the DoS attacks and their consequences picked our interest in this topic. This issue which considers less sophisticated attack and easy to launch is intensively present in the IT world. There also has not been an eventual solution for
Chapter 1. Introduction

9

this issue. The significance of the DoS effects on business and economy was best motivation for us to start this project.

Hypothesis:
New mechanisms of IPv6 protocol include IPv6 extension header and packet fragmentations are different from IPv4 Protocol. Are these mechanisms providing good opportunities for an attacker to run DoS on the target devices?

Packets with some of the IPv6 extension header options do not forward at the interrupt levels in the hardware of an intermediate device thus they need a software interaction for their forwarding process. On the other hand, fragmentation and defragmentation can only be handled at source and destination nodes therefore supposedly the fragment packets should not affect intermediate devices but only the target nodes.

This thesis uses the IPv6 fragment and IPv6 extension header mechanisms separately and combined as methods to launch DoS attack on the intermediate nodes as well as target nodes. We assume by penetrate testing these two mechanisms we can find a way to launch DoS attack on the devices in the network. Furthermore in the case of success, we assume that enabling an IPv6 access-list on a router against the abrupt traffic will successfully stop this kind of attack to begin our counter measurement with and further our investigation.

1.2 Project goal
Goal of the Project is to investigate the effectiveness of DoS by taking advantage from fragmentation and extension header mechanisms as methods for DoS attacks on a solely IPv6 networks. Moreover, to demonstrate this type of attacks that can be launched on the network without requiring high competence of an attacker. We also tried to concentrate on different types of DoS attacks that have low impact on the local area devices such as the default gateway router of a local network but very high impact on the targets’ devices (remotely). The idea is to exploit the hop-by-hop extension header mostly with the router alert and especially any packet with more than two extension headers to achieve the goal of strangling the target with the CPU utilization. Further with this strategy, reassembling of the fragment packet can be also very CPU intensive. This thesis is done to investigate the following questions:

How effective is DoS in IPv6 networks?
How to protect IPv6 networks from DoS?

1.3 Method
The project done for this thesis was based on investigating the strength of some possible methods of launching the DoS on future solely IPv6 networks with open source tools. This thesis concentrates on DoS related attacks that have low impact on the local area devices such as the default gateway edge router but very high impact on the targets. The main objective of this research is to exploit the extension header and fragmentation mechanism in IPv6 protocol against DoS attack. In this quantitative research, individual test cases were run 10 times and samples were
logged for two different methods. Method number 1 has 7 different test cases which evaluate the IPv6 extension header and method number 2 evaluates the fragmentation mechanism for this protocol with 36 different test cases. These tests are explained in detail later in this paper.
Chapter 2

2 Background

Security and privacy for individuals are in the state of the destruction. DoS could be also a method to compromise the security of online services. The following are some of the recent major DoS attacks in the internet.

In the year 2011 security experts requested a patch against DoS attacks for Microsoft and Juniper devices [3].

For the first time in February 2012, Distributed Denial of Service (DDoS) attacks were launched with IPv6 in the Internet [4]. According to this article Arbor networks traffic floods or DDoS attacks are more than 50% chances in the present internet. In addition, the Arbor networks security reports had predicted that IPv6 DoS attacks will gradually increase over time as IPv6 widely is deployed in the internet.

There are two problems which make IPv6 particularly vulnerable. Firstly, the immature network infrastructure and secondly unorganized or misconfigured gateways that link to IPv4 and IPv6 networks [5]. The survey reported that varying costs of dealing with DDoS attacks are about $1,300 to $ 1.5 million depending on business types. “The attack can cause the organizations unable to provide online services to their customers or other companies to sell their products. DDoS attack generates the one-to-many dimensions attribute to the DoS attacks causing its impact much more severe. A DDoS victim can suffer from the impact that varies in total system failure, file corruption or partial loss of services”. [5]

![Top Network Attacks](image)

Figure 1: McAfee Labs Threat report and Statistics [7]

Figure 2 is published by McAfee lab on February 2015 showing that DoS is the second security threat (22%) in the internet.
Nowadays, deploying IPv6 in the networks to replace IPv4 is inescapable. The reason is mostly because of the larger portion (128 bits) of IP addresses in IPv6. Moreover, IPv4 ARP attacks are replaced with IPv6 ND attacks. Spoofed Router Advertisements (RA); DAD; flooding; fragmentation; and Smurf attacks are vulnerable in IPv6 network.

The non-existence of NAT functionality gives capability to the attackers to launch their attack directly towards the target devices. In other words, the IPv6 nodes cannot be hidden behind the NAT.

In figure 2 which is based on The CERT Coordination Centre (CERT/CC, was originally founded by DARPA as the Computer Emergency Response Team), the result of their research about the security treats is shown. Based on this research, Intruder knowledge about playing sophisticated attacks decreased rapidly due to the help of free online available tools and open sources programs and individuals.

All the above reasons did motivate us to investigate deeper about DoS attacks in IPv6 networks infrastructure and their comparison.
Chapter 3

3 Related work and Literature Review

Beginning of the Internet, IPv4 protocol was introduced and implemented for IP based networks. It was used mostly for research and development purposes and security was not a big concern at that part of the time. Therefore IPv4 protocol has minimal security options compared to newer version, IPv6. Later when security issues were presented security became the main focal point for IP based networks. Since IPv4 protocol has its limitations in security, upper layer security protocols were introduced. For example, digital signatures, encryption methods, entity authentication, access control, IPsec, Secure Socket Layer (SSL), HTTPS, and so on…

Even with upper layer security architectures, the lower layers are still exposed and unencrypted on the public networks. An intruder or attacker utilizes this opportunity to gather information about IPv4 based systems and their communications. This bug leads the network with IPv4 protocol based to DoS attacks, spoofed attacks, and network interception. On the other hand, even with higher security concerns in the design of the IPv6 protocol, this protocol is still vulnerable for these kinds of attacks.

3.1 Related work

3.1.1 IPv6 Security Analysis

“IPv6 Security Analysis” is a technical report done by Jose Gonzalo Bejar in the year 2014 at Syracuse University. This thesis reports IPv6 security penetrated by using tools such as Kali and THC. It also focuses on exploring Man-In-The-Middle (MITM), DoS and reconnaissance attacks in solely IPv6 based networks.

Scanners are the first tools used for reconnaissance attack to explore the network and open ports in the network. The large size of IPv6 addresses scanning is very challenging by using traditional scanning methods therefore in their project instead they crafted multicast addressing which is more or less detrimental in respect of the time needed.

DoS attacks in this thesis were achieved with 3 different methods all locally with router advertisement messages, invalid gateway IPv6 address and ICMPv6 redirect massages. They tried to show that DoS attacks are still present issue in IPv6 based networks. Operating systems do not protect their routing tables from fake routes thus leads to inject DoS attacks on the hosts.

Limitation:

“Most of the attacks described in this report were only valid locally and their results are limited.” [9]. Their method also to launch DoS was only based on the router advertisement messages, invalid gateway and ICMPv6 redirect massages.
3.1.2 Handling of Overlapping IPv6 Fragments

This thesis is published by IETF and is written by Suresh Krishnan. It represents the fragmentation process overlapping allowance in the IPv6 protocol and its security issues. It states that the algorithm specified in the IPv6 protocol for fragmentation could cause security issues with the firewalls. The current counter measurement of the most firewalls to deny the packets with overlap offset based on the [RFC1858] is criticized. There is also stated that they fail to work properly on IPv6 protocol for TCP packets. The security issue case considered as follows: a TCP packet with SYN=1 and ACK=1 is sent to the target behind the firewall. The firewall assumes this packet as respond to an already requested packet from the target by allowing any further packets with the same fragmentation ID to pass through firewall. Therefore, the attacker can use the same fragmentation ID for the rest of his malicious traffic to the target. The dissolve this issue, the author recommends to silently drop any overlapped IPv6 fragmentation packets with the same fragmentation ID [10].

Problems with IPv6 security features
IPv6 could be considered a solution to tackle the DoS issues on the IPv6 protocol but IPv6 relies on a Public-key-infrastructure (PKI) that has not yet been fully standardized.

Additional work needs to be done to Internet-Key-Exchange (IKE) and in improving protection against DoS and flooding attacks.

3.2 IPv6 Features

Fragmentation and extension headers of IPv6 are well defined in RFC 2460 [11].

3.2.1 IPv6 packet Fragmentation:

IPv6 packet header does not include fragmentation option but instead it is supported in the next header options. In IPv6 network, source nodes perform path Maximum Transmission Unit (MTU) discovery in TCP and UDP traffic transportations. All IPv6 routers or any other intermediate nodes must not attempt to perform fragmentation on IPv6 datagrams. If an IPv6 router gets an IPv6 packet that is bigger than the size of MTU then this router will discard the packet and will generate “packet too big” ICMPv6 message to send to the host. Source node of an IPv6 network is allowed to perform payload fragmentation according to minimum MTU size of route path. IPv6 fragmentation is done by fragmentation control header that acts the same as IPv4 fragmentation control fields.

In IPv6 protocol, packet fragmentation options are added into the next header option after the original packet is fragmented. Next header contains reserved field, fragmentation offset, res, flags M, and Identification.

3.2.2 Extension Header Option

IPv6 extension header carries information with 8-bit next header field in IPv6 header. Optional internet-layers information is encoded in extension headers in between
original IPv6 header at layer 3 and upper-layer header (layer 4, transport layer) refer to ISO model. IPv6 header may contain one or more different values in extension headers chain. 8-bit next header defines the functionality and links all headers of an IPv6 datagram. Few functionalities of next header field are defined below.

- It is embedded in all IPv6 packet headers.
- It contains reference number of the first IPv6 extension header type.
- Next header field of “first packet” contains the information of extension header type of second header. Second header’s, next header field contains the information about third header and so on.
- The last header of an IPv6 datagram’s next header field contains the protocol number of encapsulated information for higher-layer protocol. Figure 3 shows an example of IPv6 datagram extension header.

Figure 3: IPv6 datagram’s extension header process. [12]

Extension headers are instructional part of IPv6 protocol and they define functions and services of an IPv6 datagram as it is shown in the table 1. Note that the first column in this table represents the initials used for the figures in this thesis.

<table>
<thead>
<tr>
<th>Initial</th>
<th>Header Type</th>
<th>Next Header Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6HD</td>
<td>Basic IPv6 Header</td>
<td>-</td>
</tr>
<tr>
<td>Hop-by-Hop</td>
<td>Hop-by-Hop Options</td>
<td>0</td>
</tr>
<tr>
<td>Des Option</td>
<td>Destination Options</td>
<td>60</td>
</tr>
<tr>
<td>Rout_HD</td>
<td>Routing Header</td>
<td>43</td>
</tr>
<tr>
<td>FR_HD</td>
<td>Fragment Header</td>
<td>44</td>
</tr>
<tr>
<td>AH</td>
<td>Authentication Header</td>
<td>51</td>
</tr>
<tr>
<td>ESPH</td>
<td>Encapsulation Security Payload Header</td>
<td>50</td>
</tr>
<tr>
<td>MLD</td>
<td>Mobility Header</td>
<td>135</td>
</tr>
<tr>
<td>-</td>
<td>No Next header</td>
<td>59</td>
</tr>
<tr>
<td>TCP</td>
<td>Upper layer-TCP</td>
<td>6</td>
</tr>
<tr>
<td>-</td>
<td>Upper Layer-UDP</td>
<td>17</td>
</tr>
<tr>
<td>ICMPv6</td>
<td>Upper Layer-ICMPv6</td>
<td>58</td>
</tr>
</tbody>
</table>

Table 1: Next header Code values

*Hop-by-Hop Options:*
Hop-by-hop options defined as “0” zero value of the next header in IPv6 extension header. This option used for Router alerts, Jumbo grams, Multicast Listener Discovery (MLD) and Resource Reservation Protocol (RSVP).

**Fragmentation header options:**
Fragmentation option is defined as value “44” of the next header in IPv6 extension header.

**Destination header options:**
Destination option is defined as value “60” of the next header in IPv6 extension header. It is used for routing.

**Mobility header options:**
This is used for support of IPv6 Mobile Services.

**Authentication header:**
Authentication Header (AH) is defined as value “51” of next header in IPv6 extension header. It is used for IPsec purposes.

**Encapsulation Security Payload header (ESP):**
ESP header is defined as value “51” of the next header in IPv6 extension header. Data followed by encapsulation security header is encrypted.

Figure 4 is an example of an IPv6 datagram followed by two extension headers chaining. First header from the left contains “next header” as zero represents the hop-by-hop options. Second header contains “next header” value as “44” that represents fragmentation and it is followed by the third header which contains value “6”. This value represents TCP as the upper layer protocol.

![IPv6 datagram with two extension headers](image)

**3.3 Router Input Queue for IPv6 Packets**
The forwarding procedure of an input packet in the router interface is to be done at an interrupt level. At this level for this procedure, if there is no match in the cache table the packet is added in the input queue to be processed. If the process timing causes the queue to be overloaded then some packets would be dropped with a predefined algorithm by the router. In this project environment, the FIFO queuing default method of the router was kept and no changes in the queue buffer of the interfaces had been made. Moreover, the interface output queue generally needs shorter time to forward a packet on the physical channel. Therefore, dropping packets in the output interface is less likely [13].
Flushes counter is incremented on the switched traffic processing. It counts upward when the Selective Packet Discard (SPD) which is the policy of the router’s IP processing queue to selectively discard a packet. This policy has the responsibility to drop packets when the queue is between minimum and maximum threshold to ensure that the router would not starve for CPU from receiving the important packets such as routing protocols and keep alive packets.

The Router Processor (RP) divides its processor memory into pools. Each pool contains a number of memory blocks of equal size. These memory blocks are called buffers.

When an interface processor needs to pass packet to the RP it asks for a fix buffer size as followed below from RP with the size bigger or equal of the packet size.

- Small — 104 bytes buffers
- Very Big — 4520 byte buffers
- Middle — 600 byte buffers
- Large — 5024 byte buffers
- Big — 1524 byte buffers
- Huge — 18024 byte buffers

When the interface processor asks for a buffer, if there was a free buffer it would be allocated otherwise message would be created for the buffer algorithm to make new buffer for that pool size. But at this stage, the packets are not dropped. Instead, the buffer algorithm tries to allocate a bigger size buffer to the packets. If no next level buffer could be allocated. Otherwise the packet is dropped.

Buffer failure can be one of the most possible explanations of the packet dropping. If buffer failure happens RP will receive a high level of requests to make more buffers which might take several microseconds. This time consumption might cause more packets to drop and more requests issued towards RP. It also might happen that the buffers created by RP would be used with the same rate as they are created. In this case, RP has to spend more time on buffer creation than packet processing which could result in severe packets dropping.

### 3.4 IPv6 Packets with Special Need for Processing

Some type of packets has to be handled with the software in the router. Such types are as follows [15]:

- Packets with a hop-by-hop option header
- Packets with the same destination IPv6 address as that of routers
- IPv6 uRPF: Software performs this uRPF for all packets.
- Packets with a TTL that is less than or equal to 1
- IPv6 reflexive ACLs: Software handles these reflexive ACLs.
- Packets with a TTL that is less than or equal to 1

IPv6 security devolvement security is the biggest concern. IPsec is embedded in IPv6 protocol [1]. IPv6 enhancements offer different services and easy troubleshooting options; however different options often lead to more security issues in the
networks. Different feature of IPv6 have been expressed at “Chapter 2” in detailed. In comparison with IPv4 networks, some security tools and devices such as firewalls still do not support IPv6. Some others do not support IPv6 properly due to the misconfigurations by the administrators [16]. This leads IPv6 network more likely to be vulnerable for attacks.

Furthermore, according to the IPv4-mapped address with AF_INET 6 sockets, IPv4 addresses using IPv6 Application Programming Interface (API) [17]. API is programmed and designed to support IPv6 and IPv4 (Dual stack) on the same interface. This ambiguity makes network nodes to be exposed for IPv6 based vulnerabilities.

### 3.5 DoS Attacks

One of the best descriptions of a denial of service attack is provided by the US-CERT that defines a DoS as an attack where an “attacker attempts to prevent legitimate users from accessing information or services. By targeting a host and its network connections, an attacker may be able to prevent a targeted host from accessing email, websites, online accounts (banking, etc.), or other services that rely on a affected computer” [8]. Denial-of-service attacks are most common exploits happening on todays’ internet. DoS attacks host two sub types such as Malformed Packet attacks and Packet flood attacks. The most common DoS attacks occur with flooding either bogus or undefined traffic to the targeted host. Targeted host get busy with handling this flooding, so it fails to give services for legitimate users. There exist different DoS type attacks at different layers of Open Systems Interconnection Model (OSI) model. Table 2 represents the possible DoS attacks in Different Layers of OSI.

<table>
<thead>
<tr>
<th>OSI Layer</th>
<th>DoS types</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.</td>
<td>Application</td>
</tr>
<tr>
<td></td>
<td>Email Spams; Web DoS.</td>
</tr>
<tr>
<td>6.</td>
<td>Presentation</td>
</tr>
<tr>
<td></td>
<td>Malformed SSL requests.</td>
</tr>
<tr>
<td>5.</td>
<td>Session</td>
</tr>
<tr>
<td></td>
<td>Telnet DoS.</td>
</tr>
<tr>
<td>4.</td>
<td>Transport</td>
</tr>
<tr>
<td></td>
<td>SYN Floods; Smurf Attacks.</td>
</tr>
<tr>
<td>3.</td>
<td>Network</td>
</tr>
<tr>
<td></td>
<td>ICMP/v6 Flooding; RA; NS, and etc…</td>
</tr>
<tr>
<td>2.</td>
<td>Data-Link</td>
</tr>
<tr>
<td></td>
<td>MAC Flooding</td>
</tr>
<tr>
<td>1.</td>
<td>Physical</td>
</tr>
<tr>
<td></td>
<td>Dummy packet Attack; Packet with more bit errors.</td>
</tr>
</tbody>
</table>

**Malformed Packet Attacks**

Malformed IP packets consist of invalid header values, packet size, offsets, next header options and fragmentation. When a host receives an invalid packet then it has to allocate some resources to handle this packet. If a targeted host receives enough amounts of malformed IP packets then the system could crash or hangs from allocating system resources for the legitimate programs.

Intrusion Detection System (IDS) attack policy defines blocking methods for malformed IP packets. The philosophy behind protection of malformed IP attacks is, disallowing anything that you do not have a known reason to allow [18]. Similar rules are applied for IPv6 based networks by restricting the unknown services for IPv6, such as next header, IPv6 destination options and IPv6 hop-by-hop options.
Packet flood attacks
Packet flooding attacks are most famous DoS attack. Flooding is a technique used in IP based networks, when node sends packets to all outgoing links. It is similar technique as IP broadcasting. In the context of DoS attacks, flooding is used by flooding of unrelated or crafted network traffic in the network to stop network service.
Chapter 4

4 Methodology and Experiments

This section defines tools, network topology and methods which were used to carry out the experiments and analysis.

4.1 Tools

**THC**

The Hacker Choice (THC-IPv6) is an open source toolkit maintained by "Van Hauser". THC allows the penetration test on the IPv6 protocol to challenge the weaknesses of node. This toolkit includes over 50 separate tools that allow performing such a task on IPv6 based protocols and headers. The THC tool is capable of IPv6 node discovery, IPv6 router impersonate, and initiate DoS attacks.

THC is an assembled hacker group from around the world. It is an open source community who develops and expose the security vulnerability of IP based networks. The aim of their project is to expose the security breaches of products. THC is founded in 1995 and it has been published scientific thesis and releases security penetration tools [19-B].

Note: The version used in this project: Last update 2014-12-30, current public version: v2.7

Some of the tools that THC allows [19-A]:

- "parasite6": ICMPv6 neighbor solicitation/advertisement spoofer that can be used to launch Man-In-The-Middle attack.
- "fake_router6": To advertise a node as a highest priority router on the network to redirect the traffic to the defined node
- "redir6": This tool takes advantage of the icmp6 redirect spoofer to launch man-in-the-middle attack.
- "dos-new-ip6": Avoid a new IPv6 node joined to the network to get any IPv6 address.
- "denial6": Seven different methods of denial-of-service tests against a target by taking advantage of the IPv6 extension header mechanism.
- "fragmentation6": Performs fragment packets with 36 different types of options to check the firewall and its implementation. This tool could be used with flood mode to launch a denial-of-service.
- "ndpexhaust26": to flood the target /64 networks with ICMPv6 Too Big error messages to make a bottle neck.

The tools "fragmentation6" and "denial6" were used in this project. They are explained extensively in their related section 2.3 respectively on this thesis.

**Wireshark**
Wireshark is an IP based network protocol analyzer and sniffer. It reads packets from the network by the help of pcap, tcpdump and etc. and details them into easy understandable way. It is an open source network analyzer founded in 1998 [20]. It works in two different modes “Promiscuous” and “Non-Promiscuous”. The difference between them is, in promiscuous mode node’s NIC can sniffs or read from all the traffic packets on the channel while in non-promiscuous mode it only reads the packets belonging to the hosted node.

Wireshark supports rich set of features to represent IP packet information [20]. Following are a few of them.

- Live capture and offline analysis.
- Deep inspection of hundreds of protocols, with more being added all the time.
- Standard three-pane packet browser.

Its default fields include; packet number, time, source address, destination address, name of the protocol, information about the protocol.

- Captured network data can be browsed via a GUI, or via the TTY-mode TShark utility
- Read/write many different capture file formats: tcpdump (libpcap), Pcap NG, Catapult DCT2000, Cisco Secure IDS iplog, Microsoft Network Monitor, Network General Sniffer® (compressed and uncompressed), Sniffer® Pro, and NetXray®, Network Instruments Observer, NetScreen snoop, Novell LANalyzer, RADCOM WAN/LAN Analyzer, Shomiti/Finisar Surveyor, Tektronix K12xx, Visual Networks Visual UpTime, WildPackets EtherPeek/TokenPeek/AiroPeek, and many others.
- Live data can be read from Ethernet, IEEE 802.11, PPP/HDLC, ATM, Bluetooth, USB, Token Ring, Frame Relay, FDDI, and others (depending on your platform)
- Decryption support for many protocols, including IPsec, ISAKMP, Kerberos, SNMPv3, SSL/TLS, WEP, and WPA/WPA2.
- Coloring rules can be applied to the packet list for quick, intuitive analysis.

### 4.2 Network Topology

The project done in this thesis is based on investigating the strength of some possible methods of launching the DoS on future solely IPv6 networks with open source tools. This thesis tries to concentrate on DoS related attacks that have low impact on the local area devices such as the default gateway edge router but very high impact on the targets’ devices (remotely) with different autonomous system number that an attacker would not have any administrative control on.

To satisfy the requirement of the thesis, our experiments contrive in two scenarios as follows. For simplicity only edge routers has been configured and used.
• **Scenario 1**: Basic design between two autonomous networks with routing capabilities but without any security implemented.

• **Scenario 2**: Based on “Scenario 1” but explicit Access-List (ACL) has been implemented on ISP2 to measure counter measurement.

**Scenario 1:**

“Scenario 1” is a basic representation of a network topology used throughout this thesis. It has been designed and configured with basic functions and routing between IPv6 based nodes.

Figure 5 represents this topology and how the nodes were connected to each other.

![Network Topology](image)

**Figure 5: Network Topology**

**Note:** *ISP routers configuration is attached in section Appendix 2.*

**Test Equipment**

- **PC 1**: a DELL pc which ran Windows 7 Enterprise Service pack 1 OS with Intel core (TM)2 Duo CPU 3.00 GHZ 64bit and 4GB RAM with Gigabit network connection Intel 82567LM-3 model Using VMware Player 6.0.2 to run Ubuntu 14.10 i386 Desktop version. The LAN setting for VMware is on auto configure and all IPv4 were disabled.
- **PC 2**: a DELL pc which ran Windows 7 Enterprise Service pack 1 OS with Intel core (TM)2 Duo CPU 3.00 GHZ 64bit and 4GB RAM with Gigabit network connection Intel 82567LM-3 model.
- **PC 3**: HP laptop which ran a Linux 14.04.1 LTS with 1GB RAM and Intel Atom CPU N450 with 100MB RJ45 LAN.
Chapter 4. Methodology and Experiments

- PC 4: a DELL pc which ran Windows 7 Enterprise OS with Intel core 5-2400 @3.10GHz 64bit and 8GB RAM with Gigabit network connection Intel 82579LM model.
- SW1: Cisco Catalyst 3550 switch with Cisco IOS Software, C3550 Software (C3550-IPSERVICESK9-M), Version 12.2(44) SE6, RELEASE SOFTWARE (fc1).
- SW2: Cisco Catalyst 3550 switch with Cisco IOS Software, C3550 Software (C3550-IPSERVICESK9-M), Version 12.2(44) SE6, RELEASE SOFTWARE (fc1).

Note: Windows and Ubuntu OS were fully updated by the time that the attacks were launched. The IPv4 were disabled in all devices to have a pure functional IPv6 in the whole network. The attacks were launched from 15 of December 2014 to 15 of March 2015.

Scenario 2:

Scenario 2 is based on the scenario 1 with an extended IPv6 access-list configured on ISP2. ACL is configured manually to block specific IP (attacker’s PC) to reach the outside network. Figure 6 represents this topology.

4.3 Methods

The main objective of this research is to exploit the extension header and fragmentation mechanism in IPv6 protocol against DoS attack. In this quantitative research, individual test cases were run 10 times and samples were logged for two
different methods. Method number 1 has 7 different test cases which evaluate the IPv6 extension header and method number 2 evaluates the fragmentation mechanism for this protocol with 36 different test cases. For every method and for each test case, routers had been reloaded before head to refresh all the routers statistics. The results shown in the tables are the average of these results. The samples had about 1% variance on the CPU utilizations and about 5% on the bandwidth usages. The test cases were run in different areas of the main network topology as it is shown in figure 5 in section 2.2.

**Method 1: DoS with IPv6 Extension Header Mechanism**

DoS Attack was performed by using the tool “denial6” in THC with 7 different test cases against a remote area. The attack also was run against firewall with the explicit denial of the attacker IPv6 address. In this method the victim node was flooded with a different type of packet by using the extension header mechanism. Hop-by-hop option, router alert, destination options, AH and packets with big size extension headers are some of the structures of this method’s packet types.

In this method, the impact of the tool “denial6” with 7 different options as they are explained below is investigated on the topology nodes.

- Test case 1: Great size hop-by-hop header with router-alert followed by unknown options. Figure 7 shows the format of this packet.

  ![Figure 7: Denial packet format for Test case 1](image1)

  - Test case 2: Great size destination option filled by unknown options and ICMPv6. Figure 8 shows the format of this packet.

  ![Figure 8: Denial packet format for Test case 2](image2)

  - Test case 3: hop-by-hop header with router alert option with 181 times repeated headers. Figure 9 shows the format of this packet.
- Test case 4: hop-by-hop header with router alert option followed by duplicated destination option, 179 times and ICMPv6 protocol. Figure 10 shows the format of this packet.

- Test case 5: IPv6 Packets with AH header and ICMP protocol. Figure 11 shows the format of this packet.

- Test case 6: Just the first fragment packet of an ICMP protocol with a hop-by-hop header and router alert. Figure 12 shows the format of this packet.

- Test case 7: Great size hop-by-hop header filled with unknown options without router alert. Figure 13 shows the format of this packet.
Method 2: DoS with IPv6 Fragmentation Mechanism

DoS Attack was performed by using the tool “fragmetion6” in THC with 36 different test type cases of IPv6 fragmentation mechanism on the local and remote areas. The same tests also were run against the ACL on a router that acts as a firewall. In addition, the speed of the traffic flow with two different speed rates is investigated. These different test cases packet types have been explained as follows:

Please note that “real data” in the following description represents that the data payload was filled with repeated “A” and “fake data” the repeated “Z” as the data payload. The reason is that there are in some of the test cases below packets with duplication format (the same offset, data payload size and fragmentation ID). This feature distinguishes the packets after they passed the network nodes such as firewall from each other at the sniffing point by the packet analyzers.

Fragmentation test case number 1
Attack type number 1: Flooding the target node with 4 fragment packets with different data within the stream as an ICMPv6 ping request original packet. Each sequence of these 4 packets has the same fragmentation ID number. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with fake data and TTL=1
3. Duplicate second fragment packet with real data and TTL=64
4. Third and final fragment packet

Figure 14 shows the packets in this test case in more details.

Fragmentation test case number 2
Attack type number 2: Flooding the target node with 4 fragment packets with different data within the stream as an ICMPv6 ping request original packet. Each sequence of these 4 packets has the same fragmentation ID number. Only the first packet has a real data and the second packet has a TTL of 1. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with real data and TTL=64 as the offset is to 51 with 400 bytes data payload
3. Duplicate second fragment with fake data and TTL=1, offset set to 51 and 400 bytes data payload
4. Third and final fragment packet

Figure 15 shows the packets in this test case in more details.

Figure 15: fragmentation6 test case2

**Fragmentation test case number 3**
Attack type number 3: Flooding the target node with 4 fragment packets with different data within the stream as an ICMPv6 ping request original packet. Each sequence of these 4 packets has the same fragmentation ID number. Only the first packet has a real data. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with real data
3. Duplicate second fragment packet with a fake data
4. Third and final fragment packet

Figure 16 shows the packets in this test case in more details.

Figure 16: fragmentation6 test case3

**Fragmentation test case number 4**
Attack type number 4: Flooding the target node with 4 fragment packets with different data within the stream as the ping request original packet fragmentation.
Denial of Service attack in IPv6 networks and Counter measurements

Each sequence of these 4 packets has the same fragmentation ID number. Only the second packet has a fake data but the third packet has the same offset number (51-100) and the same size as the second packet. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with fake data
3. Duplicate second fragment with real data
4. Third and final fragment packet

Figure 17 shows the packets in this test case in more details.

---

**Fragmentation test case number 5**

Attack type number 5: Flooding the target node with 4 fragment packets with different data within the stream as an ICMPv6 ping request original packet. Each sequence of these 4 packets has the same fragmentation ID number. In this test case, The attacker sent an extra fragment packet after the first 3 fragment packets forces the destination node to investigate it in order to reassemble the original packet since it has the same fragmentation ID as the previous packets. This packet also leaves the M flag set indicating that more packets for this fragmentation ID will continue after. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet
3. Third and final fragment (M flag reset: “0”)
4. Duplicate second fragment packet with fake data

Figure 18 shows the packets in this test case in more details.
Chapter 4. Methodology and Experiments

Fragmentation test case number 6
Attack type number 6: Flooding the target node with 3 fragment packets with different data within the stream as an ICMPv6 ping request original packet. Each sequence of these 3 packets has the same fragmentation ID number. The second packet and the third one have a conflict (Overlap) on their offset. The second packet offset starts from 51 and finishes at 100 (50 octet payload length) but the third packet starts its offset from 85 and finishes at 150.

Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet
3. Third and final fragment packet which overlaps into previous packet

Figure 19 shows the packets in this test case in more details.

Fragmentation test case number 7
Attack type number 7: Flooding the target node with 3 fragment packets with different data within the stream. Each sequence of these 3 packets has the same fragmentation ID number. The second packet does have both real data and fake data (the fake data part is the 120 bytes which overlaps with the last packet). The second packet and the third one have a conflict (Overlap) on their offset. The Second packet...
offset starts from 51 and finishes at 100 (50 octet payload length) but the third packet starts its offset from 85 and finishes at 150.

Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet
3. Third and final fragment packet which overlaps with the second one

Figure 20 shows the packets in this test case in more details.

**Fragmentation test case number 8**

Attack type number 8: Flooding the target node with 3 fragment packets with different data within the stream. Each sequence of these 3 packets has the same fragmentation ID number. The second fragment packet has a real data and is transmitted last. The second packet and the third fragment packet have a conflict (Overlap) on their offset. The third fragment packet’s offset starts at 101 but the second packet’s offset starts from 51 and finishes at 121. The third packet has a mix of real and fake data for the part that the offset has a conflict with the second packet as the data payload.

Each time these packets were sent with the following order: (Third fragment packet was sent before the second fragment packet):

1. First fragment packet
2. Third and final fragment packet - but not the last
3. Second fragment packet which overlaps into the third (Previous packet)

Figure 21 shows the packets in this test case in more details.
Chapter 4. Methodology and Experiments

Figure 21: fragmentation6 test case8

Fragmentation test case number 9

Attack type number 9: Flooding the target node with copies of one fragment packet with ping request data (ICMPv6, type 128) as a complete packet. This fragment packet has a valid data with M flag reset to 0. The packet was sent as follows:

- One fragment packet with ping request

Figure 22 shows the packets in this test case in more details.

Figure 22: fragmentation6 test case9

Fragmentation test case number 10

Attack type number 10: Flooding the target node with copies of one fragment packet with ping request data (ICMPv6, type 128) as a complete packet. This fragment packet has double fragmentation headers. Each of these fragmentation headers has a unique fragmentation ID, offset of “0” and their M flag set to “0”. The packet was sent as follows:

- Double fragmentation header in a complete fragment packet and ping request as the data payload

Figure 23 shows a sample of this packet for this attack initiated from the attacker.
Fragmentation test case number 11
Attack type number 11: Flooding the target node with copies of one fragment packet with ping request data (ICMPv6, type 128) as a complete packet. This fragment packet has 137 fragmentation headers. Each of these fragmentation headers has a unique fragmentation ID, offset of “0” and their M flag set to “0”.

The packet was sent as follows:

- 137 fragmentation headers in a complete fragment packet and ping request as the data payload

Figure 24 shows a sample of this packet for this attack initiated from the attacker.

Fragmentation test case number 12
Attack type number 12: Flooding the target node with copies of one fragment packet with 8 bytes of ping request data (ICMPv6, type 128) as a complete packet. This fragment packet has 175 fragmentation headers. Each of these fragmentation headers has a unique fragmentation ID, offset of “0” and their M flag set to “0”.

The packet was sent as follows:
- 175 fragmentation headers in a complete fragment packet and ping request as the data payload

The format of the packet headers for this type is very similar to the previous type (Attack type number 11) with differences of only one more fragmentation headers. Figure 25 shows a sample of this packet for this attack initiated from the attacker.

Figure 25: fragmentation6 test case 12

**Fragmentation test case number 13**

Attack type number 13: Flooding the target node with 54 fragment packets with total packet size of 65486 bytes for the original packet size for a ping request, ICMPv6 type 128. Each fragment packet has a size of 1232 bytes except for the last one with the size of 198 bytes. The offsets of these packets were set correctly. All of these 54 packets have the same fragmentation ID. The next set of the 54 packets in the flood steam also has the same ID. The packets were sent as explained in orders.

- 54 fragment packets for a ping request

Figure 26 shows a sample of this packet for this attack initiated from the attacker.

Figure 26: fragmentation6 test case 13
Fragmentation test case number 14

Attack type number 14: Flooding the target node with 54 fragment packets with total packet size of 65495 bytes for the original packet size ping request, ICMPv6 type 128. Each fragment packet has a size of 1232 bytes except the last one with the size of 207 bytes. The offsets of these packets were set correctly. All of these 54 packets have the same fragmentation ID. The next set of the 54 packets in the flood steam also has the same ID.

The packets were sent as explained in orders.

- 54 fragment packets for a ping request

Figure 27 shows a sample of this packet for this attack initiated from the attacker.

![Fragmentation test case 14](image1)

Fragmentation test case number 15

Attack type number 15: Flooding the target node with 54 fragment packets with total packet size of 65535 bytes for the original packet size ping request, ICMPv6 type 128. Each fragment packet has a size of 1232 bytes except the last one with the size of 247 bytes. The offsets of these packets were set correctly.

The packets were sent as explained in different orders.

- 54 fragment packets for a ping request

Figure 28 shows a sample of this packet for this attack initiated from the attacker.

![Fragmentation test case 15](image2)
Chapter 4. Methodology and Experiments

**Fragmentation test case number 16**
Attack type number 16: Flooding the target node with 54 fragment packets with total packet size of 66920 bytes for the original packet size ping request, ICMPv6 type 128. Each fragment packet has a size of 1424 bytes. The offsets of these packets were set correctly and the packets were sent as it is explained in different orders.

47 fragment packets for a ping request
Figure 29 shows a sample of this packet for this attack initiated from the attacker.

![Figure 29: fragmentation6 test case 16](image)

**Fragmentation test case number 17**
Attack type number 17: Flooding the target node with 4 fragment packets with different data within the stream as a ICMPv6 Echo ping request (128) original packet. The sequence of these 4 packets has the same fragmentation ID number and 400 bytes data. In this option, the attacker sends the third fragment packet as a duplication data overlapping totally with the offset of the second fragment packet (with the fake data) with the real data (offset 51 to 100). Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with fake data
3. Duplicate second fragment with real data and TTL=64
4. Third and final fragment

Figure 30 shows a sample of this packet for this attack initiated from the attacker.

![Figure 30: fragmentation6 test case 17](image)
Note: This test case is very close to the test case number 4 as it was explained before. In fact, we could not find any differences from the capture packets with Wireshark between them regarding all 4 types of packet used in both test cases though there were some differences in the source codes for the conditions of generating one of these 4 packets.

**Fragmentation test case number 18**

Attack type number 18: Flooding the target node with 4 fragment packets with different data within the stream as an ICMPv6 Echo ping request (128) original packet. The sequence of these 4 packets has the same fragmentation ID number and 400 bytes data. In this option, the attacker sends the second fragment packet with the real data. The third fragment packet with fake data then is sent to overlap completely with the offset of the second fragment packet (offset 51 to 100).

Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet with fake data and TTL=64
3. Duplicate second fragment packet with real data and TTL=64
4. Third and final fragment

Figure 31 shows a sample of this packet for this attack initiated from the attacker.

![Figure 31: fragmentation6 test case 18](image)

**Fragmentation test case number 19**

Attack type number 19: Flooding the target node with copies of only first fragment packets with size of 408 bytes Data from offset 0 to 50 and ICMPv6 as the next header (58). The M flag is set to “1” indicating the destination node that more packets with the same fragmentation ID will arrive.

This packet each time was sent as follows:

- First fragment

Figure 32 shows a sample of this packet for this attack initiated from the attacker.
Fragmentation test case number 20
Attack type number 20: Flooding the target node with just a fragment packet that could be considered as the second fragment packet with size of 400 bytes data from offset 1051 and ICMPv6 as the next header (58). The M flag is set to “1”. This packet each time was sent as follows:

- Second fragment

Figure 33 shows a sample of this packet for this attack initiated from the attacker.

Fragmentation test case number 21
Attack type number 21: Flooding the target node with just the last fragment packet with size of 400 bytes data from offset 7501 and ICMPv6 as the next header (58). The M flag is reset to “0”. The size of the supposed original packet from which this packet could be its last fragment packet is 56800 bytes. This packet each time was sent as follows:

- Final fragment

Figure 34 shows a sample of this packet for this attack initiated from the attacker.
**Fragmentation test case number 22**

Attack type number 22: Flooding the target node with 3 fragment packets with real data within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the second one with 175 bytes which is not a multiplex of 8 to align the packet format. In this option, the attacker sends the second fragment packet with offset of 175 instead of 51 and finishes at 197 instead of 349. The first fragment packet starts from “0” to 50 and the third fragment packet starts from the 350 offset and ends at 399. The offset format of the second fragment packet leaves a 1000 bytes gap from the first fragment packet and about 1200 bytes to the third fragment packet offset.

Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet (about 1000 bytes offset gap from both other fragment packets)
3. Third and final fragment packet

Figure 35 shows a sample of this packet for this attack initiated from the attacker.

**Fragmentation test case number 23**
Attack type number 23: Flooding the target node with 3 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the second one (175 bytes). In this option, the attacker sends the second fragment packet with offset of 550 instead of 51 and finishes at 572 instead of 1109 without right size (a multiple of 8) to align the packet format. The first fragment packet starts from “0” to 50 and the third fragment packet starts from the 1110 offset and ends at 11159. The offset format of the second fragment packet leaves a 4000 bytes gap from the first fragment packet and about 4300 bytes to the third fragment packet offset. Each time these packets were sent with the following order:

1. First fragment packets
2. Second fragment packet (about 4000 bytes offset gap from both other fragment packets)
3. Third and final fragment packet

Figure 36 shows a sample of this packet for this attack initiated from the attacker.

**Fragmentation test case number 24**

Attack type number 24: Flooding the target node with 3 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the second one (175 bytes). In this option, the attacker sends the second fragment packet with offset of 2050 instead of 51 and finishes at 2071 instead of 4099 without Pad1 formatting to align the packet format. The first fragment packet starts from “0” to 50 and the third fragment packet starts from the offset 4100 and ends at 4149. The offset format of the second fragment packet leaves a 16000 bytes gap from the first fragment packet and about 16200 bytes to the third fragment packet offset. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet (about 16000 bytes offset gap from both other fragment packets)
3. Third and final fragment packet

Figure 37 shows a sample of this packet for this attack initiated from the attacker.
Denial of Service attack in IPv6 networks and Counter measurements

Figure 37: fragmentation6 test case 24

Fragmentation test case number 25
Attack type number 25: Flooding the target node with just one fragment packet with size of “0” bytes data from offset 0 and TCP as the next header (6). The M flag is set to “0”. Each time this packet was sent as follows:

- One fragment packet with 0 byte TCP data

Figure 38 shows a sample of this packet for this attack initiated from the attacker.

Figure 38: fragmentation6 test case 25

Fragmentation test case number 26
Attack type number 26: Flooding the target node with just one fragment packet with size of “1” bytes data from offset 0 and TCP as the next header (6). The M flag is set to “0”. Each time this packet was sent with the following order:

- One fragment packet with “1” byte of TCP data

Figure 39 shows a sample of this packet for this attack initiated from the attacker.
Fragmentation test case number 27

Attack type number 27: Flooding the target node with 3 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the third one (“0” bytes). In this option, the attacker sends the third fragment packet with offset of “0” instead of 100 and the M flag set “0”. The first fragment packet starts from “0” to 50 and the second fragment packet starts from the 51 offset and ends at 100. Each time these packets were sent with the following order:

1. First fragment
2. Second fragment
3. Third fragment packet with offset of “0” and “0” size data

Figure 40 shows a sample of this packet for this attack initiated from the attacker.

Fragmentation test case number 28

Attack type number 28: Flooding the target node with 3 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the third one (“1” bytes). In this option, the attacker sends the third fragment packet with offset of “0” instead of 100 and the M flag set “0”. The first fragment packet
Denial of Service attack in IPv6 networks and Counter measurements

starts from “0” to 50 and the second fragment packet starts from the 51 offset and ends at 99. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet
3. Third fragment packet with offset of “0” and 1 byte size data

Figure 41 shows a sample of this packet for this attack initiated from the attacker.

Figure 41: fragmentation6 test case 28

**Fragmentation test case number 29**

Attack type number 29: Flooding the target node with 3 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these 3 packets has the same fragmentation ID number and 400 bytes data except for the third one (“1” bytes). In this option, the attacker sends the third fragment packet with offset of “1” instead of 100 and the M flag set “0”. The first fragment packet starts from “0” to 50 and the second fragment packet starts from the 51 offset and ends at 99. Each time these packets were sent with the following order:

1. First fragment packet
2. Second fragment packet
3. Third fragment packet with offset of “1” and 1 byte size data

Figure 42 shows a sample of this packet for this attack initiated from the attacker.

Figure 42: fragmentation6 test case 29
Fragmentation test case number 30
Attack type number 30: Flooding the target node each time with 48 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these packets has the same fragmentation ID number and 1416 bytes data except for the first, 47th and the last one. In this option, the attacker sends the last fragment packet with offset of 8191 with 7 bytes data and the M flag set “0”. The first fragment packet has 400 bytes data and ICMPv6 ping request header. Each time these packets were sent with the following order:

1. First fragment
2. 2nd to 74th fragment packets
3. 48th fragment packet with offset of “8191” and 7 byte size data

Figure 43 shows a sample of this packet for this attack initiated from the attacker.

Fragmentation test case number 31
Attack type number 31: Flooding the target node each time with 48 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these packets has the same fragmentation ID number and 1416 bytes data except for the first, 47th and the last one. In this option, the attacker sends the last fragment packet with offset of 8191 with 8 bytes data and the M flag set “0”. The first fragment packet has 400 bytes data and ICMPv6 ping request header. Each time these packets were sent with the following order:

1. First fragment
2. 2nd to 74th fragment packets
3. 48th fragment packet with offset of “8191” and 8 byte size data

Figure 44 shows a sample of this packet for this attack initiated from the attacker.
Fragmentation test case number 32
Attack type number 32: Flooding the target node each time with 48 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these packets has the same fragmentation ID number and 1416 bytes data except for the first, 47th and the last one. In this option, the attacker sends the third fragment packet with offset of 8191 with 9 bytes data and the M flag set “0”. The first fragment packet has 400 bytes data and ICMPv6 ping request header. Each time these packets were sent with the following order:

1. First fragment packet  
2. 2nd to 74th fragment packets  
3. 48th fragment packet with offset of “8191” and 9 bytes size data  

Figure 45 shows a sample of this packet for this attack initiated from the attacker.

Fragmentation test case number 33
Attack type number 33: Flooding the target node each time with 48 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of these packets has the same fragmentation ID number and 1416 bytes data except for the first, 47th and the last one. In this option, the attacker sends the last fragment packet with offset of 8191 with 1414 bytes data 2 and the M flag set “0” while the 47th packet has 174 bytes data size. This data size leaves about 1220 bytes between the 47th
packet and the last one. The size of this packet also is not a multiple of 8. The first fragment packet has 400 bytes data and ICMPv6 ping request header. Each time these packets were sent with the following order:

1. First fragment packet
2. 2nd to 74th fragment packets (47th packet create offset gap with the 48th packet)
3. 48th fragment packet with offset of “8191” and 1414 bytes size data

Figure 46 shows a sample of this packet for this attack initiated from the attacker.

Figure 46: fragmentation6 test case 33

**Fragmentation test case number 34**

Attack type number 34: Flooding the target node each time with 4 fragment packets within the stream as an ICMPv6 payload data for the original packet. The sequence of all the packets within the stream has the same fragmentation ID number. The first packet has 192 bytes data and starts from offset “0” to 24, the second and third packet have both 200 bytes data start from 25 to 49 and 50 to 74 respectively. The fourth has 616 bytes data, starts from offset 75. The fourth packet does not have M flag set to “0”, therefore the rest of the traffic stream packets could be considered as the rest of the fragment packets for one original packet since all packets within flood stream have the same fragmentation ID number of “0x03bbbb”. Each time these packets were sent with the following order:

1. 1st fragment with 192 bytes data with ICMPv6 header
2. 2nd fragment packets with offset of 25 and 200 bytes data
3. 3rd fragment packet with offset of 50 and 200 bytes data
4. 4th fragment packet with offset of 75 and 608 bytes data with M flag set

Figure 47 shows a sample of this packet for this attack initiated from the attacker.
Denial of Service attack in IPv6 networks and Counter measurements

Figure 47: fragmentation6 test case 34

Fragmentation test case number 35

Attack type number 35: Flooding the target node each time with 4 fragment packets within the stream. The sequence of the packets within the stream has hierarchical fragmentation ID number.

The first packet has 192 bytes data, with two fragmentation header consequently with different fragmentation ID as “0x1aaaa” for the first one and “0x01bbbb” for the second one. Both of these headers have M flag set and ICMPv6 header followed as the upper layer header in this packet.

The second packet has 200 bytes data, one fragmentation header with fragmentation ID of “0x1aaaa” referring to the first fragmentation header in the first packet’s header with offset of 26 and M flag set to “0”.

The third packet also has 200 bytes data with offset starting from 50 to 74 and fragmentation ID “0x01bbbb” referring to the second fragmentation header in the first packet’s header and M flag set to “1”.

The fourth and final packet has 608 bytes data, starts from offset 75 with fragmentation ID of “0x01bbbb” and M flag set to “0”. In summary, this test case has two level multi-fragmentations for the 2nd packet for an original ICMPv6 packet sent to the target. Each time these packets were sent with the following order:

Two levels multi fragmentation:

1. 1st fragment packet with 192 bytes data, two fragmentation header (first fragmentation header points at the second packet with different ID) with ICMPv6 header.
2. 2nd fragment packets with 200 bytes data, different ID and reset M flag
3. 3rd fragment packet with offset of 50 and 200 bytes data with M flag set
4. 4th fragment packet with offset of 75 and 608 bytes data with M flag set

Figure 48 shows a sample of this packet for this attack initiated from the attacker.
Fragmentation test case number 36

Attack type number 36: Flooding the target node each time with 4 fragment packets within the stream. The sequence of the packets within the stream has hierarchical different fragmentation ID number. The first packet has 192 bytes data, with three fragmentation header consequently with different fragmentation ID as “0x2aaaa”, “0x2bbbb” and “0x2cccc”. Their M flag set with offset of “0” and ICMPv6 header followed as the upper layer header in this packet.

1. The second packet has 200 bytes data, one fragmentation header with fragmentation ID of “0x2aaaa” with offset of 27 and M flag set to “0”.
2. The third packet also has 200 bytes data with offset starting from 51 to 74 and fragmentation ID “0x02bbbb” and M flag set to “0”.
3. The fourth and final packet has 608 bytes data, start from offset 75 with fragmentation ID of “0x02bbbb” and M flag set to “0”. In summary, this test case is a 3 times multi-fragmentation of the original packet.

Each time these packets were sent with the following order:

Three levels multi fragmentation:
1. 1st fragment packet with 192 bytes data, three fragment headers with ICMPv6 header
2. 2nd fragment packet with 200 bytes data, different ID and reset M flag
3. 3rd fragment packet with offset of 50 and 200 bytes data and reset M flag
4. 4th fragment packet with offset of 75 and 608 bytes data and reset M flag

Figure 49 shows a sample of this packet for this attack initiated from the attacker.
4.4 Experiments

As a course of experimenting DoS attacks in solely IPv6 based networks with lab equipment, a topology was built under the limited systems but effective way to examine the DoS attacks behavior in the network. There had been two topologies used under this process. The main topology of the network connects two different Autonomous System (AS) networks with IPv6 eBGP protocol and each of these networks has a local area. The specification of this topology is shown in the figure 5. Every two nodes were connected with maximum bandwidth of 100Mbps. In this thesis, the AS of 65002 where the attacker’s node is connected is considered as local area while any IPv6 address in the AS of 6501 is defined as remote area.

Note: Window and Ubuntu OSes were fully updated by the time that the attacks were launched. The IPv4 protocol was disabled in all devices to have a pure functional IPv6 in the whole network. The attacks were launched from 1st of December 2014 to 30th of April 2015.

We refer to the bandwidth usage as percentage of 100Mbps in this thesis for each node. Also, the CPU utilization of each end nodes such as PCs is the CPU utilization just to deal with threat of the IPv6 traffic input into their network adaptor interfaces. Please note that ISP2 router has its two Gigabit interfaces (G0/0 and G0/1) connected with Cat5 cable to the nodes and therefore it uses only 10% of its bandwidth (100Mbps) in every tests done in this thesis. In any test case that maximum bandwidth is changed to 10Mbps we clearly state this specification.

**DoS with IPv6 Extension Header Mechanism**

The experimental of the Denial of Service Attack in 2 different model based topology on both scenario 1 and 2 were evaluated. To test the effectiveness of DoS by using Method 1 (Denial6 tool) the following commands were used.
1. Denial attack on the remote area target (PC3):
The attack had been launched from the attacker’s PC to PC3 as a target. Following command was used in the command terminal of the attacker’s PC.

- **Command:** “denial6” -f eth0 2001:1::2 #
  “#” Represents the test cases number from 1 to 7 in this command.

2. Denial on the remote area target and router as a firewall:
The attack had been launched from the attacker’s PC to PC3 as a target with ISP2 as a firewall configured with an explicit ACL as it is shown in the figure 6. The commands used for this attack were the same as above. ISP2 was configured as a firewall with an access-list.

**DoS with IPv6 Fragmentation Mechanism**

The experimental of the fragmentation Attack in 3 different model based on both scenarios 1 and 2 were evaluated. To test the effectiveness of DoS by using Method 2 (fragmentation6 tool) the following commands were used.

1. Fragmentation attack on Local area:

The evaluation of the CPU utilization for PC1, PC2 and the default gateway router ISP2 shown in figure5 was performed. To launch this attack, the attacker’s PC used IPv6 Local area multicasts address as a target. Following commands were used in the command terminal of the attacker’s PC in the flood mode.

- **Command:** fragmentation6 eth0 ff01::2 #

  “#” Represents the test cases number from 1 to 36 in this command.

2. Fragmentation on the remote ISP1 router:

The purpose of this test is to demonstrate how much a remote router as an attack’s destination node for a fragmentation IPv6 traffic had to allocate its resources to defragment the abrupt fragment packets. The attack was launched at the attacker’s PC command terminal and ISP1 was targeted. To achieve this, the following commands were used.

- **Command:** fragmentation6 eth0 2001:3::1 #

  “#” Represents the test cases number from 1 to 36.

3. Fragmentation Attack on Remote Area with Access List:

In this section, a router’s resources necessary to drop a fragmentation flow packets is investigated. The reason is, as a counter measurement for a fragmentation attack after the identification of the attacker’s source IPv6 address could be to stop the traffic with an ACL on a router on the path. That is why an ACL was enabled on the
inbound interface of the router ISP2 against malware IPv6 fragment packets originated for a specific IPv6 source address. The attack was launched at attacker’s PC command terminal targeting PC3. To achieve this, the following commands were used.

- **Command**: `fragmentation6 eth0 2001:1::2 #`

“#” Represents the test cases number from 1 to 36.
5 Results and Analysis

In this section the result of the methods and analysis is demonstrated. In every method and for each test case, routers had been reloaded in advance to refresh all the routers statistics.

Please note that we did not observe any dramatic changes on any of the nodes’ statistics in respect of the CPU utilization or bandwidth usage as the attack did continue over 10 minutes. No attack was launched more than 15 minutes on the topology.

IPv6 CEF service was enabled on both router nodes to help reducing the CPU utilization. In cases that this service was not used it is clearly stated in the test section in this thesis. For the purpose of better investigation of the issues on the routers the command load-interval seconds was set to 30 seconds on their interfaces in most cases.

The connectivity for the whole network was tested with ping and trace route prior of any attack in every scenario. We also did not observe any CPU utilization on the Switch nodes in the test cases therefore the statistics of these nodes were ignored in this thesis.

5.1 Denial with IPv6 Extension Header Mechanism

In this section the outcome of the “denial6” tool on the node resources in respect of the CPU utilization necessary to deal the packets flow of the 7 different test cases is shown according to the network topology.

5.1.1 Denial6 attack on the remote area target (PC3):

The method 1 experiment’s results on the PC3 as a remote target are explained in this section. The detailed results of these test cases as the traffic current was initiated from the attacker node towards the PC3 as the destination node IPv6 address are explained as following:

**Case 1: “denial6” tool test case 1**

**Attacker node:**
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 40.43 second, source 2001:2:: 10 sends an ICMP Echo (ping) request (128), Sequence= 47806, correct checksum with hop limit 255, Payload length=424 bytes with next header: ICMPv6 (58) and IPv6 hop-by-hop option (0) header. The length of
hop-by-hop option is 50 octet (408 bytes) and includes 71 different options to the destination 2001::2. Note that the sum of these 71 options length excides from 408 bytes defined in hop-by-hop extension length. These options are also filled with unknown (undefined) options numbered from 1 to 63, Pad1 and PadN mostly with the length of 4 and no payloads.

The first option of the hop-by-hop option is router alert (repeated only once in the whole options) type 5 and length of 2. The router alert type is Multicast Listener Discovery (MLD). With the hop-by-hop option and router alert enabled, all the routers in the path have to investigate these packets along.

The attacker flooded the target node (PC3) with the same packets and the same number of sequence instantly as it is shown in Figure 50.

![Figure 50: Wireshark Screen shot test case #1](image)

The behavior of the topology nodes along the path as the attacker’s traffic flowed is investigated as follows. Please bear in the mind that we did not notice any differences in which IPv6 CEF was enabled compared with the scenario when this service was not enabled. Therefore, in the following investigation we will just investigate the scenario in which IPv6 CEF was enabled.

**ISP2 node:**

The behavior of ISP2 with IPv6 CEF enabled is investigated as follow:

ISP2 router received IPv6 traffic generated by the attacker machine with the input ratio of 53Mb/s into interface G0/0. The CPU utilization of the router ISP2 got as high as 82%-45% in less than 8 seconds in respond to deal with the attacker IPv6 traffic. The “IPv6 input” process with 45% interrupt CPU utilization was the main cause of this high CPU utilization. The interface (G0/0) in average used 74 middle size pools out of its default 75 to be able to handle the traffic. With the IPv6 packets being buffered inside the interface queue, ISP2 router was unable to handle these packets as fast as they were receiving. Therefore, the whole queued data of this interface had to be flushed about 440000 times within about 20 seconds (when one of our logs was taken on the router) after the attack was launched.

To investigate further on how this router dealt with the input traffic, it was clear that the accepted IPv6 traffic input into ISP2 interface G0/0 (not being flushed in the queueing process inside the interface G0/0) was dropped inside the router itself. No IPv6 traffic was forwarded out interface G0/1 towards the ISP1 router. In other words, ISP2 router decided to drop the whole IPv6 traffic.

The examination of the ISP2 router’s memory in the procedure of dealing with the IPv6 traffic originated from the attacker’s node reveals that the process “pool manager” was the top process to get buffer from memory. This process was holding
157 KB of the memory. It had top 3 process of the memory allocation for the packet headers.

**ISP1 node:**
The analysis of the above statistics and also using Wireshark on the path between ISP1 and ISP2 showed that traffic overloaded solely the ISP2 router and was totally dropped in this router. ISP1 did not receive any of the IPv6 traffic stream on its F0/1 interface from ISP2 router.

**Case 2: “denial6” tool test case 2**

**Attacker:**
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 0.000004 second, source 2001:2::10 sends an ICMP Echo (ping) request (128), Sequence= 47806 with 8 byte data filled with 0 to the 2001:1::2 IPv6 address. The frame length for this packet is 1494 bytes (11952 bits) with the protocol of Ethernet IPv6 and IPv6.nxt:IPv6.dst_opticmpv6: data format. Destination option points at the next header as ICMPv6 (58), with the length of 1424 bytes followed by 11 times pad1. This destination header option then is followed by 235 times “Tunnel Encapsulation Limit” with different length starts from 2 to 63 followed each with 3 times pad1 with value of 0. The variety of the length for Tunnel Encapsulation is invalid. In short, it could be said that this packet is filled with 1424 bytes unknown options plus 8 bytes ICMPv6 data. The attacker flooded the target node (PC3) with the same packets and the same number of sequence instantly as it is shown in Figure 51.

![Figure 51: Wireshark Screen shot test case #2](image)

The behavior of the network nodes as the packet flows in the topology is described as follows:

**ISP2 node:**
In the scenario where IPv6 CEF was disabled on the router the ISP2 router’s CPU reached 93% utilization including 50% interrupt level less than 5 second after the attack was launched. The cause of this very high CPU utilization was “IPv6 Input” based on the output of the CPU utilization monitoring command in the terminal of this router. The reason for this high interrupt level was the interface queue buffering process and the CPU interruption of the big size pool data (in average 74 pool out of the maximum 75 with the size of 1494 bytes were buffered). The router not being able to process the IPv6 incoming traffic with the same ratio as they were inbounding caused the interface overloaded. This interface was reset 3 times within 20 seconds after the attack was launched. The interface also flushed about 2.5% of the incoming
traffic with Selective Packet Discard (SPD) algorithms to avoid more overloading. ISP2 router had to process all the accepted incoming traffic and also was able to forward in average 54% of the incoming traffic outside of its interface (G0/1) towards ISP1. To investigate further and to determine how much effect the forwarding process out of outbound interface might have on the router’s CPU, the cable between ISP1 and ISP2 was removed. The result on the CPU utilization dropped by 20% to the number 72%41%. This reduction could be due to the reason that output interface was generated 9% interrupt in the output buffer on the interface G0/1. The 74%41% also could be considered how much resources this router needed to just drop the packets since it did not have any route to the destination address.

In addition, by checking the log files of the route memory management we indicated that both interfaces of the ISP2 router ran out of hardware pool for “big size” and had been requested for software pool, 181 times for interface G0/0 and 107 times for interface G0/1.

With the service CEF enabled on the ISP2 router, the CPU performance upgraded to 38%37% CPU utilization without having “Input IPv6” as the main case of CPU usage. The reason for this lower CPU usage is that incoming interface (G0/0) did not buffer the IPv6 traffic and could not forward the traffic as it was coming into the outbound interface (G0/1) with fast method. Interface G0/1 also forwarded the traffic with fast method. Therefore, not many interrupts issued and no packet processing was needed for the Router processor from inbound interface. Outbound interface (G0/1) however, needed to buffer the traffic as it was forwarding them towards ISP1 router. The amount of in average 39 big size buffers was used and about 2.5% of the traffic was dropped at this interface.

**ISP1 node:**

In the scenario without IPv6 CEF enabled, the ISP1 router’s CPU reached 96% utilization and 61% interrupt level less than 5 second after the attack was launched caused by “IPv6 input”. The inbound interface of ISP1 (F0/1) had almost the same behavior as ISP2 inbound interface (G0/0). The Interface hardware buffer pool filled with the incoming traffic as the big pool size buffer (75+1 out of 75). This interface however had worse statistic as it was reset 245570 times less than 1 minute after the attack had begun and it dropped about 7.5% of the incoming traffic. The accepted incoming traffic by the interface was treated in process level. The input rate was at the moment sometime less than 1 minute when a log was taken 61Mbps on the F0/1 and the output rate was 50Mbps on the F0/0. This router however had requested less software pool for big size pool compared with ISP2 with the number 71.

With CEF enabled on the ISP1 router, no packets were dropped or flushed due to the buffering in the interfaces of this router. The CPU usage also reduced to 37%36%. This router could forward by Fast Switching method all the incoming IPv6 traffic towards PC3.

**PC3 node:**
In the scenario without IPv6 CEF enabled, PC3 received only 68% of the traffic originated from the attacker machine. In this case, the ping from PC3 to the G0/0 interface (2001:2::1) of the ISP2 was delayed from 0.5ms before the attack to 13ms. In the second scenario however, the PC3 received 93% of the traffic originated but ping to the 2001:2::1 did not show much differences in delay. The traffic also did not cause high CPU usage or memory utilization on this PC.

**Case 3: “denial6” tool test case 3**

**Attacker node:**
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 0.69485 second, Source 2001:2::10 sends an IPv6 packet with payload of 1456 bytes to 2001:1::2. The frame length for this packet is 1510 bytes (12080 bits) with the protocol of Ethernet IPv6 and IPv6.nxt:IPv6.hop_opt:IPv6.dst_opt. The payload length of this packet is 1456 bytes. The hop-by-hop option is with alert “5” and the value of “0” thus indicating MLD option for every node to be investigated. Destination option (60) is the next header and it is repeated 180 times. Each of these destination options has “0” lengths and aligned with 6 times pad1 with the value of “0” (which is not an appropriate way). The last header option is the value of 59 referring to no next header. In general, this packet has a 1452 bytes length size with (180 times) wrong IPv6 header options and wrong pad1 setting for the alignment set for each node in the topology to be processed.

The attacker flooded the target node (PC3) with the same packets and the same number of sequence instantly as it is shown in Figure 52.

![Wireshark Screen shot test case #3](image)

Figure 52: Wireshark Screen shot test case #3

In this test case, the both scenarios which the IPV6 CEF was enabled or disabled, the routers nodes did not show any differences from each other, therefore we just investigated the nodes in the scenario which IPv6 CEF enabled on the routers.

The behavior of the topology nodes along the path as the attacker’s traffic flowed is investigated as follow:

**ISP2 node:**
The inbound interface of this router (G0/0) showed the sign of overloading with traffic by resetting this interface 52 times within 15 second after the attack was launched. This interface was able to forward about 87.5% the incoming IPv6 traffic through fast mode at interrupt level and 12% of the traffic at process level. About 0.5% of the incoming traffic was flushed with SPD algorithm in this interface. The
input queue buffer in the interface filled with 74 big size buffers out of 75 defined by default.

In general, the ISP2 router had asked for big pool size buffer for inbound interface G0/0 190 times and for outbound interface G0/1 115 times. The log files clearly demonstrated that most of the CPU usage was consumed to allocate memory for the IPv6 header needed to be processed. The logs also showed that in total within 20 seconds after the attack was launched, Program counter (PC) had asked for 1937 times memory allocation. This resulted for the memory of the router to be hold by the pool manager process about 1 Mbyte and to be the highest memory allocated in process “getbuffer” about 5Mbyte.

The router CPU utility in this scenario reached the amount of criteria point of 99%/47% within less than 6 seconds after the attack was launched. The main cause of this high amount as it is shown from CPU utilization monitor command in terminal was “IPv6 Input”. The outbound interface (G0/1) of the ISP2 router also had to buffer the outgoing IPv6 traffic about in average 23 out of 40 maximum queue buffer sizes. This interface however dropped about 43% of the incoming traffic (in one sample log file: 8930599 dropped packets out of 20719582 received packets) on the G0/0 interface.

In general ISP2 router was able to forward about 51% of the received IPv6 traffic on the G0/0 out to the G0/1 interface.

**ISP1 node:**
The router ISP1 showed a dramatic CPU utilization as this test case of the attack was launched within 6 seconds. The CPU utilization in average of 97%/51% caused by “IPv6 Input” was captured in this router log files. The inbound interface of this router F0/1 was reset 24984 times since the attack was initiated within about 15 seconds. All the accepted incoming IPv6 traffic on this interface however was preceded by process method and about 5% of this traffic was dropped. The outgoing traffic through interface F0/0 also was processed with process level and not fast mode. In general, almost all the traffic accepted by interface F0/1 was forwarded out by interface F0/0 but all the dropped packets in this router happened in the inbound interface (F0/1) due to input queueing. This interface used in average 75+1 big size pool out of maximum default 75 pool sizes that was available on this interface. This router compared with ISP2 router however holds less memory for the pool management, in fact just about 227Kbytes.

**PC3 node:**
The PC3 “Ubuntu Server” received 48% of the traffic originated from the attacker machine. The CPU of this machine however did not show any high amount of utilization and we believed that all the IPv6 traffic was dropped at the NIC interface without involving any extra memory usage or CPU usage. The ping from this node to the default gateway of the local area (2001:2::1) however was changed from 0.5ms to 27ms in the scenario which IPv6 CEF was enabled and to the total loss in the scenario which IPv6 CEF was not enabled on any of the routers.
Case 4: “denial6” tool test case 4

Attacker node:
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 0.71 second, the source 2001:2::10 sends an ICMP Echo (ping) request, with the sequence=47806 with length of 1510 bytes (12080 bits) of which the payload length is 1456 bytes to 2001:1::2. The IPv6 header for this packet starts with IPv6 hop-by-hop option with the router alert and is followed by 179 duplicated destination header options. These destination options are without any information “NULL” and aligned with 6 times Pad1. The last header portion is ICMP request (128) with 8 bytes Data filled with “0”.

The attacker flooded the target node (PC3) with the same packets and the same number of sequence instantly as it is shown in Figure 53.

<table>
<thead>
<tr>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 0.71</td>
<td>2001:2::10</td>
<td>2001:1::2</td>
<td>ICMPv6</td>
<td>128</td>
<td>seq=47806</td>
</tr>
<tr>
<td>5 0.71</td>
<td>2001:2::10</td>
<td>2001:1::2</td>
<td>ICMPv6</td>
<td>128</td>
<td>seq=47806</td>
</tr>
<tr>
<td>5 0.71</td>
<td>2001:2::10</td>
<td>2001:1::2</td>
<td>ICMPv6</td>
<td>128</td>
<td>seq=47806</td>
</tr>
<tr>
<td>5 0.71</td>
<td>2001:2::10</td>
<td>2001:1::2</td>
<td>ICMPv6</td>
<td>128</td>
<td>seq=47806</td>
</tr>
</tbody>
</table>

Figure 53: Wireshark Screen shot test case #4

The behavior of the topology nodes along the path as the attacker’s traffic flowed is investigated as follows:

ISP2 node:
All the incoming traffic into ISP2 router was dropped within this router therefore no packet was forwarded out from interface G0/1 towards ISP1 router. Continuously, the victim (PC3) did not receive any traffic. This router also showed the same behavior for both scenarios in which either IPv6 CEF was enabled or not.

The inbound interface of this router (G0/0) received 10Mbps IPv6 traffic from the attacker machine and could not deal with these packets at the interrupt level so they had to be processed. This router had to buffer in average 52 packets in its big pool size input buffer. The log files of the router show that most of the router CPU usage was allocated to the pool manager in this router. CPU utilization monitoring commands in these log files showed “IPv6 Input” as the major CPU utilization process which in less than 5 seconds drove the CPU as high as 74%/30%.

Case 5: “denial6” tool test case 5

In this scenario, enabling IPV6 CEF had very useful effects on both routers in the topology and resulted in delivering the whole traffic originated by attacker to the desired victim (PC3). The nodes behavior however along the path is investigated as follows:

Attacker node:
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the Time 0.06 second, source 2001:2::10 sends an IPv6 ICMPv6 packet with length of 32 bytes to 2001:1::2. The frame length for this packet is 86 bytes (688 bits) and the protocol of Ethernet IPv6 is next header pointing to authentication header (51). The authentication header starts with next header indicating of ICMPv6 and length of 16 with 0 values for SPI and ICV with the sequence of 0. ICMPv6 as the final header has the sequence of 47806 with 8 bytes data with the value of 0. The attacker flooded the target node with the same packets and the same number of sequence instantly as it is shown in Figure 54.

The behavior of the topology nodes along the path as the attacker’s traffic flowed is investigated as follows:

**ISP2 node:**
The scenario in which IPv6 CEF was enabled on the ISP2 router, the incoming packets from the attacker machine into interface G0/0 could be forwarded out into the interface G0/1 without causing a drastic CPU utilization (10%/9%). This router also did not drop any traffic due to interface queuing overload and could be able to handle the traffic with its predefine 74 middle size pool on interface G0/0.

In the second scenario without IPv6 CEF enabled however, the interface G0/0 was reset 30303 times about 15 seconds after the attack had launched. The interface G0/0 had suffered from a queue overloading since the router could not route the traffic with the speed as it was input, outside from interface G0/1. All the inbound traffic had to be treated with at process level method and this caused the ISP2 router CPU to 88%/42% less than 6 seconds and stayed at the same level for the rest of the test time.

**ISP1 node:**
The ISP1 router with IPv6 CEF enabled on the router showed 49%/48% CPU utilization less than 15 seconds after the attack was performed. Both the interfaces (F0/0 and F0/1) on this router however did not show any queue overloading, therefore no reset on the interfaces. This router was able to forward all the incoming traffic received from ISP2 towards PC3 without any packet loss.

In the second test without IPv6 CEF being enabled, the ISP1 had a high CPU utilization as much as 95%/55% and the inbound traffic reset the interface F0/0 89046 times and 149547 packets were dropped about 20 seconds at a log sample on the router after the attack had been started. Despite the extreme CPU utilization the
ISP1 could forward the packets that were accepted by the interface F0/0 towards the target PC3.

**PC3 node:**
In the scenario with CEF service PC3 received about 99% of the traffic generated by the attacker machine but in the second scenario only about 38% of the traffic originated by the attacker machine was delivered to the PC3. However, the ping test connectivity from PC3 to the interface G0/0 of ISP2 during attack showed that the topology connectivity was halted for this simple ping. PC3 in both scenario needed only 1% of its CPU to drop the packets.

*Case 6: “denial6” tool test case 6*

**Attacker node:**
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 0.62 second, source 2001::2::10 sends an IPv6 fragment packet with length of 30 bytes to 2001::1::2. The frame length for this packet is 84 bytes (672 bits). The protocol of Ethernet is IPv6 with the next header as hop-by-hop option with router alert as MLD type and indicating fragmentation as the next header. The fragmentation header has offset of “0”, the indication of the first fragment packet for the original packet, M flag set, telling the nodes that there will be more fragmentation packets. The next header is ICMPv6 protocol 128 type(ping request). The payload of this packet has 14 bytes data. The identification of the fragmentation header is “0xffffed9d1”.

The attacker machine however did not send the rest of the fragmentation packets as the rest of the original packet payload but just floods the target node with this packet.

A sample of this traffic flow is shown in Figure 55.

![Figure 55: Wireshark Screen shot test case #6](image)

The behavior of the topology nodes along the path as the attacker’s traffic flowed is investigated as follows:

In this method, the scenarios in which the IPV6 CEF was enabled or disabled, the node did not show any different behavior from each other, therefore we just investigated the nodes with IPv6 CEF enabled on the router scenario.
ISP2 node:
The scenario in which IPv6 CEF was enabled on the ISP2 router, the incoming packets from the attacker machine into interface G0/0 could not be routed out from the interface G0/1 by causing the CPU utilization as high as 78%/44% within 5 seconds after the attack had begun. This router used 74 small size pools for the inbound traffic to cope with this traffic. The inbound interface (G0/0) had flushed many packets with SPD algorithm to avoid overloading on this interface.

Case 7: “denial6” tool test case 7

In this section the behavior of the nodes which IPv6 CEF was enabled on both ISP1 and ISP2 is investigated due to the fact that this service did not show major differences over these two routers compared with the time that attack was launched without this option being enabled on the routers.

Attacker node:
Here a packet sample captured by Wireshark sometime after the attack had begun is described:

At the time 0.785 second since the attack was launched, source 2001:2::10 sends an ICMPv6 packet with length of 424 bytes to 2001:1::2 IPv6 address. The frame length for this packet is 478 bytes (3824 bits). The payload length of this packet is 424 bytes which includes 408 bytes of hop-by-hop option with 74 time unknown options including “Tunnel Encapsulation Limit”, “Pad1” and “Calipso” options. Next header is pointing at ICMPv6 Echo ping request 8 bytes data filled with “0”s and the sequence of 47806. The attacker then flooded the target with the same type of packets as it is shown in the figure 56.

The behavior of the topology nodes along the path as the attacker’s traffic flowed are investigated as follows:

ISP2 node:
The behavior of ISP2 with IPv6 CEF enabled is investigated as follows:

ISP2 router received IPv6 traffic generated by attacker machine with the input rate of 52Mb/s into interface G0/0. This interface in average used 74 middle size pools out of its default 75 to cope with the traffic. This traffic arise the CPU utilization of this router up to 99% which from this number 45% was the interrupt level, less than 7 seconds after the attack was launched.

The input traffic into interface G0/0 was forwarded out of the interface G0/1 with the rate of 31Mbps. The traffic forwarded out of interface G0/1 towards ISP1 router was about 99% of the accepted traffic by interface G0/0. There were not any dropped
or flushed packets that we could observe on the interface G0/1. All the packets in the IPv6 traffic were punt through IPv6 CEF and none of it was dropped. ISP2 router memory management was asked for about 1800 times for packet header memory allocation so this router allocated about 22MB memory to these requests about within 2 minutes after the attack was launched.

**ISP1 node:**
The ISP1 router with IPv6 CEF enabled on the router, showed 96%/49% CPU utilization less than 5 seconds after the attack was performed. The IPv6 traffic was received on interface F0/1 from ISP2 router with the average speed of 17.5 Mbps. This interface used 76 middle size queues to buffer this traffic out of its default 75 queue size. A sample log taken on this router sometime less than 1 minute after the attack began reveals that this interface had to flush the whole input queue on interface (F0/1) about 21000 times. This interface also showed a sign of being overloaded as the number of throttles increased to about 60000 times in the period of 5 minutes. The ISP1 router however, could forward about 99.9% of the input traffic out towards the target (PC3) with the speed of in average 10Mbps from interface F0/0. The interface F0/0 did not need to buffer any traffic in its queue and was not overloaded. This router only dedicated 92KB memory for pool manager service and asked for the memory allocation for the packet headers about 1150 times in the period of 1 minute.

**PC3 node:**
The remote target node (PC3) received the IPv6 traffic launched by the attacker machine with the speed of in average 18Mbps from ISP1 router. This node however was able to deal with the traffic at its NIC card without dedicating extra memory or huge CPU utilization (Less than 2%).

To investigate the effectiveness of this attack it was clear that the ping originated from this node (PC3) to the local area of the ISP2 router (2001:2::1) was just about 25% successful. This result was mostly caused due to the fact of high CPU utilization on both routers in the path. The packets which could get respond from the ISP2 interface G0/0 (2001:2::1) had a delay of in average 59ms compared with the normal condition before the attack, 0.5ms. This means that even the 25%, the successful ping packets, experienced about 120 times more delay during the period of the attack in the path inside the topology.

Table 3 shows the detailed result of these test cases as the traffic current was initiated from the attacker node towards the PC3 as the destination node IPv6 address. As it is shown below, the outcome of these attacks were powerful on the routers. This table shows the result of the test cases which routers did not have by default IPv6 CEF service enabled on them. In this table, the column “Test case #” refers to the attack test case numbers. The “ISP2 Inbound Bandwidth” and “ISP1 Inbound Bandwidth” columns are representing the traffic that was accepted by the routers’ inbound interfaces. The CPU column details the CPU utilization on each node.
Table 3: Denial attacks and their result in the first test

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Bandwidth</th>
<th>ISP2 CPU</th>
<th>ISP1 CPU</th>
<th>Ubuntu CPU</th>
<th>Ubuntu Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58%</td>
<td>85%46%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>94%</td>
<td>93%50%</td>
<td>96%62%</td>
<td>1%</td>
<td>62%</td>
</tr>
<tr>
<td>3</td>
<td>94%</td>
<td>99%43%</td>
<td>96%56%</td>
<td>1%</td>
<td>48%</td>
</tr>
<tr>
<td>4</td>
<td>10.8%</td>
<td>74%33%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>5</td>
<td>10.5%</td>
<td>88%42%</td>
<td>96%55%</td>
<td>1%</td>
<td>4%</td>
</tr>
<tr>
<td>6</td>
<td>10.5%</td>
<td>79%45%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>7</td>
<td>57.92%</td>
<td>99%41%</td>
<td>96%46%</td>
<td>1%</td>
<td>18%</td>
</tr>
</tbody>
</table>

Since most of the above attacks were designed to force the routers to examine every packet at process level, we tried to test them on the IPv6 CEF service which is a switch level forwarding process. Table 4 shows the result for the “denial6” tool different test cases on the topology nodes in respect with their CPU utilization and bandwidth usage with IPv6 CEF enabled on the routers.

Table 4: Denial attacks and their result in the second test

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Bandwidth</th>
<th>ISP2 CPU</th>
<th>ISP1 CPU</th>
<th>Ubuntu CPU</th>
<th>Ubuntu Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58%</td>
<td>84%45%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>94%</td>
<td>38%37%</td>
<td>37%36%</td>
<td>1%</td>
<td>93%</td>
</tr>
<tr>
<td>3</td>
<td>94%</td>
<td>99%47%</td>
<td>96%56%</td>
<td>1%</td>
<td>48%</td>
</tr>
<tr>
<td>4</td>
<td>10.5%</td>
<td>74%30%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>5</td>
<td>10.5%</td>
<td>10%9%</td>
<td>49%48%</td>
<td>1%</td>
<td>10.4%</td>
</tr>
<tr>
<td>6</td>
<td>10.5%</td>
<td>78%44%</td>
<td>0%0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>7</td>
<td>57.92%</td>
<td>99%46%</td>
<td>96%49%</td>
<td>2%</td>
<td>18%</td>
</tr>
</tbody>
</table>

As it is clear from the comparison of above two tables, the IPv6 CEF service only was able to help reducing the packet process timing in the test cases number 2 and 5. The reason for this reduction could be only the fact that packets in these test cases did not have hop-by-hop option header to force the routers to examine them. The consequence of the reduction of CPU utilization was higher traffic rate delivered out of the interface F0/0 of the router ISP1 towards the PC3. It might be safe to conclude that 94Mbps traffic originated from the attacker node in the test case number 2 could have limited impact on the router while IPv6 CEF is enabled but this might not be true about test case number 5. It is because we have to consider that in test case number 5 the traffic originated from the attacker node had the speed of 10.5Mbps. If this traffic could be generated with the same speed ratio as the case number 2, we could have different results.

Further analyses for each of the test cases are described as below.

**Analysis:**

*Test case 1:*
The management of the ISP2 dealing with the IPv6 packets as drop action was a right decision since there was an unknown option inside the IPv6 header based on the defined IPv6 standard protocol [11]. However, the compensation of this action was very heavily in respect of the CPU utilization in this router mostly because of the packets queued inside its input interface. These packets were buffered inside the interface because the big size of hop-by-hop option (408 bytes including 71 options for hop-by-hop option) did not allow the router to drop these packets with the same speed as they were received due to the time consuming of the processing of each individual packet.

This type of the packet formatting with 58Mbps IPv6 traffic stream on a Gigabit interface (5.8% of the physical maximum bandwidth) of the ISP2 could drive the CPU of this router to a very high level. Therefore, we propose that a router has to be restricted with a new standard regarding the large length of the hop-by-hop option which could be a fix length standard defined by IPv6 standard prior to examine the options of this feature.

Test case 2: This test could be a good example of how large extension header as Destination header option without hop-by-hop option could affect the CPU processing of the routers. IPv6 CEF service enabled the routers to not process the whole large, unknown and wrongly formatted packets header (extension header option section) and forward the packets based on their destination IPv6 addresses. Therefore, the CPU utilization dropped from 93% to 38% percent (See Table 3 and Table 4, test case 2). However, also with the 38% CPU utilization we observe a very high interrupt CPU level of 37% and drop in process of forwarding packets out of the interfaces of the routers.

In table 4, test case 2; it can be seen that with lower CPU utilization, both routers were able to forward higher ratio of the traffic to the target node (PC3). The Ubuntu server operation system on the PC3 was able to drop traffic based on the IPv6 standard specification [11] since the destination options were not legitimate without dedication any CPU utilization. However this node was still under the risk of losing 93% of its bandwidth to deal with the attacker’s traffic.

As the result, this test shows that the routers in the network were not transparent to IPv6 extension headers which in fact they had to totally ignore the extension header in absence of the hop-by-hop option header.

Test case 3: The structure of the packet was designed to force the routers in the path to process each packet as they were forwarded them due to the hop-by-hop option enabled with router alert (MLD). The packet header’s extension header also had a very large size due to the many duplication destination option headers. As it is defined in the RFC 2460[11], destination option header should not be presented in the IPv6 extension header more than twice and it has only to be examined by the destination node. However, in this type of packet, destination option header has repeated 180 times followed by 6 times Pad1 alignment (instead of PadN).
As it is shown in Table 3 and Table 4 test case #3, the IPv6 CEF service was unable to lower the CPU utilization on this type of attack. The network topology however was able to deliver about half of the attacker traffic to the destination node (PC3). The reason could be the huge size of extension header plus hop-by-hop option that made the routers under heavy load and result to total loss of connectivity in the network.

**Test case 4:**
The router ISP2 seemed to allocate too much of its resources to implement this action.

To compare this test case with the previous test case (number 3), the packet only had ICMPv6 ping request as the upper layer protocol and one less duplicated destination option while in option 3 there was not any. We believe that as the RFC 2463 [21] Section 5.2 paragraph (4) and section 2.4 paragraph (F) defines, to avoid the DoS caused by enormous IPv6 packets, the router ISP2 limited the rate of this traffic and eventually did not forward any of it to the ISP1 router.

**Test case 5:**
In this method, enabling IPv6 CEF seemed to be a solution for the attack but the CPU utilization on the ISP1 even with this service enabled could cause a real degradation in a productive network where there in more traffic from different nodes in flow. We could not determine what caused this high CPU utilization on this router.

**Test case 6:**
The size of packets in this option has a small length (30 bytes payload). The router ISP2 used the same treatment as it is explained earlier in test case 4 for this category when the ICMPv6 ping request was the data of the packet.

This option might be a good reason to prove that even small packets with hop-by-hop options could cause high impact in respect of the router CPU utilization due to the slow processing of this option.

**Test case 7:**
This form of attack of flooding the target with the big unknown hop-by-hop option without router alert has a huge impact on the routers in the path. Because of the hop-by-hop option header, each router has to investigate each individual packet as comes inside the router to look up the router alert. The idea of this form of attack in theory should not make any huge problem since router alert should be followed exactly as first header option and if not the router has to ignore the header option but in practice we saw that this could severely affect the router resources.

In general, the statistics of the results are shown in the following figures. Please note that in the case numbers 4,5 and 6, the attacker node was unable to generate more traffic on its 100Mbps bandwidth NIC interface due to some unknown reason to us. Comparison of the time when IPv6 CEF was enabled or disable is shown in the Figure 57 below.
Chapter 5. Results and Analysis

Figure 57: Denial6 on routers with and without CEF

The bar chart in the Figure 58 shows the result of these 7 test cases on the PC3 node for each test case. It is obvious in this figure that the traffic flow in test case number 1, 4 and 6 had totally collapsed inside the local area router and did not routed to ISP1 router and the rest of this test case did not have a noticeable effect on the PC3 CPU utilization.

Please note that the tests on the ISP1 lo0 with these 7 test cases were resulted very similar to what is plot on the figure below, therefore we refused to mention the statistics of those tests in this thesis.

Limitation: We were unable to generate identical traffic speed by the attacker for these 7 test cases.
5.1.2 “Denial6” attack on a Router as a firewall.

“denial6” tool attack on the remote area PC3 with ISP2 as a firewall configured with an explicit ACL.

In this section the investigation of the effect for the 7 different test cases with the tool “denial6” as they were explained extensively at section 2.3 is described. The router ISP2 configured as a firewall is based on the topology shown in the figure 6. The local area ISP2 router was configured with an explicit ACL based on the source and destination IPv6 address. We tried to demonstrate how a router with a Gigabit interface would deal with malicious packet traffic streams.

ACL enabled on the ISP G0/0 as inbound traffic with the format shown in the figure 59.

```
ipv6 access-list test-frag
  deny ipv6 host 2001:2::10 host 2001:1::2
  sequence 50 permit ipv6 any any
```

Figure 59: ACL configured on router
Table 5: Denial on the remote PC3

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Bandwidth</th>
<th>ISP2 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58%</td>
<td>23%22%</td>
</tr>
<tr>
<td>2</td>
<td>94%</td>
<td>16%15%</td>
</tr>
<tr>
<td>3</td>
<td>94%</td>
<td>28%25%</td>
</tr>
<tr>
<td>4</td>
<td>10.8%</td>
<td>28%25%</td>
</tr>
<tr>
<td>5</td>
<td>10.5%</td>
<td>23%22%</td>
</tr>
<tr>
<td>6</td>
<td>10.5%</td>
<td>24%22%</td>
</tr>
<tr>
<td>7</td>
<td>57.92%</td>
<td>23%22%</td>
</tr>
</tbody>
</table>

The attacker initiated the packet traffic flow with 7 different test cases towards the remote node PC3 (2001::1::2). The static IPv6 of the attacker as the source IPv6 address was set to 2001:2::10. The ISP2 router had been configured with the explicit ACL defined above to stop abrupt traffic based on the source and destination IPv6 addresses.

The results of these test cases in detail are shown in the table 5. In this table the “Attacker Bandwidth” is represented in percentage, the bandwidth in which the traffic was initiated out of the total 100Mbps available bandwidth on the NIC of this node.

The “ISP2 CPU” column shows the CPU utilization percentage in average that ISP2 showed on its console terminal while each test case was running. The first number percentage is the number of total CPU utilization including the second percentage number while the second percentage number is the level of the I/O interrupts sent form the router interfaces to the CPU.

The Wireshark that was run on the link between the two ISPs did not show any traffic that could bypass this ACL on the ISP2.

Analysis

The analysis of the router CPU usages in the table above can conclude that the majority of the CPU utilization caused by these 7 test cases traffics was caused by the interrupts sent from the inbound interface to the CPU of the router.

5.2 DoS with IPv6 “fragmentation mechanism”

In this section we investigate the effect of the IPv6 fragmentation on the topology nodes with the tool “fragmentation6” with 36 different options (type of attacks). This tool is designed to penetrate the Firewall weaknesses on the network but we used it to launch DoS attack in flood mode. The flood mode uses the same type of packets as explained in section 2.3, each time with different fragmentation identification number. Test cases and models are described in the method 2.
5.2.1 Fragmentation attack on Local area:

In this section, the effect of the 36 different fragmentation test cases as it was explained in section 2.3 examined on the local area of the attacker node with respect of the CPU utilization and the bandwidth. The attacker floods the local area by sending the traffic to the FF02::1 multicast IPv6 destination address to target local area nodes.

Table 6: Details of fragmentation attack on Local area

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Bandwidth</th>
<th>ISP2 Inbound Bandwidth</th>
<th>ISP2 CPU</th>
<th>Ubuntu CPU</th>
<th>Windows CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>91.2%</td>
<td>61.5%</td>
<td>99%57%</td>
<td>5.8%</td>
<td>13%</td>
</tr>
<tr>
<td>2</td>
<td>91.2%</td>
<td>73%</td>
<td>99%57%</td>
<td>6.0%</td>
<td>16%</td>
</tr>
<tr>
<td>3</td>
<td>91.2%</td>
<td>70%</td>
<td>99%58%</td>
<td>17.9%</td>
<td>25%</td>
</tr>
<tr>
<td>4</td>
<td>91.2%</td>
<td>69.7%</td>
<td>99%57%</td>
<td>14.3%</td>
<td>18%</td>
</tr>
<tr>
<td>5</td>
<td>91.2%</td>
<td>74.3%</td>
<td>99%57%</td>
<td>14.5%</td>
<td>20%</td>
</tr>
<tr>
<td>6</td>
<td>91.2%</td>
<td>76.6%</td>
<td>99%57%</td>
<td>6%</td>
<td>13%</td>
</tr>
<tr>
<td>7</td>
<td>91.6%</td>
<td>78.6%</td>
<td>99%58%</td>
<td>8.3%</td>
<td>12%</td>
</tr>
<tr>
<td>8</td>
<td>93.6%</td>
<td>73.5%</td>
<td>99%57%</td>
<td>8.4%</td>
<td>13%</td>
</tr>
<tr>
<td>9</td>
<td>93.6%</td>
<td>84%</td>
<td>99%32%</td>
<td>9.6%</td>
<td>11%</td>
</tr>
<tr>
<td>10</td>
<td>93.6%</td>
<td>68%</td>
<td>99%30%</td>
<td>7.5%</td>
<td>12%</td>
</tr>
<tr>
<td>11</td>
<td>93.6%</td>
<td>4.3%</td>
<td>99%12%</td>
<td>8.8%</td>
<td>26%</td>
</tr>
<tr>
<td>12</td>
<td>93.8%</td>
<td>3.3%</td>
<td>99%10%</td>
<td>7.3%</td>
<td>26%</td>
</tr>
<tr>
<td>13</td>
<td>93.6%</td>
<td>65.3%</td>
<td>64%35%</td>
<td>6.7%</td>
<td>5%</td>
</tr>
<tr>
<td>14</td>
<td>93.6%</td>
<td>81.2%</td>
<td>67%35%</td>
<td>6.7%</td>
<td>2%</td>
</tr>
<tr>
<td>15</td>
<td>93.6%</td>
<td>60.6%</td>
<td>64%34%</td>
<td>3.2%</td>
<td>1%</td>
</tr>
<tr>
<td>16</td>
<td>94%</td>
<td>78%</td>
<td>55%30%</td>
<td>6.5%</td>
<td>5%</td>
</tr>
<tr>
<td>17</td>
<td>91%</td>
<td>72%</td>
<td>99%57%</td>
<td>13.8%</td>
<td>20%</td>
</tr>
<tr>
<td>18</td>
<td>91%</td>
<td>67%</td>
<td>99%57%</td>
<td>8.7%</td>
<td>16%</td>
</tr>
<tr>
<td>19</td>
<td>91%</td>
<td>70.7%</td>
<td>98%56%</td>
<td>10.9%</td>
<td>28%</td>
</tr>
<tr>
<td>20</td>
<td>91%</td>
<td>75.4%</td>
<td>98%56%</td>
<td>10.2%</td>
<td>31%</td>
</tr>
<tr>
<td>21</td>
<td>91%</td>
<td>72.4%</td>
<td>91%64%</td>
<td>10.2%</td>
<td>30%</td>
</tr>
<tr>
<td>22</td>
<td>90%</td>
<td>61.9%</td>
<td>98%58%</td>
<td>8.2%</td>
<td>25%</td>
</tr>
<tr>
<td>23</td>
<td>90.5%</td>
<td>50.5%</td>
<td>98%58%</td>
<td>9.0%</td>
<td>19%</td>
</tr>
<tr>
<td>24</td>
<td>91%</td>
<td>59%</td>
<td>98%59%</td>
<td>9.1%</td>
<td>23%</td>
</tr>
<tr>
<td>25</td>
<td>73%</td>
<td>20.7%</td>
<td>99%54%</td>
<td>9.5%</td>
<td>41%</td>
</tr>
<tr>
<td>26</td>
<td>73%</td>
<td>8.6%</td>
<td>99%53%</td>
<td>8.3%</td>
<td>46%</td>
</tr>
<tr>
<td>27</td>
<td>89.7%</td>
<td>89.7%</td>
<td>99%58%</td>
<td>7.4%</td>
<td>19%</td>
</tr>
<tr>
<td>28</td>
<td>89.8%</td>
<td>74.9%</td>
<td>99%59%</td>
<td>8.7%</td>
<td>19%</td>
</tr>
<tr>
<td>29</td>
<td>89.7%</td>
<td>89.7%</td>
<td>98%59%</td>
<td>10.3%</td>
<td>16%</td>
</tr>
<tr>
<td>30</td>
<td>94%</td>
<td>71.8%</td>
<td>59%31%</td>
<td>7.0%</td>
<td>9%</td>
</tr>
<tr>
<td>31</td>
<td>94%</td>
<td>81.3%</td>
<td>60%31%</td>
<td>5.6%</td>
<td>4%</td>
</tr>
<tr>
<td>32</td>
<td>93.8%</td>
<td>77.3%</td>
<td>59%31%</td>
<td>7.4%</td>
<td>3%</td>
</tr>
<tr>
<td>33</td>
<td>94%</td>
<td>84.9%</td>
<td>59%31%</td>
<td>7.5%</td>
<td>5%</td>
</tr>
<tr>
<td>34</td>
<td>90%</td>
<td>29.3%</td>
<td>99%18%</td>
<td>8.2%</td>
<td>17%</td>
</tr>
<tr>
<td>35</td>
<td>90%</td>
<td>3.3%</td>
<td>99%26%</td>
<td>9.4%</td>
<td>25%</td>
</tr>
<tr>
<td>36</td>
<td>90%</td>
<td>3.3%</td>
<td>99%18%</td>
<td>8.6%</td>
<td>25%</td>
</tr>
</tbody>
</table>
The Wireshark was run on the attacker node and it did not receive any reply back for any of the 3 nodes on the local area for any of these fragmentation test cases. Table 6 displays the result of these test cases as the traffic current was initiated from the attacker node towards the local area as the destination node. The “ISP2 CPU” column shows the CPU utilization percentage on average that ISP2 showed on its console terminal while each test case was running. The first percentage number is the number of total CPU utilization including the second percentage number while the second percentage number is the level of the I/O interrupts sent form the router interfaces to the CPU.

Moreover, the column “Test case #” is referring to the attack type numbers explained earlier in this category in section 2.3. Please note that we were unable to define the exact process CPU utilization on the Ubuntu machine for IPv6 input traffic. Tools such as “nethogs” and “tcpdump” were used to trace the abrupt inbound IPv6 but these tools were unable to identify the process number for this traffic, thus we could not determine exactly what process was handling the traffic inbound in NIC. The CPU utilization for Ubuntu in the table 6 is the difference of the CPU utilization before and after each attack was launched on the local area.

**Analysis:**

As it is shown in the table 6, the outcome of these attacks was powerful on the ISP2 router’s CPU utilization, but it had very low impact on the Ubuntu machine (PC1). The strongest test cases for the PC1 (Windows 7) node were the test case number 26 and 25. The CPU utilization on the windows was reached as high as 46% and 41% respectively.

The bandwidth columns in the table are representing the percentage of the traffic out of 100MBbps that was initiated by the attacker node and was accepted by the routers’ inbound interfaces. In these tests, we observed a high packet drop on the interface G0/0 (the inbound interface of ISP2) mostly due to overload especially in test case number 11, 12, 26, 35 and 36. This could be caused by the slow processing of reassembling the queued fragment packets on this router interface which caused overflow and thus flushing the entire queue. This attack could also be considered a fragmentation attack on a local router.

The bar chart in the figure 60 shows the effect of the 36 fragmentation test cases on the local area. The data is sorted based on 3 level criteria, highest router CPU utilization as the first level, highest PC2 CPU utilization as the second level and finally highest PC1 CPU utilization for the sort algorithm.

It can be seen that the number of very high router CPU utilization is most likely but Windows and Ubuntu OSes have a relatively low utilization. The impact of these attacks except for the last 6 test cases in this figure was connectivity loss between PC1 and PC3 to the remote area network in few cases but most likely drove the delay up to 100 times from 0.5ms to 50ms for a simple ipv6 ping request. Overall, we can see a clear IPv6 fragmentation DoS issue in the local area as the first hop security in the IPv6.
5.2.2 Fragment on a router.

Fragmentation on the ISP1 lo0:

In this section the impact of the router ISP1 resources in respect of the CPU utilization necessary to defragment the packets flow of the 36 fragmentation test cases is shown according to the topology defined in the Figure 5. These test cases are explained extensively in the section 2.3. Table 7 shows the detailed result of these test cases as the traffic current was initiated from the attacker node towards the ISP1 loopback0 as the destination node IPv6 address. This router prior of any decision making had to reassemble these packets and consequently, this process had a great deal of the load on it. As it is shown below, the outcome of these attacks is robust on the router’s CPU.

In this table, the column “Test case #” refers to the attack test case numbers. The “ISP2 Inbound Bandwidth” and “ISP1 Inbound Bandwidth” columns are representing the traffic that was accepted by the routers’ inbound interfaces. In the test cases we observed high packet drop on the interface F0/1 (the inbound interface of ISP1) mostly due to overload. This could be caused by the slow processing of reassembling the queued fragment packets on this router interface which made overflow and thus flushing the entire queue. The “ISP2 CPU” and “ISP1 CPU” columns show the CPU utilization percentage in average that each router showed on their console terminals while each test case was running. The first number percentage is the number of total CPU utilization including the second percentage number while the second percentage number is the level of the I/O interrupts sent form the router interfaces to the CPU.
Table 7: Details of fragmentation attack on remote ISP1 router

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Outbound Bandwidth</th>
<th>ISP2 Inbound Bandwidth</th>
<th>ISP2 CPU</th>
<th>ISP1 Inbound Bandwidth</th>
<th>ISP1 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>91.2%</td>
<td>90.6%</td>
<td>53%32%</td>
<td>22.4%</td>
<td>95%48%</td>
</tr>
<tr>
<td>2</td>
<td>91.2%</td>
<td>88%</td>
<td>54%32%</td>
<td>21.5%</td>
<td>95%48%</td>
</tr>
<tr>
<td>3</td>
<td>91.2%</td>
<td>77%</td>
<td>13%12%</td>
<td>18%</td>
<td>95%47%</td>
</tr>
<tr>
<td>4</td>
<td>91.2%</td>
<td>78.5%</td>
<td>13%13%</td>
<td>18%</td>
<td>95%48%</td>
</tr>
<tr>
<td>5</td>
<td>91.2%</td>
<td>80%</td>
<td>13%13%</td>
<td>19%</td>
<td>95%47%</td>
</tr>
<tr>
<td>6</td>
<td>91.2%</td>
<td>81%</td>
<td>12%11%</td>
<td>20.7%</td>
<td>95%47%</td>
</tr>
<tr>
<td>7</td>
<td>91.6%</td>
<td>83.5%</td>
<td>12%12%</td>
<td>17.5%</td>
<td>95%48%</td>
</tr>
<tr>
<td>8</td>
<td>93.6%</td>
<td>82%</td>
<td>12%12%</td>
<td>21%</td>
<td>95%47%</td>
</tr>
<tr>
<td>9</td>
<td>93.6%</td>
<td>85%</td>
<td>11%10%</td>
<td>27%</td>
<td>97%33%</td>
</tr>
<tr>
<td>10</td>
<td>93.6%</td>
<td>84.5%</td>
<td>11%10%</td>
<td>15%</td>
<td>98%22%</td>
</tr>
<tr>
<td>11</td>
<td>93.6%</td>
<td>79%</td>
<td>10%9%</td>
<td>1%</td>
<td>99%1%</td>
</tr>
<tr>
<td>12</td>
<td>93.8%</td>
<td>83.6%</td>
<td>8%8%</td>
<td>0.9%</td>
<td>99%1%</td>
</tr>
<tr>
<td>13</td>
<td>93.6%</td>
<td>79%</td>
<td>9%9%</td>
<td>49%</td>
<td>96%64%</td>
</tr>
<tr>
<td>14</td>
<td>93.6%</td>
<td>85.5%</td>
<td>10%9%</td>
<td>53.5%</td>
<td>96%62%</td>
</tr>
<tr>
<td>15</td>
<td>93.6%</td>
<td>83%</td>
<td>10%9%</td>
<td>54.5%</td>
<td>96%63%</td>
</tr>
<tr>
<td>16</td>
<td>94%</td>
<td>82%</td>
<td>9%8%</td>
<td>54%</td>
<td>96%61%</td>
</tr>
<tr>
<td>17</td>
<td>91%</td>
<td>77%</td>
<td>13%13%</td>
<td>18%</td>
<td>95%48%</td>
</tr>
<tr>
<td>18</td>
<td>91%</td>
<td>84%</td>
<td>13%13%</td>
<td>19.5%</td>
<td>91%48%</td>
</tr>
<tr>
<td>19</td>
<td>91%</td>
<td>86%</td>
<td>14%12%</td>
<td>20%</td>
<td>95%48%</td>
</tr>
<tr>
<td>20</td>
<td>91%</td>
<td>85.7%</td>
<td>13%13%</td>
<td>20%</td>
<td>95%48%</td>
</tr>
<tr>
<td>21</td>
<td>91%</td>
<td>81%</td>
<td>13%13%</td>
<td>25%</td>
<td>94%60%</td>
</tr>
<tr>
<td>22</td>
<td>90%</td>
<td>83%</td>
<td>14%13%</td>
<td>17.7%</td>
<td>95%48%</td>
</tr>
<tr>
<td>23</td>
<td>90.5%</td>
<td>84%</td>
<td>15%13%</td>
<td>17%</td>
<td>94%48%</td>
</tr>
<tr>
<td>24</td>
<td>91%</td>
<td>83%</td>
<td>14%14%</td>
<td>18%</td>
<td>94%50%</td>
</tr>
<tr>
<td>25</td>
<td>73%</td>
<td>62%</td>
<td>50%50%</td>
<td>2.9%</td>
<td>95%52%</td>
</tr>
<tr>
<td>26</td>
<td>73%</td>
<td>59.5%</td>
<td>50%50%</td>
<td>2%</td>
<td>96%42%</td>
</tr>
<tr>
<td>27</td>
<td>89.7%</td>
<td>86%</td>
<td>16%15%</td>
<td>13.7%</td>
<td>95%47%</td>
</tr>
<tr>
<td>28</td>
<td>89.8%</td>
<td>82%</td>
<td>16%15%</td>
<td>13.8%</td>
<td>95%47%</td>
</tr>
<tr>
<td>29</td>
<td>89.7%</td>
<td>84%</td>
<td>16%15%</td>
<td>14%</td>
<td>95%47%</td>
</tr>
<tr>
<td>30</td>
<td>94%</td>
<td>87%</td>
<td>10%8%</td>
<td>54%</td>
<td>96%59%</td>
</tr>
<tr>
<td>31</td>
<td>94%</td>
<td>83%</td>
<td>9%8%</td>
<td>55.5%</td>
<td>96%59%</td>
</tr>
<tr>
<td>32</td>
<td>93.8%</td>
<td>89.6%</td>
<td>8%8%</td>
<td>57%</td>
<td>96%58%</td>
</tr>
<tr>
<td>33</td>
<td>94%</td>
<td>85.5%</td>
<td>8%8%</td>
<td>56%</td>
<td>96%59%</td>
</tr>
<tr>
<td>34</td>
<td>90%</td>
<td>85.8%</td>
<td>15%14%</td>
<td>2.3%</td>
<td>99%2%</td>
</tr>
<tr>
<td>35</td>
<td>90%</td>
<td>83%</td>
<td>15%14%</td>
<td>2%</td>
<td>99%5%</td>
</tr>
<tr>
<td>36</td>
<td>90%</td>
<td>84%</td>
<td>15%15%</td>
<td>2.9%</td>
<td>99%7%</td>
</tr>
</tbody>
</table>

The second part of this test was to run these attacks with 10% of the traffic flow in previous part on the remote ISP1 router. Therefore, the outbound bandwidth of the attacker’s node was limited to 10Mbps with “ethtool” and the “auto-negotiation” on the NIC interface of this node was set to “OFF”. We needed to investigate how maximum 10Mbps traffic could affect a remote router which is a very reasonable traffic speed nowadays over internet for a user.
Table 8 represents the result of the test cases as the traffic current was initiated from the attacker node towards the ISP1 loopback0 as the destination node. Please note that ISP1 router had its interface F0/1 at its maximum capability bandwidth of 100Mbps. The details for each test case can be found in the table below.

Table 8: Details of fragmentation attack on remote ISP1 router

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Outbound Bandwidth</th>
<th>ISP1 Inbound Bandwidth</th>
<th>ISP1 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.8%</td>
<td>7%</td>
<td>81%/38%</td>
</tr>
<tr>
<td>2</td>
<td>8.8%</td>
<td>3.2%</td>
<td>75%/36%</td>
</tr>
<tr>
<td>3</td>
<td>8.8%</td>
<td>7%</td>
<td>83%/47%</td>
</tr>
<tr>
<td>4</td>
<td>8.8%</td>
<td>7.8%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>5</td>
<td>8.8%</td>
<td>8.2%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>6</td>
<td>8.8%</td>
<td>8.8%</td>
<td>82%/44%</td>
</tr>
<tr>
<td>7</td>
<td>8.8%</td>
<td>4.4%</td>
<td>80%/44%</td>
</tr>
<tr>
<td>8</td>
<td>8.8%</td>
<td>8.3%</td>
<td>80%/45%</td>
</tr>
<tr>
<td>9</td>
<td>9.6%</td>
<td>8%</td>
<td>39%/20%</td>
</tr>
<tr>
<td>10</td>
<td>9.6%</td>
<td>8.1%</td>
<td>44%/19%</td>
</tr>
<tr>
<td>11</td>
<td>9.6%</td>
<td>1.6%</td>
<td>99%/2%</td>
</tr>
<tr>
<td>12</td>
<td>9.6%</td>
<td>5.4%</td>
<td>99%/1%</td>
</tr>
<tr>
<td>13</td>
<td>9.6%</td>
<td>2%</td>
<td>31%/18%</td>
</tr>
<tr>
<td>14</td>
<td>9.6%</td>
<td>2.3%</td>
<td>31%/18%</td>
</tr>
<tr>
<td>15</td>
<td>9.6%</td>
<td>4.7%</td>
<td>31%/18%</td>
</tr>
<tr>
<td>16</td>
<td>9.6%</td>
<td>4.5%</td>
<td>27%/16%</td>
</tr>
<tr>
<td>17</td>
<td>8.8%</td>
<td>6.7%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>18</td>
<td>8.8%</td>
<td>8.3%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>19</td>
<td>8.8%</td>
<td>8.3%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>20</td>
<td>8.8%</td>
<td>8.5%</td>
<td>86%/48%</td>
</tr>
<tr>
<td>21</td>
<td>8.8%</td>
<td>6.4%</td>
<td>76%/48%</td>
</tr>
<tr>
<td>22</td>
<td>8.8%</td>
<td>4.8%</td>
<td>87%/50%</td>
</tr>
<tr>
<td>23</td>
<td>8.8%</td>
<td>7.3%</td>
<td>87%/51%</td>
</tr>
<tr>
<td>24</td>
<td>8.8%</td>
<td>7.2%</td>
<td>87%/51%</td>
</tr>
<tr>
<td>25</td>
<td>7%</td>
<td>5.3%</td>
<td>96%/56%</td>
</tr>
<tr>
<td>26</td>
<td>7%</td>
<td>4.1%</td>
<td>96%/49%</td>
</tr>
<tr>
<td>27</td>
<td>8.8%</td>
<td>3.4%</td>
<td>90%/48%</td>
</tr>
<tr>
<td>28</td>
<td>8.8%</td>
<td>6.8%</td>
<td>88%/50%</td>
</tr>
<tr>
<td>29</td>
<td>8.8%</td>
<td>6.6%</td>
<td>88%/50%</td>
</tr>
<tr>
<td>30</td>
<td>9.6%</td>
<td>8.1%</td>
<td>36%/19%</td>
</tr>
<tr>
<td>31</td>
<td>9.6%</td>
<td>2.8%</td>
<td>29%/17%</td>
</tr>
<tr>
<td>32</td>
<td>9.6%</td>
<td>6.8%</td>
<td>30%/17%</td>
</tr>
<tr>
<td>33</td>
<td>9.6%</td>
<td>7.8%</td>
<td>29%/17%</td>
</tr>
<tr>
<td>34</td>
<td>8.8%</td>
<td>6.1%</td>
<td>76%/47%</td>
</tr>
<tr>
<td>35</td>
<td>8.8%</td>
<td>5.8%</td>
<td>99%/18%</td>
</tr>
<tr>
<td>36</td>
<td>8.8%</td>
<td>4.4%</td>
<td>98%/24%</td>
</tr>
</tbody>
</table>
**Analysis:**

With the further investigation of the table 7, test cases number 11, 12, 34, 35 and 36 had different peaks. The amount of the inbound traffic into the ISP1 in respect of the amount of the CPU usage is very different from other test cases. It is clear that traffic flow with about 1Mbps speed could drive the CPU utilization of this router as high as 99% and more with very low interrupt level. After checking the log files of this router, it was clear that it had very high packet drop at the inbound traffic and a very high number of overflow. For instance the test case number 35 at one of the sample logs had about 1068643 time “throttle count” (reset the interface) on the interface F0/1 and about 12663048 packets loss from input queue on this interface and 76 packets in the input queue at the moment when the log was taken sometime less than 2 minutes after this attack was launched. This sample can be seen in the figure 61 below which is the output of the “show interface summary” command on the router terminal. We could see the same pattern for the rest of these test cases. This high ratio of the packet drop could be caused by the high CPU utilization that accepted packet by the interface (the ones that were not dropped or flushed in the interface) generated on the router.

<table>
<thead>
<tr>
<th>Interface</th>
<th>IHQ</th>
<th>IQD</th>
<th>OHQ</th>
<th>OQD</th>
<th>RXBS</th>
<th>RXPS</th>
<th>TXBS</th>
<th>TXPS</th>
<th>TRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>* FastEthernet0/0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>* FastEthernet0/1</td>
<td>75</td>
<td>12663048</td>
<td>0</td>
<td>0</td>
<td>2350000</td>
<td>593</td>
<td>0</td>
<td>0</td>
<td>1068643*</td>
</tr>
<tr>
<td>Serial10/0/0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Serial10/0/1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>* Loopback0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 61: Screen shot on ISP1 for test case #35

Hereafter, when the attack numbers 4, 11 and 12 were run more than 3 minutes, the BGP connection between two router nodes was totally lost resulted to the connectivity loss of the whole topology between the local area and the remote area. Figure 62 was taken for test case number 4 which shows the BGP status on the ISP2 router. Even though the BGP connection was restored we had seen fluctuation on establishment for this protocol.

As it is determined from table 7, the fragmentation attack destined for a router can be very concerning regarding the CPU utilization. All test cases resulted to the CPU utilization as lower as 90% level and as high as 99% usage which could be very catastrophic for the network connectivity. The connectivity issues such as ping between local area to the remote area and also delay up to 1000 times were experienced right after the attack was launched less than 10 seconds on ISP1 router mostly on the case number 11, 12, 34, 35 and 36.
The bar chart in figure 63 shows the statistics of the impact of the fragmentation attack on the router ISP1 and ISP2. The chart is sorted from the test cases in which the lower load was caused on the local router ISP2 CPU but by highest overload on the remote router CPU. Based on this chart bar we concluded that test cases 12, 11, 34, 35 and 36 could cause more damages respectively.

As it can be seen even with this low speed traffic stream in 7 test cases, the router ISP1 had to dedicate its CPU from 97% and more to cope with the abrupt fragmentation IPv6 traffics. One important statistic that can be highlighted in the table 8 is the test case number 12. The attacker node in this case flooded 9.6Mbps traffic flow towards ISP1 router with 8 bytes of ICMPv6 code 128 data payload inside a packet with continuously 175 times fragmentation header with the same format as a single packet. The ISP1 router received only 5.4Mbps traffic from its interface F0/1 but that caused only 1% interrupt from the router interface to the CPU and the IOS of the router had to spend 98% of the CPU for the process “IPv6 Input” to manage it. It was also noticed that the router in the target’s path which was not aimed as a destination for the abrupt traffic was effected with the fragment packets. In the test cases 1, 2, 25, and 26 where a packet with TTL=1 was sent as a part of the original packet, the first router(ISP2) in the path had to dedicated approximately more than 50% of its CPU to handle them due to the fact that these packets had to be dropped for their TTL that become 0.
Chapter 5. Results and Analysis

The bar chart in figure 64 shows the ISP1 router CPU utilization comparison for each of 36 fragmentation test cases between two different traffic speeds originated by the attacker node. There are two bar charts for each test case in this figure. The red one is the number of the CPU usage for the 100Mbps traffic speed and the blue one is the number of the CPU usage for 10 Mbps traffic speed. This chart is sorted with two level of sorting algorithm based on the higher to the lower CPU utilization for 10Mbps as the first level and 100Mbps as the second level.

As the result of this demonstration one can conclude that in the first 6 test cases (11, 12, 35, 36, 25 and 26) the limitation of the original test case traffic speed to 10% results in the same impact on the router with higher than 90% CPU usages.

![Fragmentation on ISP1 and Traffic Speed](image)

Figure 64: Fragmentation on ISP1 router with speed sorted

5.2.3 Fragmentation on a Router as a Firewall.

Fragmentation attack on PC3 with an explicit ACL on ISP2:
In this section the investigation of the effect for the IPv6 fragmentation on a router configured as a firewall is carried out based on the topology shown in the figure 6. The local area ISP2 router was configured with an explicit ACL based on the source and destination IPv6 address. We tried to demonstrate how a router with a Gigabit interface would deal with a malicious fragment packets traffic stream.

Figure 65 shows this ACL enabled on the ISP2 G0/0 as inbound traffic.

![ACL configured on router](image)

Figure 65: ACL configured on router
The attacker initiated the packet traffic flow with 36 different test cases as it was explained in section 2.3 towards the remote node PC3(2001:1::2). The static IPv6 of the attacker as the source IPv6 address was set to 2001:2::10. The ISP2 router was configured with the explicit access-list defined above and had to stop abrupt traffic based on the source and destination IPv6 addresses. The test was run for two different bandwidth configured on the NIC of the attacker node for 100Mbps as the first test and 10Mbps as the second test.

Table 9: Details of fragmentation attacks on the ACL

<table>
<thead>
<tr>
<th>Test case #</th>
<th>Attacker Outbound Bandwidth</th>
<th>ISP2 Inbound Bandwidth</th>
<th>ISP2 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>91.2%</td>
<td>84.1%</td>
<td>65%46%</td>
</tr>
<tr>
<td>2</td>
<td>91.2%</td>
<td>86%</td>
<td>66%46%</td>
</tr>
<tr>
<td>3</td>
<td>91.2%</td>
<td>85.5%</td>
<td>33%32%</td>
</tr>
<tr>
<td>4</td>
<td>91.2%</td>
<td>84.8%</td>
<td>33%32%</td>
</tr>
<tr>
<td>5</td>
<td>91.2%</td>
<td>95%</td>
<td>33%32%</td>
</tr>
<tr>
<td>6</td>
<td>91.2%</td>
<td>84.5%</td>
<td>33%32%</td>
</tr>
<tr>
<td>7</td>
<td>91.6%</td>
<td>88.2%</td>
<td>32%31%</td>
</tr>
<tr>
<td>8</td>
<td>93.6%</td>
<td>87.5%</td>
<td>32%31%</td>
</tr>
<tr>
<td>9</td>
<td>93.6%</td>
<td>89.6%</td>
<td>18%18%</td>
</tr>
<tr>
<td>10</td>
<td>93.6%</td>
<td>88.7%</td>
<td>18%18%</td>
</tr>
<tr>
<td>11</td>
<td>93.6%</td>
<td>87.3%</td>
<td>27%26%</td>
</tr>
<tr>
<td>12</td>
<td>93.8%</td>
<td>88%</td>
<td>25%24%</td>
</tr>
<tr>
<td>13</td>
<td>93.6%</td>
<td>91.6%</td>
<td>18%17%</td>
</tr>
<tr>
<td>14</td>
<td>93.6%</td>
<td>91.6%</td>
<td>18%17%</td>
</tr>
<tr>
<td>15</td>
<td>93.6%</td>
<td>90.7%</td>
<td>18%17%</td>
</tr>
<tr>
<td>16</td>
<td>94%</td>
<td>88.3%</td>
<td>15%15%</td>
</tr>
<tr>
<td>17</td>
<td>91%</td>
<td>90%</td>
<td>32%32%</td>
</tr>
<tr>
<td>18</td>
<td>91%</td>
<td>89%</td>
<td>33%32%</td>
</tr>
<tr>
<td>19</td>
<td>91%</td>
<td>88.4%</td>
<td>33%32%</td>
</tr>
<tr>
<td>20</td>
<td>91%</td>
<td>88.5%</td>
<td>33%32%</td>
</tr>
<tr>
<td>21</td>
<td>91%</td>
<td>86.5%</td>
<td>33%32%</td>
</tr>
<tr>
<td>22</td>
<td>90%</td>
<td>86.1%</td>
<td>38%38%</td>
</tr>
<tr>
<td>23</td>
<td>90.5%</td>
<td>85.8%</td>
<td>39%38%</td>
</tr>
<tr>
<td>24</td>
<td>91%</td>
<td>85%</td>
<td>38%38%</td>
</tr>
<tr>
<td>25</td>
<td>73%</td>
<td>40.7%</td>
<td>94%93%</td>
</tr>
<tr>
<td>26</td>
<td>73%</td>
<td>40.7%</td>
<td>95%93%</td>
</tr>
<tr>
<td>27</td>
<td>89.7%</td>
<td>84.7%</td>
<td>44%43%</td>
</tr>
<tr>
<td>28</td>
<td>89.8%</td>
<td>85.5%</td>
<td>43%43%</td>
</tr>
<tr>
<td>29</td>
<td>89.7%</td>
<td>86.6%</td>
<td>44%38%</td>
</tr>
<tr>
<td>30</td>
<td>94%</td>
<td>91%</td>
<td>16%15%</td>
</tr>
<tr>
<td>31</td>
<td>94%</td>
<td>90%</td>
<td>16%15%</td>
</tr>
<tr>
<td>32</td>
<td>93.8%</td>
<td>92.8%</td>
<td>16%15%</td>
</tr>
<tr>
<td>33</td>
<td>94%</td>
<td>90.5%</td>
<td>18%15%</td>
</tr>
<tr>
<td>34</td>
<td>90%</td>
<td>87%</td>
<td>34%31%</td>
</tr>
<tr>
<td>35</td>
<td>90%</td>
<td>89%</td>
<td>40%40%</td>
</tr>
<tr>
<td>36</td>
<td>90%</td>
<td>85%</td>
<td>40%39%</td>
</tr>
</tbody>
</table>
The result of these test cases in detail is shown in the table 9. In this table the “Attacker Outbound Bandwidth” is represented in percentage, the bandwidth in which the traffic was initiated out of the total 100Mbps available bandwidth on the NIC of this node. The “ISP2 Inbound Bandwidth” is the speed of which the traffic in percentage out of 100Mbps (only 10% of the bandwidth available on this interfaces physically) was accepted by the ISP2 router interface G0/0. This does not include the packets which were dropped or any reason such as flushing or dropping by this interface due to reasons such as overflowing. Please note that ISP2 has all of its interfaces as Gigabit per second as physical interfaces but the percentage in this table is from 100Mbps. The “ISP2 CPU” column shows the CPU utilization percentage in average that ISP2 showed on its console terminal while each test case was running. The first percentage number is the number of total CPU utilization including the second percentage number while the second percentage number is the level of the I/O interrupts sent form the router interfaces to the CPU.

The Wireshark that was run on the link between the two ISPs did not show any traffic that could bypass this ACL on the ISP2.

**Analysis:**

To analyze the router CPU usages in the table 9, test cases number 25 and 26 show very different behavior with the percentages as high as 94% and 95% respectively. This clearly highlights their impact on the router. These test cases also had the highest packets drop in the interface of the router and about 30% before they were blocked by the ACL. The test case number 1 and 2 could also be considered different from the other cases due to having about 20% CPU processing timing in dealing with the “IPv6 input” traffic by the IOS of the router.

Figure 66 shows the CPU Utilization for each individual test case with the defined access-list needed to drop the malicious traffic. This bar chart is sorted from higher to lower CPU usage on the router. It is very clear that the test cases number 25 and 26 are the strongest type of attack to be dealt with. These test cases were flooding the target with just one fragment packet as the original packet with 0 byte data for TCP and 1 byte data for TCP protocol respectively.

Test cases number 1 and 2 also could be considered as very disturbing in a productive environment where a router would need its CPU more than 60% to drop these types of traffics. Test cases number 1 and 2 were flooding the target with 4 fragment packets for an original IPv6 packet fragment in 3 parts. In the test case number 1, the second packet was sent with TTL=1 and the third packet as the duplication of the second packet with TTL=64 and different data payload while in the case number 2 the second packet was sent with TTL=64 but the third packet was sent as the duplication of the second packet with different payload and the TTL=1.
Figure 66: CPU Utilization for ACL on ISP2 sorted

Figure 67 shows the result of the fragmentation with 36 test cases sorted by algorithm in two levels based on the higher CPU utilization and lower traffic speed initiated by the attacker node on the PC3 node. This figure shows how much CPU the router needed to stop an attacker machine with a known IPv6 address (2001:2::10) and its bandwidth. It is obvious that the test case number 25 and 26 despite the lower traffic stream caused higher damage on the router.

It could be concluded that despite the fact that if an attacker’s IPv6 address (out of the huge proportion of IPv6 range addresses) is known to us, in some of the test cases as it was explained before even with an explicit ACL based on the source and destination IPv6 address, the DoS could be achieved on the router which tries to eliminate this attack.

Figure 67: CPU Utilization and the traffic with access-list on ISP2 sorted
Chapter 6

6 Conclusions

The project done for this thesis was based on investigating the strength of some possible methods of launching the DoS on future solely IPv6 networks with open source tools. The DoS attacks with flooding abrupt IPv6 network traffic from one attacker node was performed with various test cases on different parts of network areas. The monitoring and analysis were done on these traffics captured by Wireshark and routers status via CLI and then statistics were built for each method and their test cases. The result of these tests on different areas and different nodes was shown in detail and the main reasons of their causes were analyzed. These analyses were considered from the attacker interface along the path in each node to the target interface. The test cases packet structure was built according to the captured packets at the attacker’s outbound interface and the source code of the tools.

Type of DoS attacks experimented in this thesis includes IPv6 extension header with 7 and IPv6 fragmentation mechanism with 36 test cases.

In the experiments where the examination of the extension header mechanism was evaluated, the results of the test cases 3, 7 and 5 showed that a packet with hop-by-hop header with router alert and 180 times repeated headers or a packet with great size hop-by-hop header with 43 times unknown options and finally authentication header with ICMPv6 protocol respectively had high impact on the remote router. Furthermore, in the test cases number 4 and 6 a packet had hop-by-hop header with router alert and 179 times repeated destination option, or hop-by-hop header and the first fragment packet as an original ICMPv6 protocol packet had maximal impact on the local area router. These packets were not able to be forwarded out of the local area router.

In the experiments where the evaluation of the fragmentation mechanism was examined, abrupt traffics were originated with two different bandwidth limitations, 100Mbps and 10Mbps from the attacker node. According to the results, the test cases number 11, 35, 36, 26 and 25 caused severe effect on the routers whereas 137 fragmentation header as a complete packet, two levels multi fragmentation, three levels multi fragmentation, one fragment packet with 1 byte TCP and one fragment packet with 0 byte TCP were the packets assembly respectively used to cause maximal DoS attack on the routers. The effects were high enough for the router to become unfucntional.

This thesis also had taken into consideration the implementation of an explicit access-list on a router as a counter measurement. When an IPv6 access-list implemented on a router in order to stop the abrupt traffic types based on the source and destination addresses, in the extension header evaluation experiments the utilization of router CPU to block the traffic at the most severe cases was
approximately 28%, while in the fragmentation evaluation experiments this number was reached as high as 95%.

We concluded that the router nodes were most impacted against the abrupt IPv6 traffic and in some cases caused total halt in network functionality due to the maximum CPU utilization.

IPv6 does not have a restrict number of the extension headers (it can be as long as the whole size of an IPv6 packet) and also the fact that the sequence of options within a header must be processed absolutely in the order they appear in the header. Therefore, in the method 1 it was observed that malware packets with very large length of many headers embedded, especially those which have hop-by-hop option enabled could force a router to dedicate a huge amount of CPU to deal with them. A packet with many destination option headers with hop-by-hop option enabled could cause the same effect on a router either locally or remotely.

Furthermore, a small size packet with one AH header had a high impact on the remote router. The reason of high CPU utilization could lay on the encryption and decryption CPU intensive mechanism but it is unclear for us why the remote router was more affected to this type of packet than the local router.

In addition, as it is defined in the RFC standard, a packet with hop-by-hop option enabled but without router alert that should have followed as an immediate option had to be dropped in a router but in our test we observed that the router did not discard them.

In the fragmentation evaluation experiments, the result of DoS on a router was extreme. It was observed that with a very low traffic bandwidth the highest CPU utilization was achieved. The main reason of this high CPU utilization was due to the interrupts that were issued by the interfaces of the routers. The high CPU utilization and the fact that input-queue of the router interfaces were full, resulted to high ratio of the packet drop at the router interfaces as well. It was also noticed that the router in the target’s path which was not aimed as a destination for the abrupt traffic was effected by the fragment packets. In the test cases where a packet with TTL=1 was sent as a part of the original packet, the first router(ISP2) in the path had to dedicated approximately more than 50% of its CPU to handle them due to the fact that these packets had to be dropped for their TTL that has become 0. Consequently based on the results of this method, IPv6 fragmentation mechanism can lead to severe DoS attacks.

Our evaluation could show that an access-list cannot be found as a solution to handle the attacks tested in this thesis. We also recommend defining a fix length of the hop-by-hop option and discarding packets with length’s size bigger than this standard prior the examination of the hop-by-hop options to avoid many recursive illegitimate headers. Further research and IPv6 header standards are necessary to satisfy the IPv6 security before IPv6 is fully deployed. It also might be necessary to make changes in the IPv6 protocol algorithms for fragmentation mechanism in respect of the way that routers handle it.
Future work:
We identify two major research challenges:

- Could defining a fix length of the hop-by-hop option be a necessity in the IPv6 protocol?

- An attacker can force a router in the path to a destination node to evaluate a fragment packet (evaluate a fragment packet is a heavy load on a router) by either set the TTL equal to the number of hops away from the target router or by enabling hop-by-hop option in a fragment packet. How can a router be protected against it?

It follows that detailed planning is needed by the network or security administrator to set up a trustworthy IPv6 environment. Since IPv6 has its particular addressing pattern, it faces the network devices such as routers with new security challenges.
Acknowledgments

We would like to thank Halmstad Cisco Network Academy lab for their support during the process of this work.
References


[Access date: March 2015]

[Access Date: March 2015]

[Access Date: 10-feb-2015].


[Access Date: 23-feb-2015].

A: Source: https://www.thc.org/thc-IPv6/README 
[Access Date: 1 January 2015].

B: Source: https://www.thc.org/#about
[Access Date: Feb 2015]


Available: http://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xhtml
[Access Date: 03-April-2015]


**Appendix 1**

### Matrix for Tables

Table 1: Next header Code values

Table 2: DoS attack types at various OSI model.

Table 3: Denial attacks and their result in the first test

Table 4: Denial attacks and their result in the second test

Table 5: Denial on the remote PC3

Table 6: Details of fragmentation attack on Local area

Table 7: Details of fragmentation attack on remote ISP1 router

Table 8: Details of fragmentation attack on remote ISP1 router

Table 9: Details of fragmentation attacks on the ACL

### Table of Figures

Figure 1: McAfee Labs Threat report and Statistics [7]

Figure 2: Sophisticated attacks versus Intruder Knowledge time frame [8]

Figure 3: IPv6 datagram’s extension header process [12]

Figure 4: IPv6 datagram with two extension headers

Figure 5: Network Topology

Figure 6: Network Topology with Firewall

Figure 7: Denial packet format for Test case 1

Figure 8: Denial packet format for Test case 2

Figure 9: Denial packet format for Test case 3

Figure 10: Denial packet format for Test case 4

Figure 11: Denial packet format for Test case 5

Figure 12: Denial packet format for Test case 6

Figure 13: Denial packet format for Test case 7

Figure 14: fragmentation6 test case1

Figure 15: fragmentation6 test case2

Figure 16: fragmentation6 test case3

Figure 17: fragmentation6 test case4

Figure 18: fragmentation6 test case5

Figure 19: fragmentation6 test case6

Figure 20: fragmentation6 test case7

Figure 21: fragmentation6 test case8

Figure 22: fragmentation6 test case9

Figure 23: fragmentation6 test case10

Figure 24: fragmentation6 test case 11

Figure 25: fragmentation6 test case 12
Denial of Service attack in IPv6 networks and Counter measurements

Figure 26: fragmentation6 test case 13 .................................................. 33
Figure 27: fragmentation6 test case 14 .................................................. 34
Figure 28: fragmentation6 test case 15 .................................................. 34
Figure 29: fragmentation6 test case 16 .................................................. 35
Figure 30: fragmentation6 test case 17 .................................................. 35
Figure 31: fragmentation6 test case 18 .................................................. 36
Figure 32: fragmentation6 test case 19 .................................................. 37
Figure 33: fragmentation6 test case 20 .................................................. 37
Figure 34: fragmentation6 test case 21 .................................................. 38
Figure 35: fragmentation6 test case 22 .................................................. 38
Figure 36: fragmentation6 test case 23 .................................................. 39
Figure 37: fragmentation6 test case 24 .................................................. 40
Figure 38: fragmentation6 test case 25 .................................................. 40
Figure 39: fragmentation6 test case 26 .................................................. 41
Figure 40: fragmentation6 test case 27 .................................................. 41
Figure 41: fragmentation6 test case 28 .................................................. 42
Figure 42: fragmentation6 test case 29 .................................................. 42
Figure 43: fragmentation6 test case 30 .................................................. 43
Figure 44: fragmentation6 test case 31 .................................................. 44
Figure 45: fragmentation6 test case 32 .................................................. 44
Figure 46: fragmentation6 test case 33 .................................................. 45
Figure 47: fragmentation6 test case 34 .................................................. 46
Figure 48: fragmentation6 test case 35 .................................................. 47
Figure 49: fragmentation6 test case 36 .................................................. 48
Figure 50: Wireshark Screen shot test case #1 ...................................... 52
Figure 51: Wireshark Screen shot test case #2 ...................................... 53
Figure 52: Wireshark Screen shot test case #3 ...................................... 55
Figure 53: Wireshark Screen shot test case #4 ...................................... 57
Figure 54: Wireshark Screen shot test case #5 ...................................... 58
Figure 55: Wireshark Screen shot test case #6 ...................................... 59
Figure 56: Wireshark Screen shot test case #7 ...................................... 60
Figure 57: Denial6 on routers with and without CEF ................................ 65
Figure 58: Denial6 on PC3 ................................................................. 66
Figure 59: ACL configured on router .................................................... 66
Figure 60: Fragmentation on FF02::1 sorted ...................................... 70
Figure 61: Screen shot on ISP1 for test case #35 ................................... 73
Figure 62: Screenshot taken on ISP2 router .......................................... 74
Figure 63: Fragmentation attack on ISP1 router sorted .......................... 74
Figure 64: Fragmentation on ISP1 router with speed sorted .................... 75
Figure 65: ACL configured on router .................................................... 75
Figure 66: CPU Utilization for ACL on ISP2 sorted ............................... 78
Figure 67: CPU Utilization and the traffic with access-list on ISP2 sorted .... 78
Appendix 2

ISP1 Router Configuration:

Current configuration : 1411 bytes 
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption 
!
hostname ISP1
!
boot-start-marker
boot-end-marker 
!
no aaa new-model
memory-size iomem 5 
!
!
ip cef 
!
!
no ip domain lookup
ip host PAGENT-SECURITY-V3 76.98.34.63 73.65.0.0
ipv6 unicast-routing
ipv6 cef 
!
multilink bundle-name authenticated 
!
voice-card 0
no dspfarm
!
interface Loopback0
ip address 2.2.2.2 255.255.255.255
ipv6 address 2001:3::1/64
!
interface FastEthernet0/0
no ip address
duplex auto
speed auto
ipv6 address 2001:1::1/64
ipv6 enable
!
interface FastEthernet0/1
no ip address
load-interval 30
duplex auto
speed auto
ipv6 address 2001:1/64
ipv6 enable
!
interface Serial0/0/0
no ip address
shutdown
no fair-queue
clock rate 64000 
!
interface Serial0/0/1
no ip address
shutdown
clock rate 125000 
!
router bgp 65001
  bgp router-id 1.1.1.1
  no bgp default ipv4-unicast
  bgp log-neighbor-changes
  neighbor 2001::2 remote-as 65002 
  address-family ipv6
  neighbor 2001::2 activate
  network 2001:1::/64
  network 2001:3::/64
  exit-address-family 
  !
  !
ip http server
no ip http secure-server 
!
control-plane
!
line con 0
  exec-timeout 0 0
logging synchronous
line aux 0
line vty 0
password cisco
login 
line vty 1 4
login 
!
scheduler allocate 20000
1000 
!
!
end
ISP2 Router Configuration:

Current configuration : 1942 bytes
!
version 15.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption !
hostname ISP2
! boot-start-marker
boot-end-marker !
no aaa new-model !
ip cef !
no ip domain lookup
ipv6 unicast-routing
ipv6 cef !
multilink bundle-name authenticated !
license udi pid CISCO2911/K9 sn FCZ172360QF !
redundancy !
interface Loopback0
ip address 1.1.1.1 255.255.255.255
ipv6 address 2001:4::1/64 !
interface Embedded-Service-Engine0/0
no ip address shutdown !
interface GigabitEthernet0/0
no ip address load-interval 30
duplex auto speed auto
ipv6 address 2001:2::1/64 ipv6 enable !
Interface GigabitEthernet0/1
no ip address load-interval 30
duplex auto speed auto
ipv6 address 2001:2/64 ipv6 enable !
interface GigabitEthernet0/2
no ip address shutdown
duplex auto speed auto !
interface Serial0/0/0
no ip address shutdown
clock rate 2000000 !
interface Serial0/0/1
no ip address shutdown
clock rate 2000000 !
router bgp 65002
bgp router-id 2.2.2.2
bgp log-neighbor-changes
no bgp default ipv4-unicast
neighbor 2001::1 remote-as 65001 !
address-family ipv4 exit-address-family !
address-family ipv6
network 2001:2::/64
network 2001:4::/64
neighbor 2001::1 activate exit-address-family !
ip forward-protocol nd !
no ip http server
no ip http secure-server !
ipv6 access-list test-frag
deny ipv6 host 2001:2::10
host 2001:1::2
sequence 50 permit ipv6 any any !
control-plane !
line con 0
exec-timeout 60 0
logging synchronous
line aux 0
line 2
no activation-character
no exec
transport preferred none
transport input all
transport output pad telnet rlogin lapb
ta mop udptn v120 ssh
stopbits 1
line vty 0
password cisco
login
transport input all
line vty 1 4
login
transport input all !
scheduler allocate 20000
1000 !
end
Mahmood Nazari:
Contact: mahnaz12 at student.hh.se
        m.nazari at engineer.com

Lokesh Galla:
Contact: lokgal13 at student.hh.se
        lokeshgalla4u at gmail.com