Islamic University of Gaza
Faculty of Engineering
Computer Engineering Department

Hacker Location Detection System (HLDS) For IPv6 Routers

Eng. Tayseer J. El Khateeb

Thesis submitted in partial fulfllment of the requirements for the degree of Master of Science in Computer Engineering Program

Thesis Supervisor
Prof. Mohammad Mikki

Gaza, Palestine

(October, 2012)

THIS PAGE INTENTIONALLY LEFT BLANK

Abstract

The Enhanced IPv6-next generation routers and IPv6-network architectures are two of the most important aspects of the Internet today. The users' demand of new and more complex services, as well as significant constraints imposed by current technologies, make the design of IPv6 next-generation routers hard and challenging. IPv6 next-generation router architectures are expected to provide capabilities, such as availability, reliability, modularity and performance scalability, as well as the support of antivirus, anti spam, and antispyware, which spans from energy-efficiency requirements to advanced levels of flexibility and programmability.

In this thesis we address the problem of network monitoring and automatic detection of vulnerabilities, weaknesses, and shortcomings of IPv4-based networks. Our main objective is to generate Software Router Plug-ins (SRPs) that will be deployed as advanced and flexible tools to IPv6 next generation routers to increase the network security. SRPs are one of the most intelligent anti-malware technique that we will be used in IPv6 routers. We propose an original methodology called Hacker Location Detection System (HLDS). HLDS is capable of both detection of hackers and identification of their location based on their IP address, MAC address, and geographic location. HLDS is a Software Router Plug-in (SRP) that will be used in IPv6 routers to provide Internet security. Since hackers are able to change IP and MAC addresses of their computers and NIC respectively, our proposed system does not depend only on IP or MAC addresses but also on location of hackers. Hence, even if hackers change these IP and/or MAC addresses our proposed system is capable of tracing these hackers.

We validate the proposed HLDS by developing experiments to measure its performance. We used some performance metrics to measure the performance of HLDS which include accuracy, False positive rate, and False negative rate. We also compare the performance results of HLDS with performance of similar tools in the literature, namely Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree. Performance results show that HLDS has a better accuracy, less false positive rate, and less false negative rate than these programs.

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

This Master's thesis has been done for Islamic University of Gaza, and for the engineering faculty in the computer engineering department.

I want to thank professor Dr. Eng. Mohammad Mikki my supervisor, for the opportunity to work in this thesis. His knowledge about the field provided me with plenty of information.

My thankfulness also goes to my mother and the spirit of my father.

I would also like to thank my wife who encouraged me strongly to complete this thesis and study in spite of the great burdens that I went through.

I want to thank my brothers for their support and patience during my studies and thesis.

I also want to thank my sisters for their support during my studies and thesis. Their appreciation towards the education has pushed me forward in my studies.

Finally, I would like to thank my children whom I am preoccupied with and for their time in the summer holidays.

Table of Contents

List of Figures

List of Table

List of Acronyms

ABSs	Anomaly-Based Systems
ACL	Access Control List
AIS	Artificial Immune System
ANN	Artificial Neural Network
CIFS	Common Internet File System
CSRF	Cross-Site Request Forgery
CGA	Cryptographically Generated Addresses
DAD	Duplicate Address Detection
DHCPv6	Dynamic Host Configuration Protocol version 6
DNS	Domain Name System
DoS	Denial of Service
DPI	Deep Packet Inspection
ETH	Ethernet
EUI	Extended Unique Identifier
FSM	Finite State Machine
GIS	Geographic information system
GPS	Global Positioning System
GRNN	General Regression Neural Network
HLDS	Hacker Location Detection System
HTTPS	Hypertext Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
ICMPv6	Internet Control Message Protocol Version 6
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IPsec	Internet Protocol Security
IPv6	Internet Protocol Version 6
LAN	Local Area Network
MAC	Media Access Control
MLP	Multilayer Perceptron
MTU	Maximum Transmission Unit
NAS	Network Attached Storage

NAT	Network Address Translator
NIDS	Network Intrusion Detection System
NIPS	Network Intrusion Prevention System
NSAP	Network Service Access Point
NVRAM	Non-Volatile Random-Access Memory
OCSVM	One-Class Support Vector Machine
OSPF	Open Shortest Path First
OUI	Organizationally Unique Identifier
PNN	Probabilistic Neural Network
QoS	Quality of Service
RAM	Random-Access Memory
RFC	Request for Comments
RH0	Routing Header Type 0
RIR	Regional Internet Registry
SBSs	Signature-Based Systems
SeND	Secure Neighbor Discovery Protocol
SIP	Session Initiation Protocol
SIEM	Security Information and Event Management
SNMP	Simple Network Management Protocol
SMO	Support Vector Machines
SLAAC	Stateless Address Autoconfiguration
SRP	Software Router Plug-in
SSL	Secure Socket Layer
SVM	Support Vector Machines
TCM	Transductive Confidence Machine
TCP	Transmission Control Protocol
TLS	Transport Layer Security
USB	Universal Serial Bus
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
WAN	Wide Area Network
XSS	cross Site Scripting

Chapter One
Introduction

1.1 IPv6 Next-Generation Routers

The Enhanced IPv6-next generation routers and IPv6-network architectures are two of the most important aspects of the Internet today. The users' demands of new and more complex services, as well as significant constraints imposed by current technologies, make the design of IPv6 next-generation routers hard and challenging. IPv6 next-generation router architectures are expected to provide capabilities, such as availability, reliability, modularity and performance scalability, as well as the support of antivirus, anti spam, and antispyware, which spans from energy-efficiency requirements to advanced levels of flexibility and programmability. These features are heavily required to lead network equipment design and to support the evolution of the IPv6 next-generation Internet, where value-added services will work through specific applicative overlays on the top of the IP layer. Thus, future network nodes are supposed to flexibly include multi-layer functionalities at both control and data planes, by allowing to map and to effectively interface them towards the network layer capabilities (e.g., QoS, reliability, multicast, etc.). In such a scenario, router architectures based on general purpose hardware and open source software (e.g., Linux and FreeBSD) recently regained a remarkable consideration from the industrial and academic communities [1].

Internet Protocol version 6 (IPv6) [2] will replace Internet Protocol version 4 (IPv4) [3] as the protocol backbone of our packet-switched networks, devices, and eventually the Internet. In fact, with the last IPv4 /8 address blocks being issued in early February 2011 and the 8 June 2011 world IPv6 day, the start-again stop-again conversion to IPv6 is rapidly approaching [4]. This movement towards IPv6 brings to light the need for knowledge of IPv6 infrastructure and how to secure it (e.g., Network Intrusion Detection, Firewalls, Access Control Lists). To date, these remain mostly little explored domains. Industry (e.g., Google, Akamai, Yahoo, Ericsson, and Cisco) has made the move towards IPv6 with events like World IPv6 day, and now it is time for everyone else to get on board. This includes the federal government and Department of Defense in USA infrastructure, which must be prepared for IPv6 in order to prevent having to catch up on the trend.

Computer networks are also required to support security techniques such as cryptographic techniques, authentication, SSL, etc. Business-critical applications such as SAP, Voice over IP, and video conferencing depend heavily on the quality of network connections. It is vital to monitor network connections, detect vulnerabilities, virus packets, spyware packets, and spam packets early and proactively and inform the parties involved. Current security in computer networks is not acceptable due to software/hardware vulnerabilities, spyware, malware, viruses, hacking, intrusion, etc. In addition, IPv4 routers have numerous technical problems. The IPv4 address space is limited to 4 billion IP addresses which have been already consumed. We run out of IP addresses unless we move to using Ipv6 routers. In practice, the only option for building large new networks is to use IPv6 routers.

1.2 Thesis Objectives and Motivation

The main goal of this master thesis is to develop IPv6 plug-ins that enhance network security and detect vulnerabilities, virus packets, spyware packets, spam packets in computer networks.

The motivation of the thesis is the lack or low security support in IPv4 –based networks and the small number of IP addresses allocated in the IPv4 protocol.

As in most security related topics the analysis always assumes the presence of a determined attacker, because this scenario poses the greatest threat. However, the concept of attacks also encompasses the possibility of a mis-configuration or a bad implementation, all without malicious intent but with similar negative effects on networks or nodes. For example, some popular operating systems send rogue router advertisement messages, thus essentially performing man-in-the-middle attacks.

1.3 Thesis Questions

Thesis asks primary questions: How we can find the IPv6 attacks? And how we can prevent these attacks? The primary objectives of the thesis can be achieved through answering the secondary questions:

- What are the current IPv6 attacks?
- What are the current router intrusion detection and prevention system?
- How can we enhance IPv6 next generation routers?

To achieve thesis objectives we use a controlled virtual IPv6 Scapy (Security Power tool) that permits repeatable testing in a sanitized environment. Scapy is a powerful interactive packet manipulation program which we use to program the SRP (Software Routers Plug-in).

1.4 Thesis Contribution

The contribution of this thesis is the design of an original network monitoring system that automatically detects network vulnerabilities, weaknesses, and shortcomings of IPv6-based networks. The designed system is called Hacker Location Detection System (HLDS) which is a

Software Router Plug-in (SRP) that could be deployed as advanced and flexible tool to IPv6 next generation routers to increase the network security. HLDS is capable of both detection of hackers and identification of their location based on their IP address, MAC address, and geographic location. HLDS is used in IPv6 routers to provide Internet security. Since hackers are able to change IP and MAC addresses of their computers and NIC respectively, our proposed system does not depend only on IP or MAC addresses but also on location of hackers. Hence, even if hackers change these IP and/or MAC addresses our proposed system is capable of tracing these hackers.

The idea of HLDS is based on the knowledge of the geographical location hackers from previous experience. This location is used by hackers to hack network packets passing through them. When a user sends a packet to another user and the packet passes through the hacker's router then HLDS will alarm the communicating users if HLDS has the location of the hacker stored in Hacker Location Database (HLDB) server used by HLDS. HLDB is a database that stores known hacker locations based on previous experience. It is updated regularly to reflect the new situation of hackers and their locations. The accuracy of HLDS is based on the accuracy of information stored about hackers in HLDB and its constant update. HLDS is able to detect different attacks including Man-in-the-Middle (MITM), mail phishing, IP spoofing, packet sniffing. These attacks attack data confidentiality, authenticity and integrity. Hence, HLDS demonstrates if computer networks have any security vulnerabilities. HLDS at the highest level of detail is composed of four modules:

- **HLDB Server Module:** HLDB Server Module contains the Hacker Location Database (HLDB) that includes the known locations of the hackers. HLDB is to be provided by a third party.
- **HLDB Update Module:** HLDB Update Module updates the HLDB based on the New hacker location information received.
- **Traceroute Module:** Traceroute Module is used to trace the route of the packet that is being monitored. It lists all routers that the packet passed through from source to destination. Information provided by traceroute includes the IP addresses of these routers as well as their geographic locations.
- **HLDS Monitor Module:** HLDS Monitor Module monitors the packets paths and generates an alarm if any packet that is being monitored has passed through a hacker router. This is determined by comparing the locations of the routers that the monitored packet passes through with the locations of the routers in the HLDB.

1.5 Thesis Structure

The rest of the thesis is organized as follows: Chapter two presents the concepts, format, configuration and the functionality of IPv6. Chapter three explains computer network attacks and threats properties, their types and effects. In addition, Chapter three compares between IPv6 and IPv4 security. Chapter four presents Network Intrusion Detection Systems, Network Intrusion Prevention Systems, and Scapy security tool. Chapter five presents our proposed methodology and approach the Hacker Location Detection System (HLDS). In Chapter five we also present how the test beds are installed and configured. HLDS block diagram, flowchart, and pseudo-code are also presented in Chapter five. Chapter six validates our proposed methodology and gives some experimental results of HLDS. It compares the performance of HLDS with that of some of the similar and popular programs in the literature. Finally, chapter seven concludes the thesis report and presents some future work.

Chapter Two

IPv6 Background

2.1 IPv6 Overview

IPv6 or Internet Protocol Version 6 is the next generation protocol for the Internet. It's designed to provide several advantages over current Internet Protocol Version 4 (or IPv4). Both IPv6 and IPv4 define network layer protocol i.e., how data is sent from one computer to another computer over packet-switched networks such as the Internet [5].

Some important changes from IPv4 to IPv6 are [6]:

- Larger Address Space: With 128 bits an IPv6 address is four times as long as an IPv4 address with 32 bits and allows for vastly more addressable nodes and networks. With IPv4 address exhaustion imminent this is the most significant incentive to deploy IPv6.
- Basic and Extension Headers: The number of fields in the IPv6 header has been reduced to a minimum to make packet processing by intermediate routers more efficient (see Figures 2.1 and 2.2).

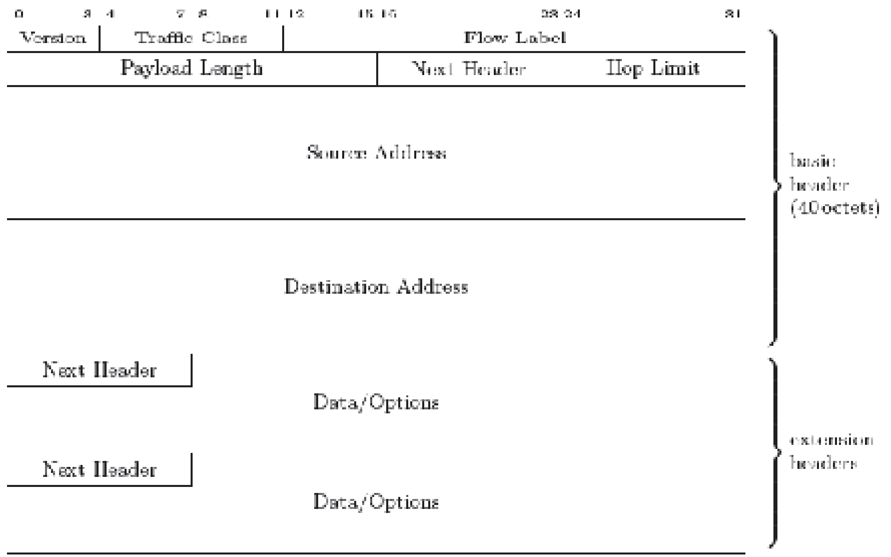


Figure 2.1.: IPv6 header format

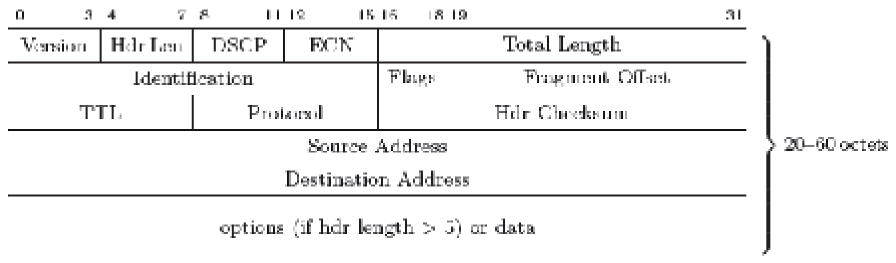


Figure 2.2.: IPv4 header format

- Multicast: IPv6 puts greater emphasis on multicast addressing, and depends on it for autoconfiguration and neighbor discovery.

- Autoconfiguration and Neighbor Discovery: IPv6 allows network devices to configure their own addresses and routes without manual configuration or additional network services (like DHCP).
- Flow Labels: A new field is included to mark sequences of packets (like TCP streams), which might aid routers with similar handling of a packet stream, for example to implement Quality of Service, without having to read every packet's Hop-by-Hop header or upper layer information.
- IPsec: Support for strong authentication, data integrity and encryption is mandatory for all nodes (in contrast to optional support with IPsec for IPv4). Albeit key management problems generally prevent a widespread use, it provides the basis for secure tunnels and authentication of other protocols (e. g. Mobile IP and OSPF).
- Mobile IPv6: To obtain roaming Internet connectivity for mobile devices, one associates hosts with both a fixed Home Address and a changing Care of Address in foreign nets. The use of IPsec enables a secure binding and tunneling between these addresses. Like with IPsec, this is specified for IPv4 as well, but its IPv6 version makes use of several IPv6 improvements (extension headers and neighbor discovery) and no longer requires special router support.
- Transition mechanisms: To enable coexistence and interconnectivity of IPv4 and IPv6 nets a number of transition mechanisms are specified and implemented, including dual-stack operation, tunnels and protocol translations. As all of these methods introduce new network paths between nodes, they enable new ways to manipulate routing paths and evade Access Control List (ACL) restrictions.

2.2 Huge Address Space in IPv6

The target of deploying IPv6 is its larger address space of 128 bits, as opposed to 32 bits in IPv4. These 128 bit addresses are split into a 64 bit subnet prefix and a 64 bit interface identifier, so every subnet has 2^{64} addresses for hosts to choose from. At first sight this seems to prevent remote reconnaissance attacks by network scanning because it is infeasible to scan significant parts of such a large address space [15]. But this is only true if the address allocation algorithm leads to a sparse and pseudo-random distribution across the available address space. The mandatory algorithm is to derive the interface identifier from the network interface's MAC address in EUI-64 format. The IEEE EUI-64 (EUI for Extended Unique Identifier) is a mapping of the 48 bit MAC address into the 64 bit address space for IPv6 interface identifiers. It concatenates the first 24 bits/3 octets of the MAC address, the constant 0xfffe, and the last 24 bits/3 octets of the MAC address. The network interface's 48 bit MAC address itself is a concatenation of a 24 bit manufacturer ID (the Organizationally Unique Identifier, OUI) and a 24 bit device specific ID. With currently about 15000 assigned OUIs (many of which are historic and not present in any current hardware), the actually used partition of the EUI-64 address space can be reduced to well below 240 addresses. So the EUI-64 addresses have considerably less entropy than randomly generated interface identifiers, but still enough to prevent exhaustive scanning. Even more entropy is gained with randomized interface identifiers, for example when using the privacy extension for stateless address auto configuration or cryptographically generated addresses. On the other hand many networks use a sequential numbering, often due to their DHCP server implementation or because it simplifies manual address management assignment. These addresses have little entropy and it is relatively easy to scan all hosts in such networks. Other address related security issues might arise from IPv4–IPv6 transition mechanisms. Because IPv4 addresses can also be represented as IPv6 addresses and the coexistence of two IP versions will lead to many tunneled connections, multiple opportunities for evasion attacks are created. For the foreseeable future all security devices, ranging from network traffic analysis to firewalls, will have to understand a variety of addressing schemes and encapsulation protocols only to determine the original protocol and source/destination addresses of packets (so they can apply the right restrictions and ACLs).

Unlike IPv4 with its 32 bit addressing space, IPv6 uses 128 bits to define its address space. This produces vast numbers of addresses that is much greater than those available in IPv4. For example if every subscriber was given a /48 Global Unicast Address prefix, each would contain 45 variable bits. This means that each subscriber would have 48 bits minus the 3 bit type field, which leaves each subscriber with 45 variable bits from which addresses can be derived. That is, the number of available prefixes is 2^{45} or about 35 trillion [7]. This increase from 32 bits to 128 bits also provides more levels of addressing hierarchy, a greater number of addressable nodes, and a simplified auto-configuration of addresses; all of which provide the network engineer increased flexibility. Specifically, the larger address space allows for many more devices and users on the Internet as well as extra flexibility in allocating addresses, thus eliminating the need for address conservation practices (e.g., NAT) and simplifying the auto-configuration process. IPv6 addresses identify interfaces within one of three hierarchical regions of the network. The scope of an address could be link-local, site-local, or global [8].

As with IPv4, IPv6 addresses have the most significant part of the address placed to the left, allowing for easy recognition of various address formats when logically dividing the 128 address bits into bit groups that can then be associated with special addressing features. The largest group of IPv6 addresses is global unicast addresses. The rest of the address groups are composed of unspecified and loopback addresses, multicast addresses, and link- and site-local addresses. Some of the ranges of addresses currently in use are illustrated in Table 2.1.

Table 2.1: IPv6 address space ranges

Use	**Binary Prefix**	**Slash**	**Description/Notes**
Unspecified	00...0	::/128	IPv6 address = 0:0:0:0:0:0:0:0 (or ::) Used before an address allocated by DHCP (equivalent of IPv4 0.0.0.0)
Loopback	00...1	::1/128	IPv6 address = 0:0:0:0:0:0:0:1 (or ::1) Local PC Loopback (equivalent of IPv4 127.0.0.1)
Multicast	1111 1111	FF::/8	There is now a link-local multicast format defined by RFC 4489.
Link-Local unicast	1111 1110 10	FE8::/10	Local LAN scope. Lower bits (EUI-64) created from MAC address or other Interface Identifier. There is now a link-local multicast format defined by RFC 4489.
Site-Local unicast	1111 1110 11	FEC::/10	Local Site scope. Lower bits assigned by user. This binary prefix has been marked Reserved by IANA to reflect the currently unsupported state of Site-Local addressing.
Global Unicast	All other values	2::/3	A note in RFC 3513 recommended that IANA should continue to allocate only from the binary prefix 001 (as in RFC 2373 version) but RFC 3587 obsoletes this recommendation.

As shown in Table 2.1, there are three types of IPv6 addresses [10]:

- Unicast: An identifier for a single interface. A packet sent to a unicast address is delivered to the interface identified by that address.
- Anycast: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to an anycast address is delivered to any one of the destination nodes identified by that address (the ”nearest” one, according to the routing protocols' measure of the distance).
- Multicast: An identifier for a set of interfaces (typically belonging to different nodes). A packet sent to a multicast address is delivered to all interfaces identified by that address.

In IPv6 broadcast addresses are not available; instead multicast is used. For example, the ”linkscope all-hosts multicast” address, ff02::1, corresponds to the IPv4 subnet-local broadcast address, 255.255.255.255. All IPv6 interfaces are required to have at least one link-local unicast address. A single interface may also have multiple IPv6 addresses of any type (Unicast, Multicast, Anycast) or scope [10]. Additionally multicast routing scalability is improved in IPv6 due to the addition of an added "scope" field to each multicast address. The addition of the "anycast address" is defined to allow for the delivery of a packet to any one of a group of nodes [11]. The IPv6 address is broken up into two 64 bit sections, the first is for network identification, while the second half identifies the host or Interface. Figure 2.3 shows the IPv6 Global Unicast Address breakdown, while Figure 2.4 shows the breakdown of the IPv6 Link Local Address.

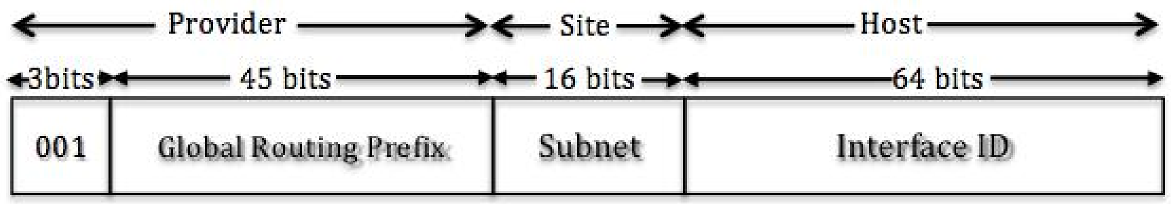


Figure 2.3: Example of global unicast IPv6 address

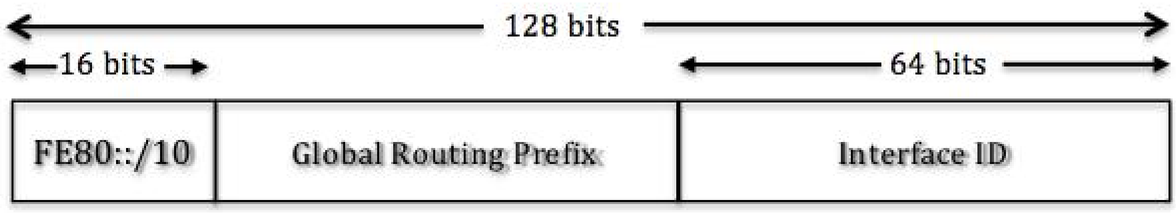


Figure 2.4: Example of link local IPv6 address

Reserved Address

The Internet Engineering Task Force IETF has reserved several predefined multicast addresses. These addresses are reserved for purposes such as news, music multicasts, and experimental purposes. In addition to the predefined multicast addresses reserved by the IETF, the following addresses are reserved:

- The IPv6 Unspecified Address: Each of the 128 bits in the IPv6 unspecified address has a value of zero. (The hexadecimal representation of this address is 0:0:0:0:0:0:0:0). Because this reserved address is actually the absence of an address, the unspecified address should neither be assigned to a node nor be used as a packet's destination. However, hosts can use this address as a source address to initialize themselves before they have configured their own unicast addresses.
- The Loop Back Address: The loop back address is another reserved address that should not be assigned to a node. (The hexadecimal representation of this address is 0:0:0:0:0:0:0:1). Packets that contain the loop back address as a destination code come back to the node from which the packet was sent. For example, you can use the loop back address to test network connections.
- Addresses Reserved for IPX Packets: IPv6 includes two options for integrating IPX networks with IPv6 networks. The first option uses an address space that is reserved for IPX packets. This reserved address space enables enterprises that use the IPX network layer protocol to map their IPX addresses to IPv6 addresses. These enterprises can then send and receive packets over the Internet. The format prefix that defines these mapped addresses is 0000010. The bits following this prefix contain the mapped 80-bit IPX address.

The second option for integrating IPX networks with IPv6 networks is to tunnel IPX packets in IPv6 packets. The IETF has defined an IPv6 header extension specifically for this purpose.

Addresses Reserved for Network Service Access Point (NSAP) Packets

To facilitate the mapping of NSAP addresses to IPv6 addresses, the IETF has reserved the set of addresses with a format prefix of 0000001. Unlike IPv6 addresses, which belong to organizations rather than a physical location, NSAP addresses describe the physical locations at which a network is attached to the Internet.

Unassigned Address

According to RFC 2373, the IPv6 addresses to which format prefixes have been assigned including unicast, anycast, multicast, and reserved addresses account for only 15 percent of the total number of addresses available. The IETF has not assigned format prefixes to the remaining 85 percent of addresses. These unassigned addresses are set aside for future use.

2.3 IPv6 Packet Format

IPv6 uses 128 bit addresses. In addition, some of the IPv4 header fields have been dropped or made optional. Examples of these dropped fields include, Flags, Identifier, and Checksum. The Fragment and Options and Padding fields have been replaced by IPv6 extension headers. This is to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header [2]. The specific fields of the IPv6 protocol header are shown in Table 2.2 [9].

Implementing a hierarchical addressing structure is just one way that IPv6 simplifies routing. Unlike IPv4, which uses variable-length packet headers, IPv6 uses fixed-length packet headers of 40 bytes. These fixed-length headers allow routers to parse packets more efficiently. IPv6 further simplifies routing requirements by using a smaller number of header fields. IPv4 packet headers contain 14 fields; IPv6 packet headers contain only eight fields. (See Table 2.2). Since IPv6 packets have fixed-length headers, the old IPv4 header length field is obviously no longer needed. Other eliminated IPv4 fields include the fragment offset, identification, flags, and header checksum fields.

The most significant deletion is the IPv4 header checksum field, which contains a computation based on the total number of bits in each particular IPv4 header. Each time a router receives an IPv4 packet, the router recomputes the number of bits the header contains. The router then checks its computation against the computation contained in the IPv4 header checksum field. If these two computations are identical, the data contained in the IPv4 header is most likely uncorrupted. In this case, the router forwards the packet. If the two computations are not identical, the router assumes the IPv4 packet is corrupted and discards it.

According to the IAB's Internet draft titled "The Case for IPv6," the IPv4 header checksum field is an unnecessary field that "has caused reduced performance in today's Internet." Because corrupted packets can be detected at both the data-link layer of the Open Systems Interconnection (OSI) model and the transport layer of the OSI model, routers do not need to check for bad packet headers. Any bad packets the data-link layer misses, the transport layer will catch.

Table 2.2: The specific fields of the IPv6 protocol header

IPv6 Header Format		
Name	**Length**	**Description/Notes**
version	4 bits	value = 6. Same location as IPv4 - everything after this changes.
traffic class	8 bits	None formally defined with IANA (late 2004). When used with Explicit Congestion Notification (ECN) (RFC 3168) may take defined values.
Flow Label	20 bits	-
payload length	16 bits	unsigned length in octets of payload (excludes header but includes extensions)
next header	8 bits	Identifier in following header - same values as IPv4 Protocol field Some common values: 0 (0x00) IPv6 Hop-by-Hop Option 1 (0x01) ICMP protocol 2 (0x02) IGMP protocol 4 (0x04) IP over IP 6 (0x06) TCP protocol 17 (0x11) UDP protocol 41 (0x29) IPv6 protocol 58 (0x3A) IPv6 ICMP protocol 59 (0x3B) IPv6 No Next Header (terminates a no upper layer frame)
hop limit	8 bits	Maximum number of hops. Formalizes the current practice when using the TTL in IPv4.
source IP	128 bits	-
destination IP	128 bits	-

IPv6 Link-Local Address Format

Link-Local addresses are automatically assigned by the end user equipment and require no external configuration (Format defined by RFC 4291). The address format uses a unique binary prefix (FE8::/10) and the remaining bits (118) are built from the local interface identifier. In the case of Ethernet (RFC 2464) the MAC (48 bits) is used to create the EUI-64 value. Each physical layer supported has a separate RFC. For example, FDDI, IEEE 802.15.4 etc. defining, among other things, how the link-local address is created. If an interface identifier has more than 118 bits the link-local address cannot be generated and the unit must be manually configured. Link-local addresses are not routable globally.

The 128 bits of a link-local address for an Ethernet interface breakdown is shown in Table 2.3.

Table 2.3: The 128 bits of a link-local address for an Ethernet interface breakdown

10 bits - Binary Prefix		
Name	**Size**	**Description/Notes**
Binary Prefix	10 bits	1111 1110 10 or FE8::/10 Link-Local Prefix
118 bits - constructed from interface MAC		
Name	**Size**	**Description/Notes**
-	54 bits	all zeros
MAC	24 bits	Top 24 bits of the 48 bit interface MAC Vendor ID. Additionally, when created using the MAC address or another EUI-48 derivation scheme, bit 7 (bits numbered from 1 in normal IETF convention) of this value is set to 1. Manually configured addresses do not set this bit. Thus both simplifying their generation and allowing external systems to identify how the address was generated. Note: This is the reverse of the meaning of this bit in the MAC address.
ID	16 bits	Fixed value of FFFE inserted
MAC	24 bits	Low 24 bits of the 48 bit interface MAC serial number.

IPv6 Multicast Address Format

The Multicast format (which also replaces broadcast in IPv4) is defined by RFC 4291. There is now a link-local multicast format defined by RFC 4489. The format of a global (non link-local) multicast address is defined in Table 2.4.

Table 2.4: The format of a global multicast address in IPv6

Name	**Bits**	**Size**	**Value**	**Description/Notes**
Binary Prefix	0 - 7	8	1111 1111	Fixed value e.g., the routing prefix, binary prefix
flags	8-11	4	0RPT	Where T may be: 0 = "well-known" or permanently (IANA) assigned group 1 = "transient" group which has no IANA assignment R flag: (RFC 3956) 0 = no Rendevous Point (RP) encoded 1 = RP encoded using method defined by RFC 3956. T = 1 must also be set. P flag: (RFC 3306) 0 = The multicast address is not based on the network prefix 1 = the multicast address is assigned based on the network prefix (format defined in RFC 3306). T = 1 must also be set.
scope	12-15	4	-	May take one of the following assigned values: 0 - reserved 1 - interface-local scope 2 - link-local scope 3 - reserved 4 - admin-local scope 5 - site-local scope 6 - (unassigned) 7 - (unassigned) 8 - organization-local scope 9 - (unassigned) A - (unassigned) B - (unassigned) C - (unassigned)

				D - (unassigned) E - global scope F - reserved
Group ID	16 - 127	112	-	Uniquely assigned by IANA if "well-known" bit = 0 set in T flag above.

Table 2.5 lists some of the more common multicast groups.

Table 2.5: Some of the more common multicast groups

Address	**Description/Notes**
FF01::1	Interface local - all nodes
FF02::1	Link local - all nodes
FF01::2	Interface local - all routers
FF02::2	Link local - all routers

IPv6 Link Local Multicast Address Format

RFC 4489 introduced the concept of a link-local (or link scoped) multicast format for situations where all configuration is stateless. Theoretically, routers (and other equipment) servicing a local (non-global) network could be now made self-configuring. Table 2.6 shows the link local multicast address format.

Table 2.6: The link local multicast address format

Name	**Bits**	**Size**	**Value**	**Description/Notes**
Binary Prefix	0 - 7	8	1111 1111	Fixed value e.g., the routing prefix, binary prefix
flags	8-11	4	0RPT	Flags have same meaning as defined for global multicast and must be set to 0011 meaning that T = 1 = "transient" group which has no IANA assignment and R = 1 = RP (Rendezvous Point) encoded using method defined by RFC 3956.
scope	12-15	4	-	Scope has the same meaning as defined for global multicast but can only take the values: 0 - reserved 1 - interface-local scope

				2 - link-local scope
Reserved	16 - 23	8	-	Reserved - must be 0
plen	24 - 31	8	-	Fixed value of 1111 1111 denotes a link-local (or link scoped) multicast address
IID (EUI-64)	32 - 95	64	-	Assigned using the normal link-local process depending on the media type.
Group ID	96 - 127	32	-	Since the T bit is set, this group ID is not defined by IANA. Though RFC 4489 does indicate that the guidelines for multicast address generation could be used (RFC 4489 and its position within the address also implies a mask of /96 applied to both the global and link-local format would yield a similar result.

IPv6 headers are daisy chained. The "Next Header" field is present in every header except the upper layer header to indicate which header comes next as shown in Figure 2.5.

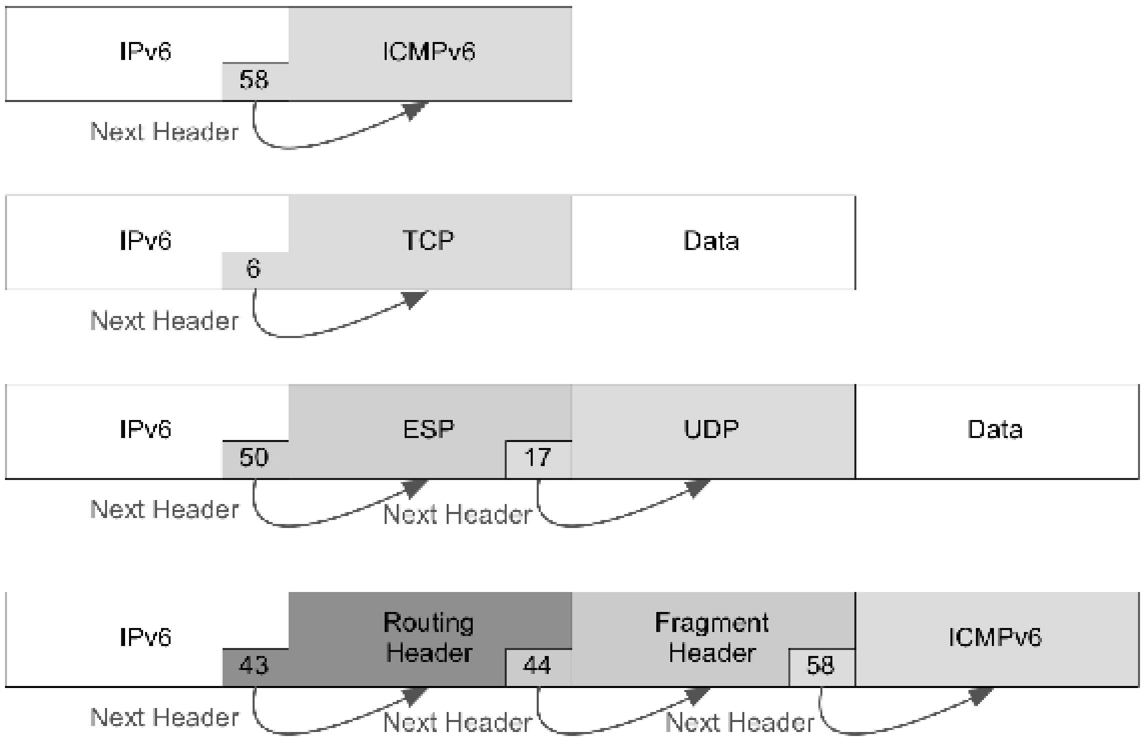


Figure 2.5: The Next Header field

Extension headers are always multiples of 8 octets. To allow skipping and processing of extension headers they all begin with 16 bit stub format as shown in Table 2.7.

Table 2.7: IPv6 Extension Header Stub format

IPv6 Extension Header - Stub format		
Name	**Length**	**Description/Notes**
Next Header	8 bits	Same values as IPv6 Next Header
Extension Hdr Len	8 bits	Unsigned integer. The total length of the extension header in multiples of 8 octets, excluding the first 8 octets e.g. a value of 0 = 8 octet header length, value = 2 = 24 octet header length etc. NB the length field in ICMPv6 does not use this convention.

Header Options

The Hop-By-Hop and Destination Headers carry a variable number of options within the header and use a classic TLD (or TLV in the standards paralance) format as shown in Table 2.8.

Table 2.8: Header options in IPv6

Name	**Length**	**Description/Notes**
Type	8 bits	The two high order (or low order depending on your numbering convention) bits indicate what action to take if the option is not recognized and may take one of the following values: 00 = skip option - keep processing 01 = discard packet 10 = discard packet and send ICMPv6 Parameter Problem (Code 2) message 11 = discard packet and, if not Multicast address, send ICMPv6 Parameter Problem (Code 2) message The third high order bit indicates whether the option can change before reaching its destination 0 = data will not change 1 = data may change. If the bit is set and an Authentication Header is present then an all zero option value must be assumed when computing any digest.
Length	8 bits	Length in octets of the option data - does not include the type or length value.
Data	variable	Depends on Type

In order to force so-called natural alignment of option fields two padding options are provided. An Option Type of 0 indicates a 1 octet pad (and does not have associated length or data fields), a standard Option with a Type of 1 allows for multiple octet padding. NB in this case a 2 octet pad will have an Option Length of 0.

2.4 IPv6 Configuration

2.4.1 Auto-Configuration

Autoconfiguration is one of the obvious advantages IPv6 has over IPv4. Autoconfiguration is a protocol that allows IPv6-enabled hosts to automatically configure and reconfigure their IPv6 addresses.

To automatically configure addresses, an IPv6-enabled host first configures an address for itself using a local network prefix and the host's own link address. (A host's link address is the physical address that identifies the host's Ethernet, Token Ring, or LocalTalk controller board). The host then uses a protocol called Neighbor Discovery to determine whether or not this link address is unique.

IPv6 Neighbor Discovery is a function of Internet Control Message Protocol version 6 (ICMPv6), a protocol that provides services, such as error reporting, for protocols that operate at the network layer of the OSI model. Using Neighbor Discovery, a host on an IPv6 network can discover whether or not its self-configured link address is unique: The host simply sends an ICMP Neighbor Solicitation multicast message to all of the hosts on the local link. If the originating host receives no reply, its link address is unique. If another host on this local link recognizes the new self-configured link address as its own link address, this host sends the originating host an ICMP message called a Neighbor Advertisement message. The Neighbor Advertisement message informs the originating host that the new self-configured address is not unique. The originating host then configures another address and sends a new multicast Neighbor Discovery message to the hosts on the link.

When the host finds a unique self-configured link address, the host then sends another Neighbor Discovery multicast message that includes the host's official link address as a source address. However, rather than sending a message to all of the hosts on its local link, the host sends the message to the router that connects that link to other network links.

When an IPv6-enabled router receives a Neighbor Discovery message from a host, the router sends that host a unicast message called a router advertisement. A router advertisement includes information such as a valid range of addresses for the subnet to which the router and host are attached. The router also tells the host whether it must use stateful or stateless autoconfiguration.

Stateful configuration requires a (DHCP) server to assign an IPv6 address to the host. If the router instructs the host to use stateful autoconfiguration, the host contacts a DHCP server with a request for a valid IPv6 address. DHCP servers assign valid IPv6 addresses dynamically--that is, each time a host makes a request, the DHCP server assigns the host IPv6 addresses from a pool of IPv6 addresses.

If the router instructs the host to use stateless autoconfiguration, the host uses information contained in the router's advertisement to generate its own address. This router-supplied information includes the numbers of the subnets associated with the host's link. The host then uses its own EUI-64 identifier to generate an interface ID for itself. Finally, the host appends the interface ID it generates to the subnet information supplied by the router. Autoconfiguration allows a company to change service providers without having to manually reconfigure addresses for every node on its company's network. Naturally, the bigger a company's network, the more time and money address autoconfiguration can save.

Address autoconfiguration also makes using roaming mobile hosts, such as a laptop or Internet-enabled cellular telephone, easier. Using autoconfiguration, a roaming mobile host can configure a valid IPv6 address for itself, regardless of the network to which it is temporarily attached. Using this current, temporary IPv6 address, the roaming mobile host can then ask a router on its home network (called a home agent) to forward packets to this newly configured address. In fact, technologies such as roaming mobile hosts may prove to be the push that starts the IPv6 ball rolling.

An IPv6 node interface can have multiple unicast addresses, a link-local address (which is the first address assigned to the interface), and one or many global or site-local addresses. Configuration of interfaces in IPv6 is controlled by the protocol itself [12]. The host auto-configuration feature allows hosts joining a link to configure link-local addresses for their interfaces as well as to check the uniqueness and validity of assigned addresses. Stateless auto-configuration is the process that allows an IPv6 host to be assigned addresses based on local router advertisements. In contrast, IPv4 uses the stateful address auto-configuration protocol, or (DHCP). In the stateful auto-configuration model, a host obtains the interface addresses as well as other required information such as the address of the default gateway a DNS server from a DHCP server. The DHCP server maintains a manually administered list of hosts and keeps track of which addresses have been assigned to which hosts. In addition, IPv6

offers stateless DHCPv6 which is a procedure during which addresses are configured according to the router advertisements along with additional information given to the host, such as default gateway and DNS servers, via a DHCP server.
IPv6 systems are typically multi-homed by default and have a link-local address configured by the host and may have a global unicast address which may be configured by one of three methods:

1. Stateful - Statically assigned = manual configuration
2. Stateful - DHCPv6 - Automatically assigned
3. Stateless - Automatically assigned (SLAAC)

2.4.2 IPv6 Stateless Autoconfiguration (SLAAC)

IPv6 systems may be configured to provide global unicast addresses using Stateless Address AutoConfiguation (SLAAC - defined by RFC 4862) using what is called generically the Neighborhood Discovery Protocol (NDP). Stateless autoconfiguration requires a router to be present but not a DHCP server. The process of creating a stateless IPv6 address is as follows:

1. Host sends a Router Solicitation message.
2. Host waits for a Router Advertisement message.
3. Host takes top bits as defined in the Prefix Information of the Router Advertisement message and combines it with the 64 bit EUI-64 address (in the case of Ethernet this is created from the MAC address) to create a Global Unicast address. The host also uses the source IP address - in the IP header - of the Router Advertisement message as its default gateway address.
4. RFC 4941 defines a method by which temporary (essentially pseudo-random from the interface derived EUI-64 address) addresses (EUI-64) may be created in order to create privacy (or anonymity).
5. Host then performs a Duplicate Address Detection to ensure the address is unique. If this check fails the host immediately aborts the autoconfiguration process and must be manually configured.

2.5 ICMPv6

IPv6 uses the Internet Control Message Protocol (ICMP6) as defined for IPv4 [13], with a number of changes [14]. ICMPv6 is vital to the proper operation of the IPv6 protocol. Unlike ICMP for IPv4, which is not required for IPv4 communications, ICMPv6 has features that are required elements which cannot be completely filtered [9]. For example the host auto configuration and Neighbor Discovery Protocol (NDP) x2.1.4 both require ICMPv6 messages to be able to complete address assignments and perform Duplicate Address Detection (DAD); both of which are vital to IPv6 operation. ICMPv6 operates on top of IPv6 as an extension header but actually works in conjunction with IPv6 for protocol operations. ICMPv6 is an integral part of IPv6, and must be fully implemented by every IPv6 node [14].

ICMPv6 is used by IPv6 nodes to report errors encountered in processing packets, and to perform other Internet-layer functions, such as diagnostics and testing (e.g., traceroute6). ICMPv6 messages contain a type and a code that relate the details of the message to the type of message, as well as a checksum and a payload of variable size. ICMPv6 error messages relay useful information back to the source of the packet about any error that may have occurred along the path. The general packet structure for ICMPv6 is shown in Figure 2.6. The fields of the ICMPv6 packet are as follows:

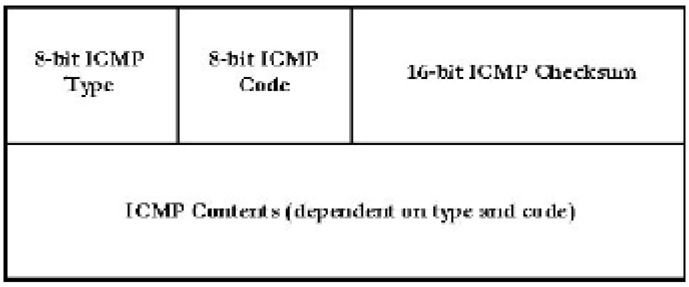


Figure 2.6: General ICMPv6 packet

- Type: The type field indicates the type of message.
- Code: The code field depends on the message type and is used to create an additional level of message granularity.
- Checksum: The checksum field is used to detect data corruption in the ICMPv6 message and parts of the IPv6 header.

ICMPv6 can be considered as the backbone of the IPv6 protocol, providing the following functions [9]:

- Neighbor Discovery Protocol (NDP), Neighbor Advertisements (NA), and Neighbor Solicitations (NS) provide the IPv6 equivalent of IPv4 Address Resolution Protocol (ARP) functionality.

- Router Advertisements (RA) and Router Solicitations (RS) help nodes determine information about their LAN, such as the network prefix, the default gateway, and other information that can help them communicate.
- Echo Request and Echo Reply support the Ping6 utility.
- PMTUD determines the proper MTU size for communications.
- Multicast Listener Discovery (MLD) provides IGMP-like functionality for communicating IP multicast joins and leaves. Multicast Router Discovery (MRD) discovers multicast routers.
- Node Information Query (NIQ) shares information about nodes between nodes.
- Secure Neighbor Discovery (SEND) helps secure communications between neighbors.

Chapter Three

Computer Network Attacks and Related work

Network threats exist in today's world in many forms. One of the more exciting and dynamic aspects of network security relates to the threat of attacks. Attack is any attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset [16]. A great deal of media attention and many vendor product offerings have addressed the topics of attacks and attack methodologies. There are thirteen general attack classes or categories [17] that attackers use to exploit IPv4 networks and hosts. These classes remain relevant with the IPv6 protocol stack. Each of the following general attack classes has either been made significantly easier, harder, or has no impact (or remains consistent) when considered in an IPv6 environment and will be covered in more detail in the following section.

1. Sniffing
2. Application Layer Attacks
3. Rogue Devices
4. Man In the Middle (MITM)
5. Flooding
6. Reconnaissance
7. Unauthorized access
8. Header Manipulation and fragmentation
9. Layer 3 spoofing
10. Address Resolution Protocol (ARP) and Dynamic Host Configuration Protocol (DHCP) attacks
11. Broadcast Amplification Attacks (smurf)
12. Viruses and Worms
13. Translation, Transition, and Tunneling

3.1 Consistent Threats

Some threats like application layer attacks remain mostly the same in IPv6 as they did in IPv4. This is because the implementation of IPv6 only affects layer 3 and has no direct impact on layer 7 of the OSI model. In fact, many of today's common attacks are application layer attacks. To this group of attacks belong buffer overflow attacks, Web application attacks (e.g., CGI attacks), different types of viruses and worms, etc [18]. Rogue device attacks such as an unauthorized laptop, rogue router, or rogue wireless access point are common in an IPv4 network and are not substantially changed in IPv6 [17]. Transition to IPv6 will not impact these types of attacks.

Spoofing Attacks

A spoofing attack is an attempt by someone or something to masquerade as someone else. This type of attack is usually considered an access attack. A common spoofing attack that was popular for many years on early Unix and other timesharing systems involved a programmer writing a fake logon program. It would prompt the user for a user ID and password. No matter what the user typed, the program would indicate an invalid logon attempt and then transfer control to the real logon program. The spoofing program would write the logon and password into a disk file, which was retrieved later. The most popular spoofing attacks today are IP spoofing and DNS spoofing. With IP spoofing, the goal is to make the data look as if it came from a trusted host when it didn't (thus spoofing the IP address of the sending host). With DNS spoofing, the DNS server is given information about a name server that it thinks is legitimate when it isn't. This can send users to a website other than the one they wanted to go to, reroute mail, or do any other type of redirection wherein data from a DNS server is used to determine a destination. Another name for this is DNS poisoning [20]. Figure 3.1 shows a spoofing attack occurring as part of the logon process on a computer network. The attacker in this situation impersonates the server to the client attempting to log in. No matter what the client attempts to do, the impersonating system will fail the login. When this process is finished, the impersonating system disconnects from the client. The client then logs in to the legitimate server. In the meantime, the attacker now has a valid user ID and password. The important point to remember is that a spoofing attack tricks something or someone into thinking something legitimate is occurring

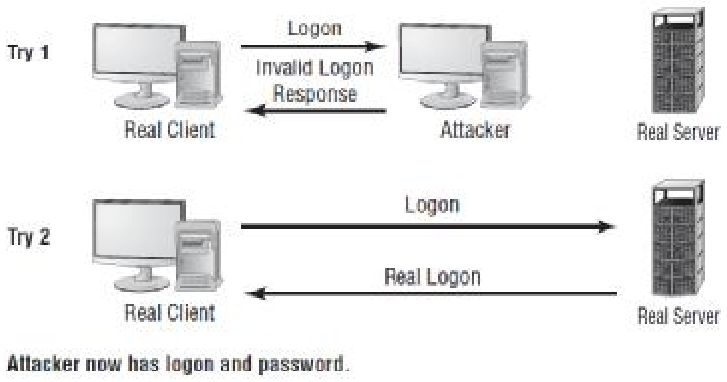


Figure 3.1: A spoofing attack during logon

Flooding Attacks

A flooding attack is a very frequent type of attack. The flooding is an attack that floods a network device, such as a router or a host, with large amounts (more that it can process) of network traffic. This attack can take the form of a local or distributed Denial of Service (DoS) attack and can cause network resources to become unavailable. Arrival of IPv6 did not change basic principles of a flooding attack [18]. However, with the introduction to new extension headers and ICMPv6 message types along with the dependence on multicast addresses, IPv6 may introduce more ways of developing flooding attacks for malicious purposes.

Sniffing

Sniffing is observing packets passing by on a network. Sniffing is popular way to steal data from a network, usually in form of passwords, ID names, etc. Passive attacks using this method have become frequent on the Internet. The person who is sniffing a network obtains data by actually sniffing the network for packets. The data is usually cached. Thus, hackers look for user IDs and the passwords of legitimate users and use the user's information to log on to the network. Once logged into the network, the hacker sniffs transmissions of packets. With this method the hacker can gather needed information about the network. Sniffing or eavesdropping on network traffic also remains unchanged. The tcpdump [19] tool has been implemented with IPv6 support, thus the sniffing of network traffic remains unchanged between IPv4 and IPv6.

Worms

A worm is a self-replicating virus that does not alter files but resides in active memory and duplicates itself. Worms use parts of an operating system that are automatic and usually invisible to the user. It is common for worms to be noticed only when their uncontrolled replication consumes system resources, slowing or halting other tasks.

Worms are another threat that remains mostly unchanged when considered with IPv6. Since worms are generally application layer threats, worms such as Melissa which spreads via email will be unaffected in an IPv6 environment [20]. However one of the popular mechanisms for worm propagation is random address-space probing. This allows a fast operating worm to scan an entire address space in a matter of hours. This was emphasized by Staniford et al. in their 2002 paper titled "How to own the Internet in your spare time" [21]. In this paper, they conclude that it is realistic for an attacker to gain control of a million or more hosts via the use of Worms on an IPv4 Internet. Thus giving the attacker the ability to conduct a mass Distributed Denial of Service (DDoS) attack, access sensitive information (e.g., credit card numbers, passwords), purposely sow confusion, as well as do deliberate damage to infrastructure. In an IPv4 32 bit address space, random address-space probing can be done quickly in order to find new hosts to infect. However when attempted with IPv6, with its 128 bit address space, scanning is a much more difficult and time consuming task. In fact, if we assume that the number of hosts on the Internet does not increase by a factor proportional to the address increase, then the work factor for finding a target in an IPv6 Internet would be approximately 2^{96} greater than that of IPv4 [20]. This work increase would seem to make the random scanning worms irrelevant due to the scan time expense. In the Bellovin et al. paper analyzing propagation strategies of worms in an IPv6 environment, they noted that address-space scanning worms such as Code Red will have a tough time effectively finding vulnerable victim hosts in an IPv6 environment [20]. The adoption of IPv6 removes one of the two ways address-space scanning is currently performed by making it infeasible to use a uniformly distributed random number generator to select new target addresses [20]. The other method, biasing the search space by scanning within the same subnet, preferentially spreads locally and is a much more feasible method [20]. However even scanning within an IPv6 subnetwork seems to be an unattainable goal for any worm. Even with only having to scan the local IPv6 subnet, a worm would be required to scan through 80 bits of local address space, which is a massive number and daunting obstacle. Thus, in an IPv6 environment, the worm threat itself does not change. However the method by which worms find hosts may have to change.

Man In The Middle (MITM)

Man in the Middle (MITM) attacks tend to be fairly sophisticated. This type of attack is also an access attack, but it can be used as the starting point for a modification attack. The method used in these attacks places a piece of software between a server and the user that neither the server administrators nor the user is aware of. The software intercepts data and then sends the information to the server as if nothing is wrong. The server responds back to the software, thinking it's communicating with the legitimate client. The attacking software continues sending information on to the server, and so forth. If communication between the server and user continues, what's the harm of the software? The answer lies in whatever else the software is doing. The man-in-the-middle software may be recording information for someone to view later, altering it, or in some other way compromising the security of the system and session. Figure 3.2 shows the Man in the Middle (MITM) attack. In recent years, the threat of Man in the Middle (MITM) attacks on wireless networks has increased. Because it's no longer necessary to connect to the wire, a malicious rogue can be outside the building intercepting packets, altering them, and sending them on. A common solution to this problem is to enforce a secure wireless authentication protocol such as WPA2 [22].

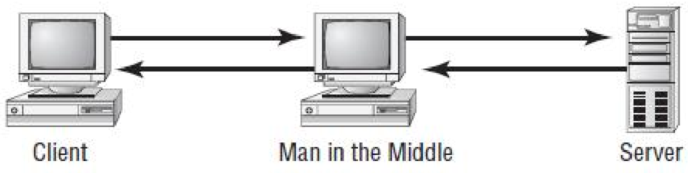


Figure 3.2: Man in the Middle (MITM) attack

The general theory of the Man in the Middle (MITM) threat does not change with IPv6. Because the IPv4 and the IPv6 headers have no security mechanisms themselves, each protocol relies on the IPsec protocol suite for security [17]. Since it is only mandatory for IPv6 implementations to support IPsec, but does not require it to be used, IPv6 falls prey to the same security risks posed by a MITM attack. Table 3.1 lists the IPv4 attacks that remain mostly unchanged when converted to IPv6. The table shows current IPv4 attack types or vulnerabilities next to their IPv6 counterpart.

Table 3.1: IPv4 attacks that remain mostly unchanged when converted to IPv6

Attack Type		
	IPv4 Attack	**Analog IPv6 Vulnerability**
Sniffing		
	TCPDUMP	TCPDUMP(IPv6 enabled)
Man in The Middle (MITM)		
	Various	Parasite6
	Various	Redirect6
Flooding		
		Flood_Advertise6
		Fake_router6

3.2 New Threats

This section outlines attacks that change significantly when considered in the IPv6 protocol address space. This section is based largely on Convery and Miller's 2004 paper on threat comparison and best practices [17]. IPv6 specific threat studies will also be covered in this section, however this will be as they relate to host and not NIDS detection.

Reconnaissance

Generally the first attack performed, reconnaissance, is an attempt by the adversary to learn about a network in an effort to find possible holes or weaknesses. Convery and Miller state that this includes both active network methods such as scanning as well as more passive data mining such as through search engines or public documents. [17]. The active host probing or port scanning is an attempt for an attacker to discover specific information about hosts and network devices on the victim's network. This includes how they interconnect and what traffic is being passed between them. Passive data mining can be considered environmental data to assist the attacker in theorizing different ways to attack the victim network. Typical IPv4 methods of collecting information are ping sweeps, port scans, and application and vulnerability scans. Reconnaissance in IPv6 differs from IPv4 in two relevant ways. The first is that ping sweep or port scan, when used to enumerate hosts on a subnet, are much more difficult to complete in an IPv6 network [17]. This is emphasized by Caicedo et al. in their 2009 paper where they note, the potentially huge size of IPv6 subnets makes reconnaissance attacks more difficult [23]. The large size of the IPv6 address space makes port scanning, whose procedures are identical for IPv4 and IPv6, more tedious and time consuming. With a default subnet on an IPv6 network being 64 bits, which means that to perform a scan on the whole subnet it is necessary to make 2^{64} probes, a next to impossible task [18]. Or as summarized by Convery and Miller, a network that ordinarily required sending 2^{56} probes now requires sending more than 18 quintillion probes to cover an entire subnet. A task that would take 28 centuries of constant 1-million-packets-per-second scanning to find the first host on the first subnet on a /64 IPv6 network containing 100 active hosts [17]. In an attempt to prove application vulnerability scans, or OS fingerprinting, have similar possibilities in IPv6 as they do in IPv4 Nerakis, in his 2006 thesis, used existing TCP/UDP packet probing methods along with IPv6 Extension Headers to attempt to determine the version and type of remote host operating systems. Nerakis discovered that existing TCP/UDP methods work, however it is more difficult to perform in an IPv6 environment. This is believed to be due in part to the larger address space and the (at the time) experimental nature of IPv6 with similar OSs possibly reusing IPv6 code [24]. IPv6's reliance on Multicast addresses will do just the opposite, making the adversaries life easier. The multicast address structure as defined in RFC 2375 [25] lets the attacker identify groups of key network components, such as the all router or all DHCP servers for a given network. This gives the attacker an almost hand delivered list of devices to scan for vulnerabilities, making his reconnaissance possible if not easy. This defined list of multicast addresses is clearly for legitimate protocol use, however it opens IPv6 up for reconnaissance as well as "simple flooding" attacks, or something more sophisticated that is designed to subvert the device [17].

Unauthorized Access

The Unauthorized Access attack is the type of attack in which an adversary tries to exploit the open transport policy found in IPv4. There is nothing in the protocol stack that limits the number of hosts that can connect to one another on an IP network. Attackers rely on this fact to establish connectivity to upper-layer protocols and applications on inter-networking devices and hosts [17]. To aid in preventing this attack the need for access controls is the same in IPv6 as it is in IPv4, though eventually the requirement and use of IPsec may enable easier host access control. Besides the mandatory support for IPsec, IPv6 technology differences that enable unauthorized access include Extension Headers, ICMP, Multicast and Anycast Inspection. In the case of Extension headers, which replaced the IPv4 IP options, all IPv6 endpoints are required to accept IPv6 packets with a routing header. This can be used by an attacker to circumvent security policies because of the possibility that the endpoint does not only accept the IPv6 packet but also processes it and forwards it, which possibly bypasses networks firewalls [17]. For ICMP, which is not required by IPv4, current best practices are generally in favor of complete filtering. However, ICMPv6 is an integral part of IPv6 operations and cannot be completely filtered without preventing communications. Since many of the utilities in IPv6, such as Neighbor Discovery, use ICMPv6, there are many opportunities to use it to aid in an Unauthorized Access Attack and thus subvert the networks' security policies or bypass firewalls.

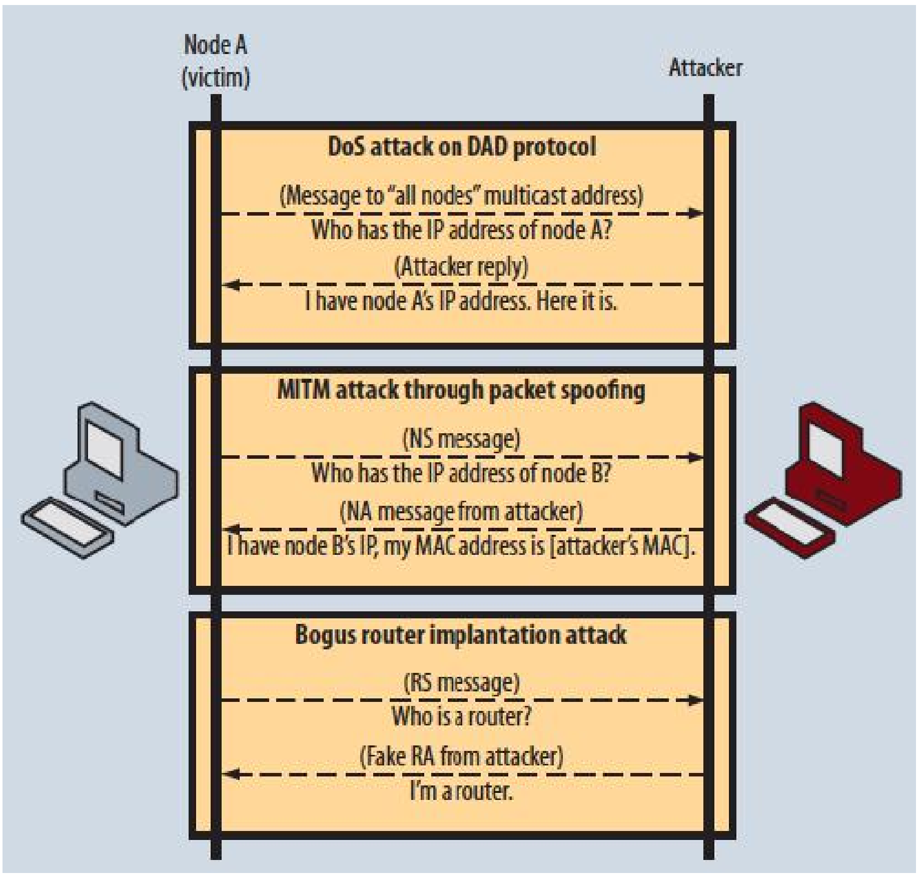
Header Manipulation and Fragmentation

The misuse of IPv6 routing and fragment headers can give an adversary tools to perform DoS attacks as well as avoid access controls. RFC 2460 [2] stipulates that all IPv6 nodes have to be able process routing headers. Because routing headers can be used to avoid access controls based on destination addresses, this presents a significant security issue. For instance, if an intruder sends a packet to a publicly accessible address with a routing header containing a "forbidden" address (address on the victim network) the publicly accessible host will forward the packet to a destination address stated in the routing header (e.g.,"forbidden" address) even though that destination address is filtered [18]. This enables the adversary the ability to perform a DoS attack by using this publicly accessible host to redirect spoofed packets. Historically fragmentation has been used in IPv4 to bypass access controls or slip attacks past routers, firewalls, and in particular NIDSs. The IPv6 specification [2] does not allow packet fragmentation by intermediary devices. In other words, only the source host can perform fragmentation. The source node could move port numbers from the first fragment to bypass security monitoring devices (which do not reassemble fragments) expecting to find transport layer protocol data in the first fragment [18]. Convery and Miller hypothesize that it is also possible to use the combination of multiple extension headers and fragmentation to create the same ability for the intruder to hide an attack. Since the "IPv6 minimum MTU is 1280 octets" a good security policy for IPv6 would have any packet with a MTU less that 1280 be dropped, but this is not always the case. Thus, an attacker can use large numbers of packets with small fragments (MTU <1280) to overload reconstruction buffers on the target system in an attempted DoS and possibly causing it to crash [17] [18].

Address Resolution Protocol (ARP) and Dynamic Host Configuration Protocol (DHCP) attacks

ARP and DHCP attacks attempt to subvert the host initialization process or a device that a host accesses for transit. These types of attacks try to get end hosts to communicate with an unauthorized or compromised device or attempt to configure these hosts with incorrect network information such as default gateway, DNS, or IP address [17]. In their 2004 paper, Convery and Miller note that IPv6 has no inherent security added to the IPv6 equivalents of DHCP (DHCPv6) or ARP (Neighbor Discovery). With the preference to Stateless Auto-configuration over DHCP, IPv6 is open for attacks in these areas. Although DHCP servers may be used on occasion, especially for setting up hosts with network configuration information, dedicated servers are not common in IPv6. Stateless Auto-configuration messages can be spoofed to allow an adversary to deny access to devices [17]. With the many types of devices that now connect to todays networks and the Internet, IPv6 replaces ARP with elements of ICMPv6 called Neighbor Discovery, which has the same inherent security as ARP in IPv4 [17]. This means that there are many options for these types of attacks in IPv6 based on vulnerabilities associated with ICMPv6. IPsec, the default security mechanism for IPv6, does not allow for automatic protection of the auto-configuration process. Thus, the Secure Neighbor Discovery Protocol (SeND) [27] was created to protect this process. SeND uses Cryptographically Generated Addresses (CGA) and asymmetric cryptography as a first line of defense against attacks on integrity and identity. In his 2007 thesis, Marcin Pohl evaluated SeND. He found that even though SeND claims to achieve mutual authentication of hosts and routers without the need for a Certification Authority (CA), SeND does not really offer mutual authentication without a CA and is susceptible to CPU exhaustion attacks [28]. However, without SeND both router and Neighbor solicitation and advertisement messages can be "spoofed" and will overwrite existing neighbor discovery cache information on a device, resulting in the same issues present in IPv4 ARP. What this means is that a spoofed router discovery could inject a bogus router address that hosts listen to and perhaps choose for their default gateway. Then the bogus router can record traffic and forward it to proper routers without detection; leaving the adversary the ability to perform MITM or DoS attacks at will [17]. [23] cover a DoS attack on the Duplicate Address Detection (DAD) protocol and its procedures in their 2009 paper. In this type of attack an attacker on the local link waits until a node sends an NS packet. The attacker then falsely responds with a neighbor advertisement (NA) packet, informing the new node that it is already using that address. Upon receiving the NA, the new node generates another address and repeats the DAD process; the attacker again falsely responds with an NA packet, thus repeating the whole process. They note, eventually the new node gives up without initializing its interface [23].

Another possible attack exploiting the stateless auto-configuration process is a MITM attack. This is possible when a node needs a MAC address of another node on the subnet and sends a NS message to the all-nodes multicast address. An attacker on the same link can see the NS message and reply to it with the corresponding NA message, thereby taking over all traffic between the two original nodes. Figure 3.3 graphically shows the process of three IPv6 auto-configuration (ARP/DHCP) type attacks.


Node A
(victim)
Attacker
DoS attack on DAD protocol
(Message to "all nodes" multicast address)
Who has the IP address of node A?
(Attacker reply)
I have node A's IP address. Here it is.
MITM attack through packet spoofing
(NS message)
Who has the IP address of node B?
(NA message from attacker)
I have node B's IP, my MAC address is [attacker's MAC].
Bogus router implantation attack
(RS message)
Who is a router?
(Fake RA from attacker)
I'm a router.
Figure 3.3: Attacks on IPv6 related to the auto-configuration process (From [23])

Broadcast Amplification Attacks (smurf)

Broadcast Amplification attacks, also known as “smurf” attacks, are DoS attacks that take advantage of the ability to send echo requests with a destination address of a subnet broadcast and a spoofed source address, using a victim’s IP address. This generates a response from all hosts on that subnet directly to the victim’s host creating a flood of echo response messages. In IPv6, since the concept of broadcast addresses is removed from the specification [1] and protocol, these types of attacks are mitigated. In regard specifically to “smurf” attacks, ICMPv6 messages should not be generated as response to a packet with a multicast destination address, a link layer multicast, or a link layer broadcast address as stated in RFC 2463 [29]. Even though this effectively kills “smurf” attacks in properly implemented IPv6 stacks, exceptions are made making these attacks possible on the local subnet. If a target has mis-implemented IPv6, it responds with an echo reply to the All-Nodes multicast address, this generates a mass of response traffic sent directly to the target.

Viruses and Worms

As discussed earlier, traditional worms and viruses remain unchanged with IPv6. The propagation methods of these types of attacks may encounter some difficulty with the large address space, which is seen in the Bellovin et al. 2006 paper [30]. Because of this difficulty most worms would be less effective in an IPv6 environment simply because of their inability to find hosts to infect.

Translation, Transition, and Tunneling

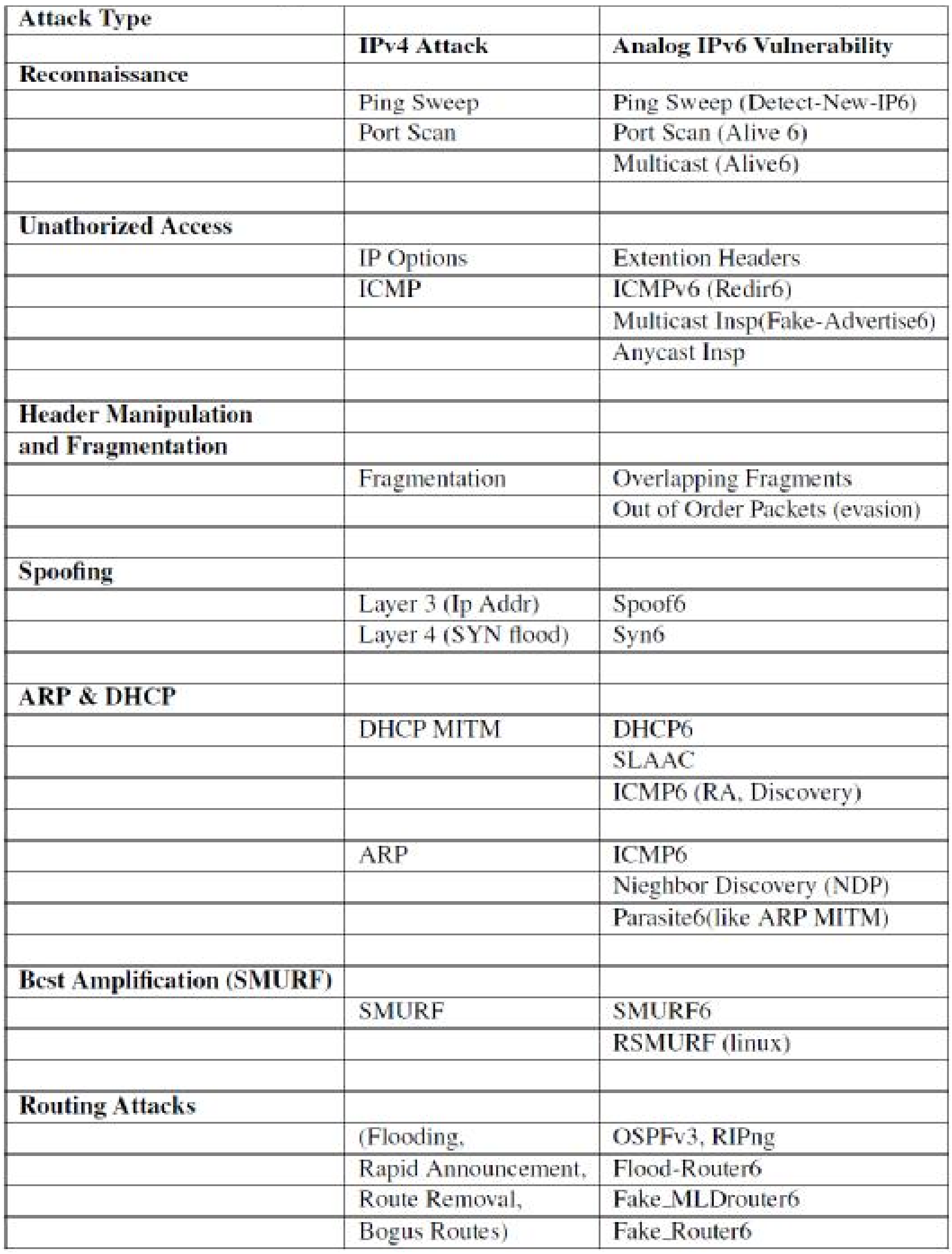
With the transition to IPv6 already underway, careful consideration must be given to the period between native IPv4 and native IPv6. As IPv4 networks are converted, there will be a considerable period of time where a transition mechanism will be required. During this ”in between” time, vulnerabilities specific to the transition mechanism must be taken into account and evaluated. There are several approaches to transitioning from IPv4 to IPv6:

- Dual stack
- Tunneling
- Translation

Each of these approaches has their own security considerations to be taken into account when deciding how to transition to IPv6. [17] state that the existence of so many transition technologies creates a situation in which network designers need to understand the security implications of the transition technologies and select the appropriate one for their network [17]. To this point when discussing IPv6 native access, we have discussed vulnerabilities and attacks that assume the end host is dual stacked, having both IPv4 and IPv6 infrastructure. Tunneling refers to the transmission of data intended for use only within a private, sometimes corporate, network through a public network in such a way that the routing nodes in the public network are unaware that the transmission is part of a private network. In this case, IPv6 is tunneled through the public IPv4 infrastructure. The IPv6 global Internet uses numerous tunnels over the existing IPv4 infrastructure. This is generally done through a tunnel broker such as Hurricane Electric [31] due to the complexity of setting up and managing these tunnels. Tunnels are difficult to configure and maintain and too complex for the isolated end user, so the concept of the tunnel broker was presented to help early IPv6 adopters to hook up to an existing IPv6 network. Additionally [74] noted that in many of the transition studies done, automatic tunneling mechanisms are susceptible to packet forgery and DoS attacks. These risks are the same in IPv4, however IPv6 increases the number of paths of exploitation for the adversaries [17]. Relay translation technologies, such as 6 to 4 [32], introduce automatic tunneling with third parties as well as additional DoS possibilities. Although, much like the case with tunneling IPv6, new avenues for exploitation are created and the risks do not change from those with IPv4 [17, 32].

Table 3.2 lists IPv4 attack classifications that have new or unique considerations when converted to IPV6. Just as in Table 3.2, the table shows current IPv4 attack types or vulnerabilities next to their IPv6 counterpart.

Table 3.2: Attacks with special consideration when converted to IPv6



Attack Type		
	IPv4 Attack	**Analog IPv6 Vulnerability**
Reconnaissance		
	Ping Sweep	Ping Sweep (Detect-New-IP6)
	Port Scan	Port Scan (Alive 6)
		Multicast (Alive6)
Unathorized Access		
	IP Options	Extention Headers
	ICMP	ICMPv6 (Redir6)
		Multicast Insp(Fake-Advertise6)
		Anycast Insp
Header Manipulation		
and Fragmentation		
	Fragmentation	Overlapping Fragments
		Out of Order Packets (evasion)
Spoofing		
	Layer 3 (Ip Addr)	Spoof6
	Layer 4 (SYN flood)	Syn6
ARP & DHCP		
	DHCP MITM	DHCP6
		SLAAC
		ICMP6 (RA, Discovery)
	ARP	ICMP6
		Nieghbor Discovery (NDP)
		Parasite6(like ARP MITM)
Bcst Amplification (SMURF)		
	SMURF	SMURF6
		RSMURF (linux)
Routing Attacks		
	(Flooding,	OSPFv3, RIPng
	Rapid Announcement,	Flood-Router6
	Route Removal,	Fake_MLDrouter6
	Bogus Routes)	Fake_Router6

3.3 Comparing IPv6 and IPv4 Security

A number of commenter's contend that IPv6 will provide a greater level of security than is available under IPv4. [75] states that because IPv6 was “designed with security in mind,” it is inherently more secure than IPv4, which does not have integrated security fields. Other commenters note that support for Internet Protocol Security (IPsec) is “mandatory” in IPv6, but only “optional” in IPv4, which should lead to more extensive use of IPsec in IPv6 networks and applications. [76] suggests that incorporating IPsec into the IPv6 protocol stack may reduce incompatibility between different vendors' implementations of IPsec. The virtually limitless address space available via IPv6 can also further network security. Many common IPv4-based network attack scenarios begin with "brute force" address and port scans of entire subnets, sites, or even the Internet as a whole. In typical IPv4 deployments, once an assigned address prefix is known, an attacker only has to scan between 2^8 subnet and 2^{16} site addresses (about 256 and 65,500 addresses, respectively) to find every host device on that network. The 64-bit space for individual interface IDs in the IPv6 address structure, on the other hand, is so vast that brute-force scans of the available address space are practically impossible. To the extent that deployment of IPv6 can enhance network security, the potential benefits to organizations and individuals can be significant. However, empirical estimates of the cost of cyber-security breaches vary widely because of differences in what is included in the cost estimates and disincentives for companies to publicly disclose the number of breaches or level of damage. Studies that focus on IT costs, such as the 2004 Computer Security Institute/FBI Computer Crime and Security Survey, have reported total losses from cyber-security breaches of approximately $142 million in 2004[35].

In addition to the above differences between IPv4 and IPv6 protocols, reference [35] states the following:

The evidence gathered in the Task Force's examination of IPv6 indicates that several potential security benefits can be realized from the eventual adoption and use of IPv6 by government, the private sector, and the Internet as a whole. At the same time, the greatest potential security benefits appear to be associated with the long-term evolution to new security paradigms, significantly different than those commonly employed in today's networks. As a result, the potential security benefits outlined above must be balanced against what might be considerable costs to complete the design and development of new security models and the potential increased risks to incrementally deploy and transition to them in existing operational networks. A number of factors may also limit the possible security benefits of IPv6 deployment in the near term. For example, although the expanded IPv6 address space may eliminate address and port scanning-based network attacks, network administrators may also lose the ability to perform brute-force address scans for the purposes of security auditing and testing. Many popular IPv4 security analysis tools are fundamentally based upon address scanning. Thus finding and identifying mis-configured or compromised hosts that are deliberately "hiding" on an IPv6 subnet may be as difficult as attacking them from the outside. This implies that in IPv6 networks both network administrators and would-be attackers must look elsewhere (e.g., DNS, server logs, neighbor discovery caches) to gather lists of active hosts. Furthermore, although IPsec support is mandatory in IPv6, IPsec use is not. In fact, many current IPv6 implementations do not include IPsec. On the other hand, though optional, IPsec is being widely deployed in IPv4. There appear to be no appreciable technical differences in the way that IPsec is implemented in either protocol, and several commenters state that there are no significant functional differences in the performance of IPsec in IPv6 and IPv4 networks. Any differences in performance are attributable to the presence of NATs in most IPv4 networks, which interfere with end to end communications using IPsec. Thus, to the extent that NATs persist in IPv6 networks, they may reduce the security benefits available via the new protocol.

Furthermore, experts generally agree that implementing any new protocol, such as IPv6, will be followed by an initial period of increased security vulnerability and that additional network staff will be necessary to address new threats posed by a dual network environment. For instance, IPv6 provides support for various configuration capabilities (e.g., neighbor discovery, address auto-configuration, router discovery, renumbering) and control (e.g., path MTU discovery). These capabilities are richer and better integrated than the auto-configuration capabilities typically found in today’s IPv4 networks and, as noted above, should result in reduced administrative costs associated with the operation of large-scale networks and potentially more streamlined implementations of some protocol functions.

Although there are clear operational advantages to these autoconfiguration and control capabilities, IPv6’s fundamental reliance on their operation also creates new threats and vulnerabilities associated with their potential misuse. This fact, coupled with a desire to support end-to-end (or host-based) security architectures in which trust among local network nodes is not assumed, requires that new levels of scrutiny be given to the security of the IPv6 Internet Control Message Protocol (ICMPv6) and its uses in neighbor discovery, and address auto-configuration. In addition, most IPv6 auto-configuration mechanisms make significant use of multicast, anycast, and scoped addressing capabilities. Care must be taken to ensure that network security systems limit the extent to which these new modes of addressing are not exploited as new attack vectors by compromised hosts.

Additionally, IPv6 inherently supports modes of addressing other than unicast (e.g., multicast, anycast, scoped unicast) that are not typically found in IPv4 operational deployments. Although these new addressing capabilities present significant opportunities for the development of new network services, security mechanisms and practices for these new modes of addressing are not as mature or well understood as those for global unicast. Additional efforts are needed to develop security solutions for IPv6 that can enable secure multicast and unicast communications while at the same time ensuring that these capabilities do not create new vulnerabilities in the networks in which they are deployed.

Although IPv4 may have presented similar security concerns when first implemented, it currently benefits in its comparison with IPv6 from 20 years of identifying and addressing security issues. As IPv6 becomes more prevalent, many security issues will likely arise as attackers give it more attention. On the other hand, the experience gained from running IPv4 networks may help bring security levels in IPv6 networks up to the level of current IPv4 networks at a faster pace. We can summarize a lot of main deference's between IPv4 and IPv6 in table 3.3.

Table 3.3: Main deference's between IPv4 and IPv6

IPv4	IPv6
32 bit address space	128 bit address space
Dynamic header length	Fix header length (40 byte)
14 fields heard	8 fields header
Hard routing header parsing	Easy routing header parsing
Need DHCP for Autoconfiguration	Autoconfiguration With Or not DHCPv6
Router or end node can fragment	Only end nodes can fragment
Source routing could be disable	Routing header required for Mobile IPv6
Layer 3 privacy	Layer 2or 3 privacy
Single address for each interface	Multi address for each interface

Chapter 4

NIDSs, NIPSs and Scapy

In this chapter we present Network Intrusion Detection Systems (NIDSs), Network Intrusion Prevention Systems (NIPSs), and Scapy security tool,.

4.1 Network Intrusion Detection Systems (NIDSs)

Network intrusion detection systems (NIDSs) are most efficient way of defending against network-based attacks aimed at computer systems [40, 41]. These systems are used in almost all large-scale IT infrastructures [42]. Basically, there are two main types of intrusion detection systems: Signature-Based Systems (SBSs) and Anomaly-Based Systems (ABSs). SBSs (e.g. Snort [43 ‹44]) rely on pattern recognition techniques where they maintain the database of signatures of previously known attacks and compare them with analyzed data. An alarm is raised when the signatures are matched. On the other hand ABSs (e.g PAYL [45]) build a statistical model describing the normal network traffic, and any abnormal behavior that deviates from the model is identified. In contrast to signature-based systems, anomaly-based systems have the advantage that they can detect zero-day attacks, since novel attacks can be detected as soon as they take place. Whereas ABSs (unlike SBSs) require a training phase to develop the database of general attacks and a careful setting of threshold level of detection makes it complex.. Payload-based systems are particularly suitable to detect advanced attacks [45], [46].

Criteria are important part of evaluation process, but we believe that most important part of IDS evaluation is setting testing objectives, it is necessary to collect enough background information before testing [72]. Lei Wei in the article [71] proposes framework for automatic Intrusion Detection System (IDS) evaluation which is shown in Figure 4.1. To get a report about the test result, there are around ten steps:

1. The process chooses a malware to attack the target system and sets configuration of the target system.
2. The process uses a sandbox, VMware, to build a virtual network for a test.
3. The evaluation system sets the attack configuration on the virtual attacker.
4. The virtual attacker machine attacks the target and, meanwhile, the traffic traces are documented and recorded by the system. Alarms raised by IDS are also collected.
5. The recorded data are saved into a data set in the shared disk.
6. The virtual attacker and target machines are restored to their initial configuration for next round of test.
7. IDS evaluator picks out all the traffic traces of a test case from the data set.
8. The IDS evaluator put those traffic traces to the tested IDS.

9. IDS result analyzer gets those traffic traces as well as the alarms generated by the IDS. He compares the traffic traces against the alarms to figure out the number of intrusions are detected by the IDS in the test scenario.
10. The evaluation framework generates the report.

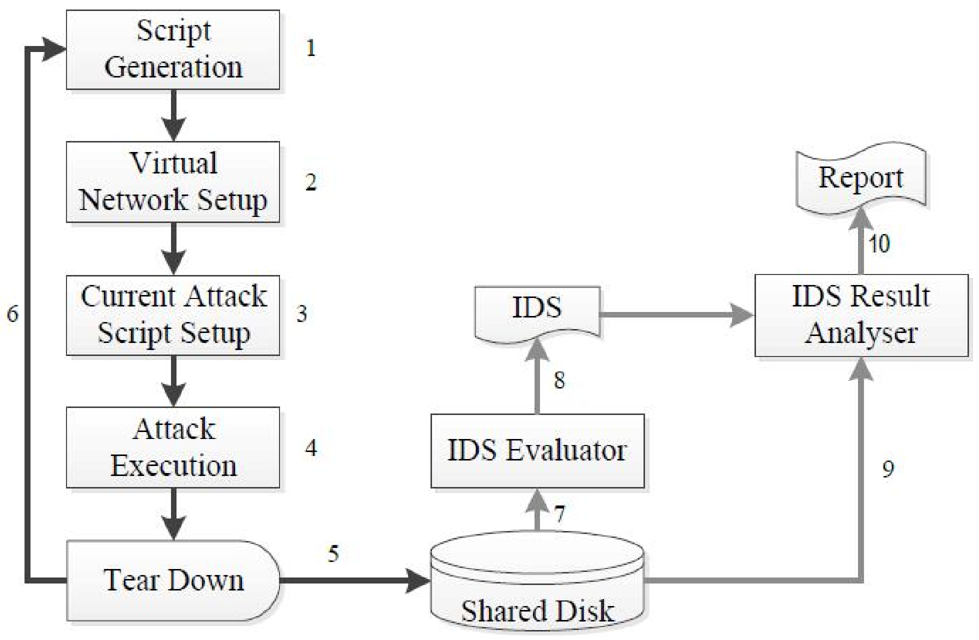


Figure 4.1: The working process of Automatic Intrusion Detection System [71]

The main problem in automatic evaluation is that there is missing IDS analysis within particular infrastructure. Even if we make automatic evaluation and comparison with other systems anyway network infrastructure should be analyzed. This process is extended with five more steps which can be seen in Figure 4.2. Such automatic evaluation can be done if it is necessary to compare some systems, but not for implementing system in organization network infrastructure. New security approaches should match the nature of the security problems, capable of adapting the strategy to new threats/attacks and of generating solutions dynamically. Each new service can include different compositions of many new technological equipment and software solutions, and these compositions entail different complex threats and risks. The composition of services does not necessarily imply that the upper services inherit the security attributes of its components. Each new composition adds and amplifies vulnerabilities and threats, and therefore each new service would require a specific security analysis [73]. Evaluation process should start from gathering user requirements for their network infrastructure and get information about environment characteristics and specific of NIDS, for example system type or architecture see Figure 4.2. From gathered information correct objectives and evaluation metrics can be set. Just after these steps can be started script generation and setting up of testing environment.

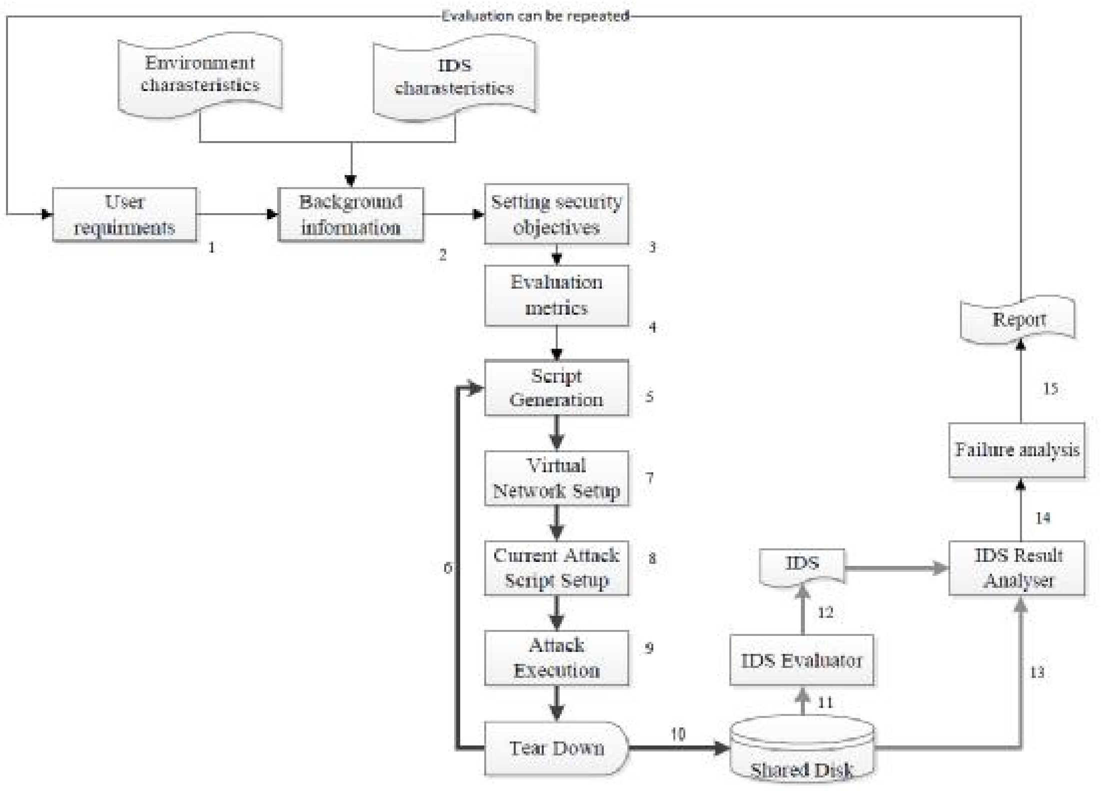


Figure 4.2: NIDS evaluation for particular network infrastructure [72]

4.1.1 Categories Of Intrusion Detection Systems

4.1.1.1 Signature Based Detection

Signature detection involves searching network traffic for a series of malicious bytes or packet sequences. The main advantage of this technique is that signatures are very easy to develop and understand if we know what network behavior we are trying to identify. For instance, we might use a signature that looks for particular strings within exploit particular buffer overflow vulnerability. The events generated by signature based IDS can communicate the cause of the alert. As pattern matching can be done more efficiently on modern systems so the amount of power needed to perform this matching is minimal for a rule set. For example if the system that is to be protected only communicate via DNS, ICMP and SMTP, all other signatures can be ignored. Limitations of these signature engines are that they only detect attacks whose signatures are previously stored in database; a signature must be created for every attack; and novel attacks cannot be detected. This technique can be easily deceived because they are only based on regular expressions and string matching. These mechanisms only look for strings within packets transmitting over wire. More over signatures work well against only the fixed behavioral pattern, they fail to deal with attacks created by human or a worm with self-modifying behavioral characteristics .Signature based detection does not work well when the user uses advanced technologies like nop generators, payload encoders and encrypted data channels. The efficiency of the signature based systems is greatly decreased, as it has to create a new signature for every variation. As the signatures keep on increasing, the system engine performance decreases. Due to this, many intrusion detection engines are deployed on systems with multi processors and multi Gigabit network cards. IDS developers develop the new signatures before the attacker does, so as to prevent the novel attacks on the system. The difference of speed of creation of the new signatures between the developers and attackers determine the efficiency of the system [47].

4.1.1.2 Anomaly Based Detection

The anomaly based detection is based on defining the network behavior. The network behavior is in accordance with the predefined behavior, then it is accepted or else it triggers the event in the anomaly detection. The accepted network behavior is prepared or learned by the specifications of the network administrators. The important phase in defining the network behavior is the IDS engine capability to cut through the various protocols at all levels. The Engine must be able to process the protocols and understand its goal. Though this protocol analysis is computationally expensive, the benefits it generates like increasing the rule set helps in less false positive alarms. The major drawback of anomaly detection is defining its rule set. The efficiency of the system depends on how well it is implemented and tested on all protocols. Rule defining process is also affected by various protocols used by various vendors. Apart from these, custom protocols also make rule defining a difficult job. For detection to occur correctly, the detailed knowledge about the accepted network behavior need to be developed by the administrators. But once the rules are defined and protocol is built then anomaly detection systems works well. If the malicious behavior of the user falls under the accepted behavior, then it goes unnoticed. An activity such as directory traversal on a targeted vulnerable server, which complies with network protocol, easily goes unnoticed as it does not trigger any out-of-protocol, payload or bandwidth limitation flags. The major advantage of anomaly based detection over signature-based engines is that a novel attack for which a signature does not exist can be detected if it falls out of the normal traffic patterns. This is observed when the systems detect new automated worms. If the new system is infected with a worm, it usually starts scanning for other vulnerable systems at an accelerated rate filling the network with malicious traffic, thus causing the event of a TCP connection or bandwidth abnormality rule [47].

4.1.2 Some Network Intrusion Detection Systems

CIDF (Common Intrusion Detection Framework):

Integrated with IETF and labeled as IDWG (Intrusion Detection Working Group) has achieved considerable progress in defining the framework, the group defined a general IDS architecture based on the consideration of four types of functional modules [47].

E-Modules (Event-Modules):

Combination of sensor elements that monitor the target system, thus acquiring information events to be analyzed by following modules [47].

D-Modules (Database-Modules):

The information from E-Modules are stored for further processing by following modules.

A-Modules (Analysis-Modules):

The Analysis of events and detecting probable aggressive behavior, so that some kind of alarm will be generated if necessary.

R-Modules (Response-Modules):

The main function of this type of block is the execution, if any intrusion occurs, of a response to perplexing the detected threat. Normally, Anomaly based Network intrusion detection systems (ANIDS) have following functional stages.

- Observation stage: A model is built on based on behavioral characteristics of the system. This can be done in many distinct ways, automatically or manually (depending on the type of ANIDS considered).
- Espial stage: If the model of the system is available, it is matched with the experiential traffic [47].

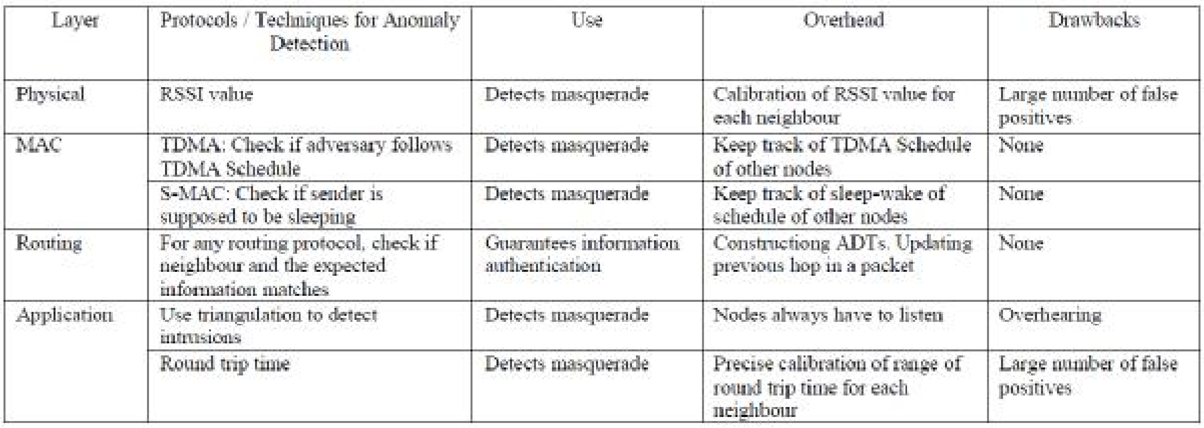
L-Modules (Location- Modules):

Our proposed modules in this thesis it depend of hacker location to detect the attack.

4.1.3 Anomaly Detection Techniques

Anomaly detection (as shown in table 4.1)is based on a host or network. Many distinct techniques are used based on type of processing related to behavioral model. They are: Statistical based, Operational or threshold metric model, Markov Process or Marker Model, Statistical Moments or mean and standard deviation model, Univariate Model, Multivariate Model, Time series Model, Cognition based, Finite State Machine Model, Description script Model, Adept System Model, Machine Learning based, Baysian Model, Genetic Algorithm model, Neural Network Model, Fuzzy Logic Model, Outlier Detection Model, Computer Immunology based, and User Intention based [47].

Table 4.1: Summary of the anomaly detection techniques



Layer	Protocols / Techniques for Anomaly Detection	Use	Overhead	Drawbacks
Physical	RSSI value	Detects masquerade	Calibration of RSSI value for each neighbour	Large number of false positives
MAC	TDMA: Check if adversary follows TDMA Schedule	Detects masquerade	Keep track of TDMA Schedule of other nodes	None
	S-MAC: Check if sender is supposed to be sleeping	Detects masquerade	Keep track of sleep-wake of schedule of other nodes	None
Routing	For any routing protocol, check if neighbour and the expected information matches	Guarantees information authentication	Constructiong ADTs. Updating previous hop in a packet	None
Application	Use triangulation to detect intrusions	Detects masquerade	Nodes always have to listen	Overhearing
	Round trip time	Detects masquerade	Precise calibration of range of round trip time for each neighbour	Large number of false positives

4.1.3.1 Statistical Models

Operational Model (or) Threshold Metric

The count of events that occur over a period of time determines the alarm to be raised if fewer then "m" or more than "n" events occur. This can be visualized in Windows lock, where a user after "n" unsuccessful login attempts. His allowed remaining attempts becomes zero. Executable files size downloaded is restricted in some organizations to about 4MB. The difficulty in this sub model is determining "m" and "n" [47].

Markov Process or Marker Model

The intrusion detection in this model is done by investigating the system at fixed intervals and keeping track of its state; a probability for each state at a given time interval is determined. The change of the state of the system occurs when an event happens and the behavior is detected as anomaly if the probability of occurrence of that state is low. The transitions between certain commands determine the anomaly detection where command sequences were important[47].

Statistical Moments or Mean and Standard Deviation Model

In statistical mean, standard deviation, or any other correlations are known as a moment. If the event that falls outside the set interval above or below the moment is said to be anomalous. The system is subjected to change by considering the aging data and making changes to the statistical rule data base. There are two major advantages over an operational model. First, prior knowledge is not required determining the normal activity in order to set limits; Second ،determining the confidence intervals depends on observed user data, as it varies from user to user. Threshold model lacks this flexibility. The major variation on the mean and standard deviation model is to give higher weights for the recent activities [47].

Multivariate Model

The major difference between the mean and standard deviation model is based on correlations among two or more metrics. If experimental data reveals better judicious power can be achieved from combinations of related measures rather than treating them individually.

Time Series Model

Interval timers together with an event counter or resource measure are major components in this model. Order and inter arrival times of the observations as well as their values are stored. If the probability of occurrence of a new observation is too low then it is considered as anomaly. The disadvantage of this model is that it is more computationally expensive.

4.1.3.2 Cognition Models

Finite State Machine

A finite state machine (FSM) or finite automation is a model of behavior captured in states, transitions and actions. A state contains information about the past, i.e. any changes in the input are noted and based on it the transition happens. An action is a description of an activity that is to be performed at a given moment. There are several action types: entry action, exit action, and transition action.

Description Scripts

Numerous proposals for scripting languages, which can describe signatures of attacks on computers and networks, are given by the intrusion detection community. All of these scripting languages are capable of identifying the sequences of specific events that are indicative of attacks.

Adept Systems

Human expertise in problem solving is used in adept systems. It solves uncertainties where generally one or more human experts are consulted. These systems are efficient in certain problem domain, and also considered as a class of artificial intelligence (AI) problems. Adept Systems are trained based on extensive knowledge of patterns associated with known attacks provided by human experts.

4.1.3.3 Cognition Based Detection Techniques

Cognition-Based (also called knowledge-based or expert systems) Detection Techniques work on the audit data classification technique, influenced by set of predefined rules, classes and attributes identified from training data, set of classification rules, parameters and procedures inferred.

Boosted Decision Tree

Boosted Tree (BT), that uses ADA Boost algorithm [48] to generate many Decision Trees classifiers trained by different sample sets drawn from the original training set, is implemented in many IDS successfully[49, 50, 51]. All hypotheses, produced from each of these classifiers, are combined to calculate total learning error, thereby arriving at a final composite hypothesis.

Support Vector Machine

Support vector machines (SVM)) [52], reliable on a range of classification tasks, are less prone to over-fitting problem, and are effective with unseen data. The basic learning process of the SVM includes two phases:

1. Mapping the training data from the original input space into a higher dimensional feature space, using kernels to transform a linearly non separable problem into a linearly separable one.

2. Finalizing a hyper plane within the feature space, with a maximum margin using Sequential Minimal Optimization (SMO) [53] or Osuna's method [54].

Artificial Neural Network

Artificial Neural Network (ANN) architectures (popular one being, Multilayer Perceptron (MLP), a layered feed-forward topology in which each unit performs a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output [55]), are able to identify not readily-observable patterns, however MLP is ineffective with new data. For general signal processing and pattern recognition problems, another branch of ANN that makes use of radial basis function, called The Modified Probabilistic Neural Network (related to General Regression Neural Network (GRNN) classifier [56] and generalization of Probabilistic Neural Network (PNN)), was introduced by Zaknich [57]. It assigns the clusters of input vectors rather than each individual training case to radial units.

4.1.3.4 Machine Learning Based Detection Techniques:

Machine learning techniques to detect outliers in datasets from a variety of fields were developed by [64] (use a One-Class Support Vector Machine (OCSVM) to detect anomalies in EEG data from epilepsy patients) and Barbara (proposed an algorithm to detect outliers in noisy datasets where no information is available regarding ground truth, based on a Transductive Confidence Machine (TCM) [58] (Unlike induction that uses all data points to induce a model, transduction, an alternative, uses small subset of them to estimate unknown attributes of test points. To perform online anomaly detection on time series data in [59], [59] presented an algorithm using support vector regression. [60] present an adaptive anomaly detection algorithm that is based on a Markov-modulated Poisson process model, and use Markov Chain Monte Carlo methods in a Bayesian approach to learn the model parameters [60].

4.1.3.5 Detection Models Based on Computer Immunology

Inspired by the ideas taken from the concept of immunology, the computational technique, Artificial Immune System (AIS) is developed in developing adaptive systems that are capable of solving problems from various domains. This has now become a tool for research which studied well and applied in solving problems in the computer security field and particularly to detect the viruses [61] in computers and also network intruders [62]. It is also applied to give a solution to the scheduling problem [63], to build support systems in taking decisions or to solve function optimization and combinatorial optimization problems [64]. Applying immunology to the basic computational model is the subject of research; the widely applied basic notions are antigen and antibody. The invaders which are foreign to the system and attack it in some way are antigens. A part of system, and which helps in detecting and eliminating the antigens are called antibodies. The detection of antigens is done by matching them. The number of antigens is much higher than the number of antibodies in the system, so a perfect matching can never occur. To keep a less number of antibodies, that reliably detect large number of antigens that were never seen before, AIS-based system is used.

4.1.3.6 Models Based on User Intention

Building the profile of normal behavior and attempting to identify certain pattern or activity deviations from normal profile. Anomaly detection is used to find unknown attacks by using the concept of profiling normal behaviors. But significant false alarm may be caused because it is difficult to obtain complete normal behaviors. Intrusion detection can be built upon multiple levels in a real computer network system. It will be a choosing the features that characterize the user or the system usage patterns in the best way, such that distinguishing abnormal activities from normal activities is done clearly. Data sources like Unix shell commands, audit events, keystroke, system calls and network packages can be used. The first crucial step in building a profiling method for intrusion detection is selecting a data source. During the early study on anomaly detection, the main focus was on profiling system or user behaviors from monitored system log or accounting log data [47].

4.2 Network Intrusion Prevention Systems (NIPSs)

NIPSs are an advance combination of NIDSs, personal firewalls and anti-viruses etc. The purpose of a Network Intrusion Prevention System (NIPS) is not only to detect an attack that is trying to interrupt, but also to stop it by responding automatically such as logging off the user, shutting down the system, stopping the process and disabling the connection etc. Similar to NIDSs, NIPSs can be divided into two types, i.e. Host-Based Intrusion Prevention Systems and Network-Based Intrusion Prevention Systems [70].

NIPSs are network security appliances that monitor network and/or system activities for malicious activity. The main functions of intrusion prevention systems are to identify malicious activity, log information about said activity, attempt to block/stop activity, and report activity [65].

Network intrusion prevention systems are considered extensions of intrusion detection systems because they both monitor network traffic and/or system activities for malicious activity. The main differences are, unlike intrusion detection systems, intrusion prevention systems are placed in-line and are able to actively prevent/block intrusions that are detected [66][67]. More specifically, NIPSs can take such actions as sending an alarm, dropping the malicious packets, resetting the connection and/or blocking the traffic from the offending IP address [68]. An IPS can also correct Cyclic Redundancy Check (CRC) errors, un-fragment packet streams, prevent TCP sequencing issues, and clean up unwanted transport and network layer options [66] [69].

Intrusion detection systems are network or host based solutions. Network-based IDS systems (NIDS) are often standalone hardware appliances that include network intrusion detection capabilities. It will usually consist of hardware sensors located at various points along the network or software that is installed to system computers connected to your network, which analyzes data packets entering and leaving the network. Host-based IDS systems (HIDS) do not offer true real-time detection, but if configured correctly are close to true real-time.

Host-based IDS systems consist of software agents installed on individual computers within the system. HIDS analyze the traffic to and from the specific computer on which the intrusion detection software is installed on. HIDS systems often provide features you can't get with a network-based IDS. For example, HIDS are able to monitor activities that only an administrator should be able to implement. It is also able to monitor changes to key system files and any attempt to overwrite these files. Attempts to install Trojans or backdoors can also be monitored by a HIDS and stopped. These specific intrusion events are not always seen by a NIDS.

While it depends on the size of your network and the number of individual computers which require intrusion detection system, NIDS are usually a cheaper solution to implement and it requires less administration and training – but it is not as versatile as a HID. Both systems will require Internet access (bandwidth) to ensure they system is kept up-to-date with the latest virus and worm signatures [77].

Before evaluating IDPS products, organizations should define the requirements that the products should meet . Evaluators need to understand the characteristics of the organization’s system and network environments ، so that a compatible IDPS can be selected that can monitor the events of interest on the systems and/or networks. Evaluators should articulate the goals and objectives they wish to attain by using an IDPS, such as stopping common attacks, identifying mis-configured wireless network devices, and detecting misuse of the organization’s system and network resources. Evaluators should also review their existing security policies, which serve as a specification for many of the features that the IDPS products need to provide. In addition, evaluators should understand whether or not the organization is subject to oversight or review by another organization. If so, they should determine if that oversight authority requires IDPSs or other specific system security resources. Resource constraints should also be taken into consideration by evaluators. Evaluators also need to define specialized sets of requirements for the following :

- Security capabilities, including information gathering, logging, detection, and prevention.
- Performance, including maximum capacity and performance features.
- Management, including design and implementation (e.g., reliability, interoperability, scalability ، product security), operation and maintenance (including software updates), and training ، documentation, and technical support .
- Life cycle costs, both initial and maintenance costs.

4.2.1 Key Functions of IDPS Technologies

There are many types of IDPS technologies, which are differentiated primarily by the types of events that they can recognize and the methodologies that they use to identify incidents. In addition to monitoring and analyzing events to identify undesirable activity, all types of IDPS technologies typically perform the following functions :

- **Recording information related to observed events**: Information is usually recorded locally, and might also be sent to separate systems such as centralized logging servers, security information and event management (SIEM) solutions, and enterprise management systems.
- **Notifying security administrators of important observed events**: This notification, known as an alert, occurs through any of several methods, including the following: e-mails, pages, messages on the IDPS user interface, Simple Network Management Protocol (SNMP) traps, syslog messages, and user-defined programs and scripts. A notification message typically includes only basic information regarding an event; administrators need to access the IDPS for additional information .
- **Producing reports:** Reports summarize the monitored events or provide details on particular events of interest .Some IDPSs are also able to change their security profile when a new threat is detected. For example, an IDPS might be able to collect more detailed information for a particular session after malicious activity is detected within that session. An IDPS might also alter the settings for when certain alerts are triggered or what priority should be assigned to subsequent alerts after a particular

threat is detected .IPS technologies are differentiated from IDS technologies by one characteristic: IPS technologies can respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which can be divided into the following groups :

- **The IPS stops the attack itself**: Examples of how this could be done are as follows :
  - Terminate the network connection or user session that is being used for the attack .
  - Block access to the target (or possibly other likely targets) from the offending user account, IP address, or other attacker attribute .
  - Block all access to the targeted host, service, application, or other resource .
- **The IPS changes the security environment:** The IPS could change the configuration of other security controls to disrupt an attack. Common examples are reconfiguring a network device (e. g., firewall, router, switch) to block access from the attacker or to the target, and altering a host-based firewall on a target to block incoming attacks. Some IPSs can even cause patches to be applied to a host if the IPS detects that the host has vulnerabilities .
- **The IPS changes the attack's content**: Some IPS technologies can remove or replace malicious portions of an attack to make it benign. A simple example is an IPS removing an infected file attachment from an e-mail and then permitting the cleaned email to reach its recipient. A more complex example is an IPS that acts as a proxy and normalizes incoming requests, which means that the proxy repackages the payloads of the requests, discarding header information. This might cause certain attacks to be discarded as part of the normalization process .

Another common attribute of IDPS technologies is that they cannot provide completely accurate detection. When an IDPS incorrectly identifies benign activity as being malicious, a false positive has occurred. When an IDPS fails to identify malicious activity, a false negative has occurred. It is not possible to eliminate all false positives and negatives; in most cases, reducing the occurrences of one increases the occurrences of the other. Many organizations choose to decrease false negatives at the cost of increasing false positives, which means that more malicious events are detected but more analysis resources are needed to differentiate false positives from true malicious events. Altering the configuration of an IDPS to improve its detection accuracy is known as tuning . Most IDPS technologies also offer features that compensate for the use of common evasion techniques .Evasion is modifying the format or timing of malicious activity so that its appearance changes but its effect is the same. Attackers use evasion techniques to try to prevent IDPS technologies from detecting their attacks. For example, an attacker could encode text characters in a particular way, knowing that the target understands the encoding and hoping that any monitoring IDPSs do not. Most IDPS technologies can overcome common evasion techniques by duplicating special processing performed by the targets. If the IDPS can "see" the activity in the same way that the target would, then evasion techniques will generally be unsuccessful at hiding attacks [78].

4.2.2 IDPS Product Selection

There are many points to select an IDS or an IPS. The following are the main six criteria [79]:

1. **Provide comprehensive protection for all types of network attacks**

   As with all security systems, the comprehensiveness of protection provided by an intrusion detection and prevention system is a critical element to its ability to accurately identify threats and effectively secure the network. Yet many products fall short. The inherent complexity of network traffic, which includes the vast number of protocols at both the network (IP, TCP, UDP, ICMP, etc.) and application (HTTP, FTP, SMTP, DNS, POP3, IMAP, etc.) layers, provides attackers ample vulnerabilities to exploit. Combine the inherent complexity with the fact that attacks come in different shapes and forms, and attackers have a virtual buffet to choose from when they are attacking your network. If a system does not support one of these protocols or types of attacks, it will ignore and miss the attack, leaving the enterprise network and its valuable corporate data unprotected. To thwart the attacker's efforts, a device needs to be able to address and protect against all types of traffic and potential attacks .

2. **Ensure a high degree of accuracy**

   Accuracy is key to an effective and efficient intrusion detection and prevention system. To create a high level of accuracy, a system must be able to track all network communications, interpret the intent of each individual communication, and then make a security decision, based on accurate evidence of an attempted attack perpetration. If the product isn't accurate, attacks may go undetected or benign traffic may be alerted on as an attack. Attacks that go by undetected is the worst scenario because it means the network is vulnerable and the administrator has to try and figure out what happened from scratch. However, if the number of false alarms that a device sends out overwhelms the administrator, or even outnumber the real ones, it is also very detrimental, wasting valuable time and resources chasing down false alarms and eroding the trust that an administrator has in the system. The system needs to be reliable and accurately detect all the attacks in the network.

3. **Process traffic in a highly efficient manner**

   Efficient data processing, to ensure that all functions of the system are optimally performed, is another important element of effective intrusion detection and prevention .The system must process traffic quickly, make security decisions instantly, and present that information to the network manager in a timely fashion, ensuring the administrator has a real-time view of the system at all times. A slow system that cannot keep up with the rapid flow of network traffic can mean missed attacks and an increasingly vulnerable corporate network. An administrator should also be able to choose the level of performance for the device to meet network requirements and ensure that both fast Ethernet and Gigabit network segments can be protected. The device needs to perform in an optimal manner, so that the administrator knows exactly what is going on in the network at any given time.

4. **Protect against an attack without latency**

   Whether an intrusion detection and prevention system can stop the attack from ever reaching its victim is the cornerstone to its prevention capabilities. How effective is an intrusion detection system that has to rely on another system to try to prevent an attack ؟The answer is obvious, but many intrusion detection products do just that, sending a request to a firewall or even the victims themselves to try to end the attack. All of these mechanisms come after the attack has already reached the victim, so even when successful, they require the network administrator to investigate exactly how much the attack was able to do before it was stopped. Any device that introduces latency to the prevention response, is not able to offer true prevention. A truly effective solution can actively prevent attacks during the detection process and drop the malicious traffic. This ensures it never reaches its intended victim, keeping the enterprise network and sensitive, mission-critical data safe and secure .

5. **Deliver ease of use**

   The ease of use of intrusion detection and prevention system translates directly into greater control and a higher degree of security. If the system enables administrators to quickly view pertinent, critical information and make adjustments, network managers can ensure the network is efficiently protected from the latest threats and the most up-to-date security policy is in force. If a device is hard to control and understand, administrators are going to waste time trying to find the information they need to do their job. It is important that the solution enables both a quick summary of the most important types of events, as well as a way to quickly drill into the raw data and manipulate it to analyze individual incidents. Providing a granular level of control in an intuitive manner to security managers, not only ensures that the system meets the company's specific security requirements, but also that valuable IT time and resources are not misspent or wasted.

6. **Simple installation and maintenance**

   In today's highly distributed, global enterprises, the intrusion detection and prevention product must be both easy and cost-effective to install and maintain. Companies simply cannot spare the time and resources required to update each individual device within the corporate network every time a change is made to the enterprise's security policy or a new attack is detected from which the enterprise must be protected. Quick system installation, security policy definition, and easy, global updates from a single, centralized location ensure that enterprise IT teams scattered around the globe can have a comprehensive, real-time view of the system and the network.

4.3 Scapy

Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery (it can replace hping, 85% of nmap, arpspoof, arp-sk, arping, tcpdump, tethereal, etc.). It also performs very well at a lot of other specific tasks that most other tools can't handle, like sending invalid frames, injecting someone's own 802.11 frames, combining techniques (VLAN hopping+ARP cache poisoning, VOIP decoding on WEP encrypted channel, ...), etc. [36].

Scapy uses the python interpreter as a command board. This means that we can use directly python language (assign variables, use loops, define functions, etc.) [37].

In the following subsections we present some of the Scapy topics.

4.3.1 Installing Scapy

The main prerequisite to run Scapy is Python 2.5, one of the features of Scapy is that once you start creating packets within the application you can then use it in python scripts/code. The following steps are what you need to get started with Scapy:

1. Install Python 2.5+
2. Download and install Scapy
3. Install additional software for special features (Optional):
4. Run Scapy with root privileges

In the following we describe these steps in some detail.

Install Python 2.5+

The official documentation for Scapy states Python 2.5 to run Scapy 2.x,

In Ubuntu Linux Operating system you can simply type:

```
$ sudo apt-get install python
```

Download and install Scapy

Once you have python installed you need to get scapy. There are a few ways you can do this. The following are some of the popular ones:

```
$ sudo apt-get install python-scapy
$ cd /tmp
$ wget scapy.net
$unzip scapy-latest.zip
$ cd scapy-2.*
$ sudo python setup.py install
```

Install additional software for special features (Optional):

Now by this point you should have Scapy installed, you can test this by simply typing this in a terminal:

```
$ sudo scapy
WARNING: No route found for IPv6 destination :: (no default route?)
```

```
        Welcome to Scapy  (2.2.0)
```

To exit out of Scapy, just type:

```
        >>> exit()
```

Version 2.2.0 seems to be latest stable build and it is the version we will use in this thesis. Now when you just ran Scapy you might have noticed some errors about components missing, Scapy can do a lot of extra things such as providing 3D maps, graphs, charts etc but you need some extra packages. Here's the command to get all those extra goodies:

```
        $ sudo apt-get install tcpdump graphviz imagemagick python-gnuplot pythoncrypto
        python-pyx (this is all one line)
```

Run Scapy with root privileges

The following is the command to do this:

```
        $ sudo scapy
```

The, you may see this:

```
        WARNING: No route found for IPv6 destination :: (no default route?)
        Welcome to Scapy (2.2.0)
        >>>
```

So that covers the install of Scapy. And you are all set to get started.

4.3.2 Creating a Packet

You will slowly build your knowledge and start experimenting yourself with Scapy. To start with we are going to create a single ICMP packet with the well-known message "HelloWorld" contained as the payload [38].

```
        Welcome to Scapy (2.2.0)
        >>> send(IP(dst="10.1.99.2")/ICMP()/"HelloWorld")
        Sent 1 packets.
        >>>
```

There are many Scapy commands that we used to program our SRP. To see this command you can type:

```
        >>>lsc()
```

4.3.3 Probe Once, Interpret Many

Network discovery is blackbox testing. When probing a network, you will send many stimuli, and some of them will be answered. If you choose the right stimuli you can obtain the information you need from the responses or from the lack of responses. Unlike many tools, Scapy will give you all the information, i.e. all the stimuli you sent and all the responses you got. You will have to look into them to get the information you are looking for. When the dataset is small، you can just dig for it. In other cases, you will have to choose a point of view on this data. Most tools choose the point of view for you and loose all the data not related to the given point of view. Because Scapy gives you the whole raw data, you can use them many times and have your point of view evolve during your analysis. For example, you can probe for a TCP port scan, visualize the data like the result of a port scan, then decide you would like to also visualize the TTL of response packet. You do not need to do a new probe each time you want to visualize other data [39].

4.3.4 Scapy Decodes, It Does Not Interpret

A common problem in network probing tools is that they try to interpret the answers they got instead of only decoding and giving facts. Saying something like I received a TCP Reset on port 80 is not subject to interpretation errors. Saying the port 80 is closed is an interpretation that can be right most of the time but wrong in some specific contexts. For instance, some scanners tend to report a filtered TCP port when they receive an ICMP destination unreachable packet. This may be right, but in some cases it means the packet was not filtered by the firewall but there was no host to forward the packet to interoperating results can help people that don't know what a port scan is. But it can also make more harm than good, as it bias the results. In fact, what happen to people that know exactly what they are doing and that know very well their tools is that they try to reverse the tool's interpretation to get the facts that triggered the interpretation, in order to do the interpretation themselves [39].

Chapter 5
Hacker Location Detection System (HLDS) Methodology

5.1 Introduction

In this chapter we present our proposed Hacker Location Detection System (HLDS) which is capable of both detection of hackers and identification of their location based on their IP address, MAC address, and geographic location. HLDS is a Software Router Plug-in (SRP) that will be used in IPv6 routers to provide Internet security. Since hackers are able to change IP and MAC addresses of their computers and NIC respectively, our proposed system does not depend only on IP or MAC addresses but also on location of hackers. Hence, even if hackers change these IP and/or MAC addresses our proposed system is capable of tracing these hackers.

The idea of HLDS is based on the knowledge of the geographical location hackers from previous experience. This location is used by hackers to hack network packets passing through them.

When a user sends a packet to another user and the packet passes through the hacker's router then HLDS will alarm the communicating users if HLDS has the location of the hacker stored in Hacker Location Database (HLDB) server used by HLDS. HLDB is a database that stores known hacker locations based on previous experience. It is updated regularly to reflect the new situation of hackers and their locations. The accuracy of HLDS is based on the accuracy of information stored about hackers in HLDB and its constant update. Hackers could use different locations at different times. This leads to false positive alarms as well as false negative alarms. A false positive alarm is generated by HLDS when the packet passes through router at a location that is identified as a hacker's location while in reality the location is not used by a hacker. This could be due to existence of not-hacker routers that use the same location that is stored in HLDB as a hacker's location. A false negative result of HLDS is when HLDS does not generate an alarm while in reality the packet passed through a router in a location that is not identified by HLDS as a hacker's location. This could be due to the fact that HLDB is not updated to include that location. Hence, the accuracy of HLDS is dependent on the frequent updating of HLDB to reflect recent changes in hackers locations. This minimizes false negative and also false positive alarms. We could minimize false positive alarms by updating HLDB so as to remove any location that does not contain a hacker router anymore. In addition, we could minimize false negative result by adding locations that are new and which contain new hacker locations. False positive alarms may also be generated when a non-hacker router uses the location of the hacker router. This could be minimized by specifying the location of the hacker router more accurately and precisely using accurate Geographic Position Systems (GPSs).

HLDS is able to detect different attacks including Man-in-the-Middle (MITM), mail phishing, IP spoofing, packet sniffing. These attacks attack data confidentiality, authenticity and integrity. Hence, HLDS demonstrates if computer networks have any security vulnerabilities.

HLDS uses the following tools and technologies:

- **Packet Tracer:** Packet tracer is used as a simulator to simulate of complex networks used in our test bed. Packet Tracer will simulate most popular network devices such as Cisco switches and routers by allowing the network administrator to try different network topologies.
- **Scapy:** Scapy is used to develop the code for the validation experiments, and to develop the software plug-in. In addition, HLDS uses Scapy to decode, craft, generate, modify, manipulate, and capture packets through writing the appropriate Scapy code. Scapy is a good security tool that enables programmers to write code effectively in regards to number of lines.
- **Traceroute:** Traceroute is the program that shows you the route over the network between two systems, listing all the intermediate routers a connection must pass through to get to its destination. It can help you determine why your connections to a given server might be poor, and can often help you figure out where exactly the problem is. It also shows you how systems are connected to each other, letting you see how your ISP connects to the Internet as well as how the target system is connected [84].

HLDS will be used as a testing tool by network users, network administrators, ISPs, and hackers. Each of these users will use HLDS for different reasons and purposes.

The contribution of this thesis is the proposed HLDS tools, addition of a SRP that enhances IPv6 routers security. This SRP enables IPv6 routers to detect a lot of vulnerabilities and Malwares.

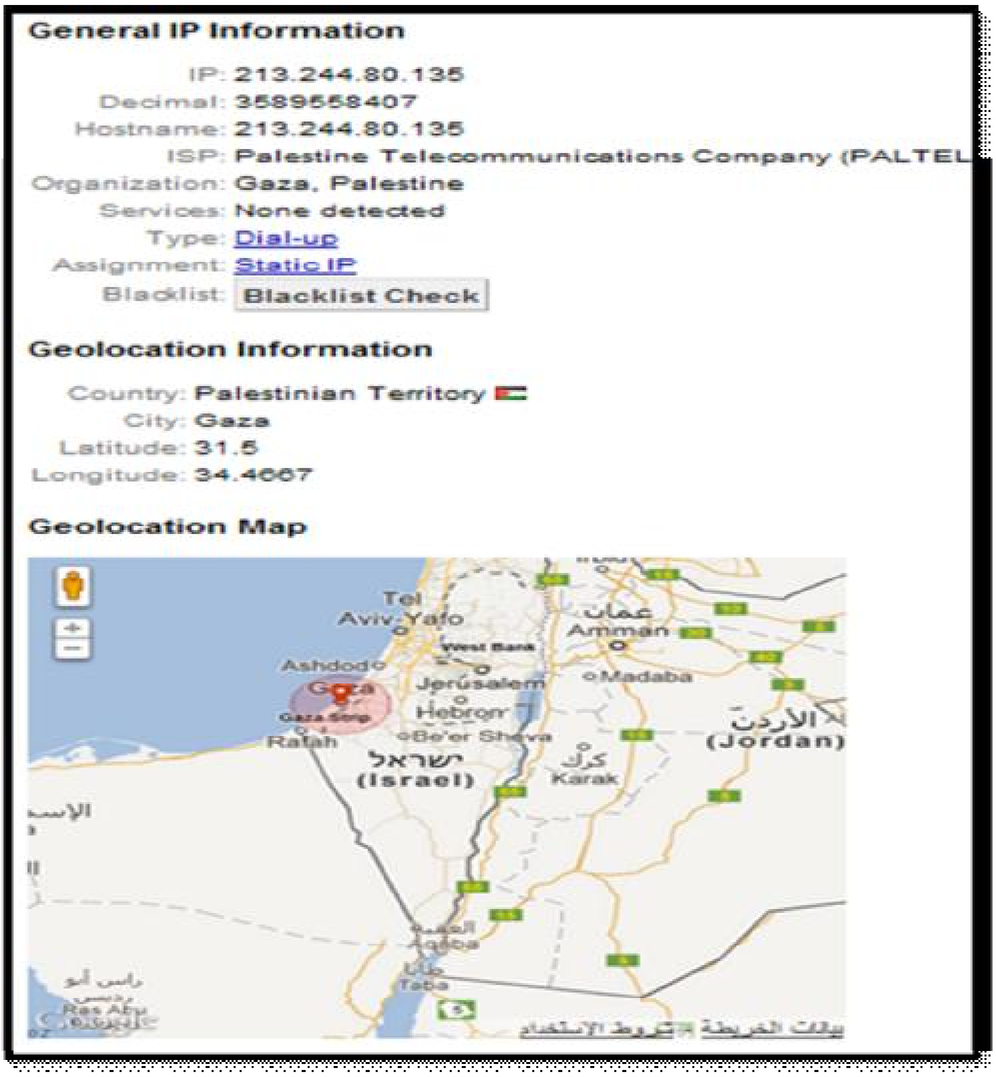
5.2 Hacker Location Detection System (HLDS) Software Router Plug-in (SRP)

After we install and configure the test bed, now we are ready to program the HLDS. Hackers can attack any computer in the world that is connected to the Internet, which means that someone could be probing your computer right now and checking it for weaknesses. Even worse is that you may catch a hacker in your computer, but as soon as he disconnects, he is gone.

Since hackers can appear and disappear at any time, there is nothing to stop them from attacking any computer they want, since it is highly unlikely that they will ever get caught. To eliminate the hacker's refuge in anonymity, HLDS developed a hack tracer, which can trace a hacker back to his Internet service provider (ISP) and possibly even find the hacker's location as well.

Knowing an IP address may identify the hacker's location, but the cryptic series of numbers that make up an IP address might not give you a clue where the hacker is located. The next time your firewall or IDS identifies a hacker's IP address, run it through McAfee Visual Trace, which is part of the McAfee Personal Firewall program [81] or “what is my IP address" [82] to see the hacker's approximate location on a world map. Figure 5.1 shows the geographical location of a particularly persistent hacker. After tracing enough hackers' IP addresses, you may be surprised to find that your personal computer in Gaza has been targeted by hackers in Korea, Canada, or Germany.

To further track down hacker activity, visit traceroute [83] and share hacker attempts on your computer with people all over the world. The more people who pool information about hackers, the more likely whatismyipaddress.com/traceroute-tool can identify specific types of attacks and pinpoint the IP addresses of the more active hackers in the world. Figure 5.2 identifies the top ISPs used by hackers around the world using traceroute.



General IP Information

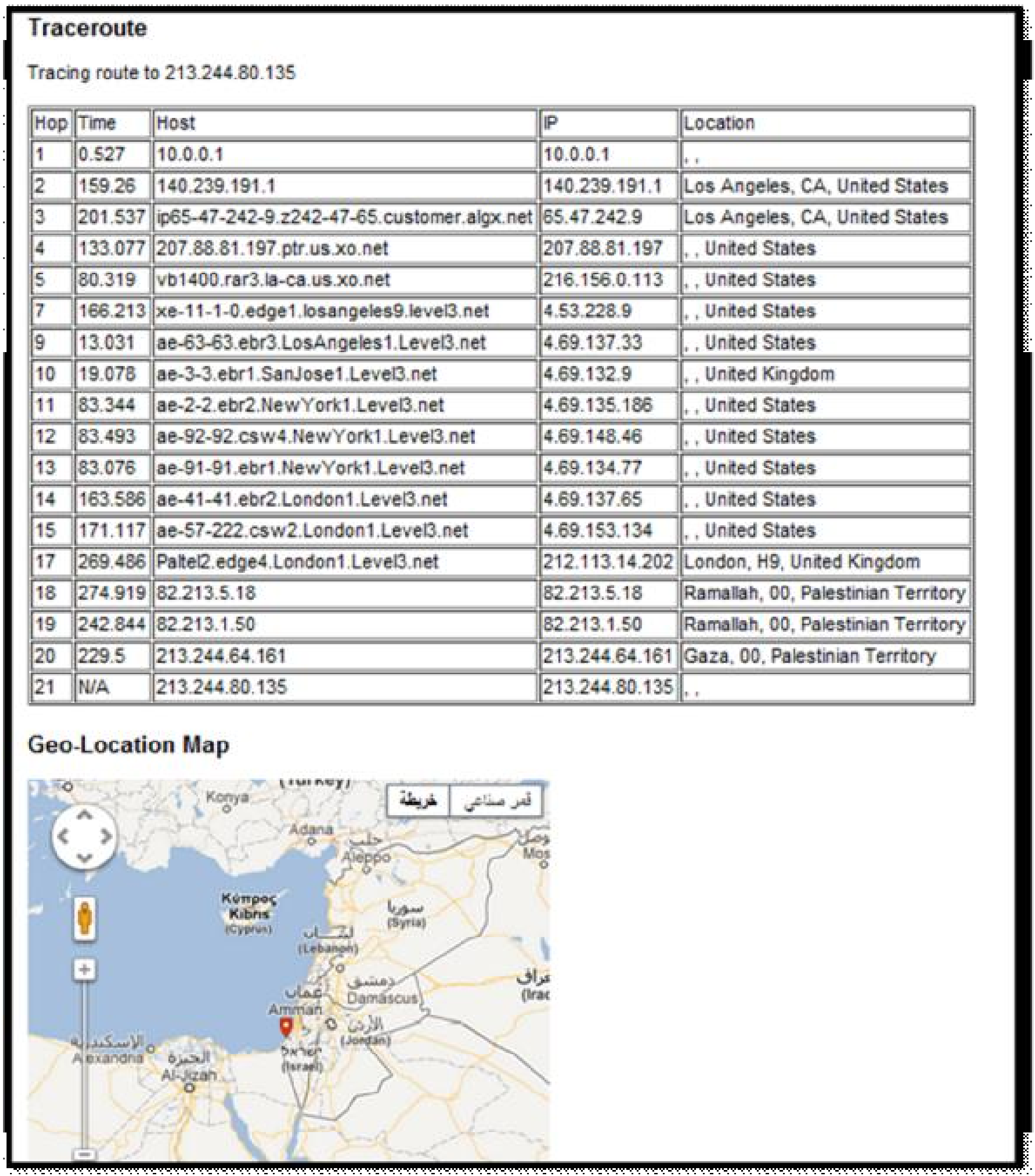
IP: 213.244.80.135
Decimal: 3589558407
Hostname: 213.244.80.135
ISP: Palestine Telecommunications Company (PALTEL
Organization: Gaza, Palestine
Services: None detected
Type: Dial-up
Assignment: Static IP
Blacklist: Blacklist Check

Geolocation Information

Country: Palestinian Territory
City: Gaza
Latitude: 31.5
Longitude: 34.4667

Geolocation Map

Figure 5.1: Geographical location of a particularly persistent hacker



Traceroute

Tracing route to 213.244.80.135

Hop	Time	Host	IP	Location
1	0.527	10.0.0.1	10.0.0.1	, ,
2	159.26	140.239.191.1	140.239.191.1	Los Angeles, CA, United States
3	201.537	ip65-47-242-9.z242-47-65.customer.algx.net	65.47.242.9	Los Angeles, CA, United States
4	133.077	207.88.81.197.ptr.us.xo.net	207.88.81.197	, , United States
5	80.319	vb1400.rar3.la-ca.us.xo.net	216.156.0.113	, , United States
7	166.213	xe-11-1-0.edge1.losangeles9.level3.net	4.53.228.9	, , United States
9	13.031	ae-63-63.ebr3.LosAngeles1.Level3.net	4.69.137.33	, , United States
10	19.078	ae-3-3.ebr1.SanJose1.Level3.net	4.69.132.9	, , United Kingdom
11	83.344	ae-2-2.ebr2.NewYork1.Level3.net	4.69.135.186	, , United States
12	83.493	ae-92-92.csw4.NewYork1.Level3.net	4.69.148.46	, , United States
13	83.076	ae-91-91.ebr1.NewYork1.Level3.net	4.69.134.77	, , United States
14	163.586	ae-41-41.ebr2.London1.Level3.net	4.69.137.65	, , United States
15	171.117	ae-57-222.csw2.London1.Level3.net	4.69.153.134	, , United States
17	269.486	Paltel2.edge4.London1.Level3.net	212.113.14.202	London, H9, United Kingdom
18	274.919	82.213.5.18	82.213.5.18	Ramallah, 00, Palestinian Territory
19	242.844	82.213.1.50	82.213.1.50	Ramallah, 00, Palestinian Territory
20	229.5	213.244.64.161	213.244.64.161	Gaza, 00, Palestinian Territory
21	N/A	213.244.80.135	213.244.80.135	, ,

Geo-Location Map

Figure 5.2: Identifying the top ISPs used by hackers around the world

While hackers can always choose when to attack a computer, that does not mean that your computer needs to be defenseless. With a good firewall, an intrusion-detection system, a securely patched operating system, and even a honeypot, you can protect your computer and possibly turn the tables on the hacker by tracing him and revealing his location for everyone to see. If enough people contact the ISP used by a particularly annoying hacker, the ISP may disconnect the hacker and force him to look for alternative ISPs. If this happens often enough, this may not discourage the hacker, but at least it can cause him enough trouble that he might restrict his activities and spare many potential victims from future attacks.

Figure 5.3 shows the block diagram of the proposed HLDS SRP used to detect, trace and locate hackers. Figure 5.3 shows that HLDS has four main modules:

- **HLDB Server Module:** HLDB Server Module contains the Hacker Location Database (HLDB) that includes the known locations of the hackers. HLDB is to be provided by a third party.
- **HLDB Update Module:** HLDB Update Module updates the HLDB based on the New hacker location information received.
- **Traceroute Module:** Traceroute Module is used to trace the route of the packet that is being monitored. It lists all routers that the packet passed through from source to destination. Information provided by traceroute includes the IP addresses of these routers as well as their geographic locations.
- **HLDS Monitor Module:** HLDS Monitor Module monitors the packets paths and generates an alarm if any packet that is being monitored has passed through a hacker router. This is determined by comparing the locations of the routers that the monitored packet passes through with the locations of the routers in the HLDB.

The inputs to the HLDS are:

- Packets to be monitored: These packets are submitted to the HLDS for checking whether they passed through a hacker router.
- New hacker location information: This information is used to update the HLDB regularly and frequently. The accuracy of HLDS depends on the acuuracy of the data in the HLDB.

Output of the HLDS is:

- Result: Results indicate whether the monitored packets passed through a hacker router or not. Results are used to warn the network managers, ISPs, and network users.

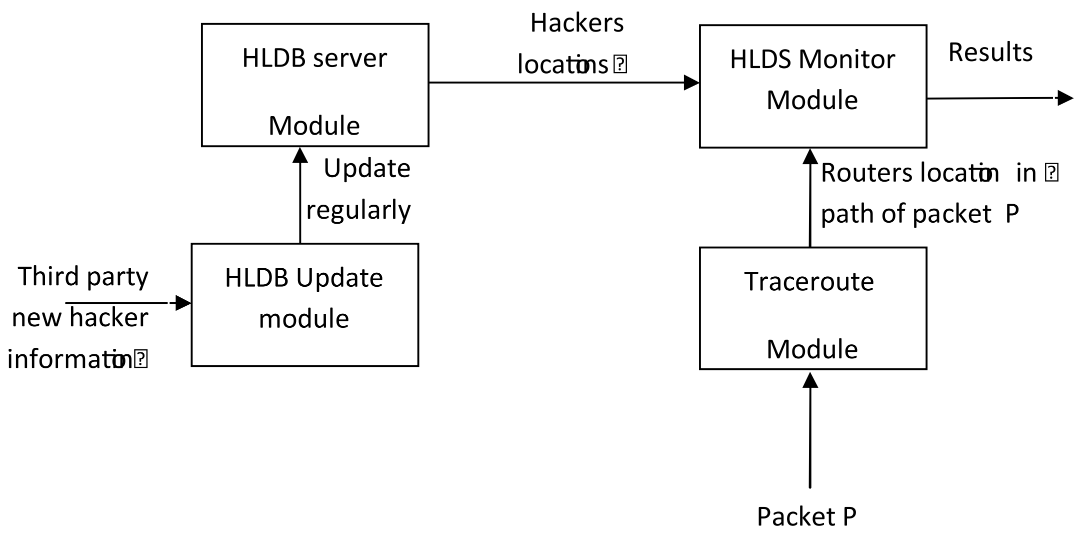


Figure 5.3: HLDS block diagram

Figure 5.4 shows the flowchart of the proposed Hacker Location Detection System (HLDS) SRP. The flowchart of Figure 5.4 shows that HLDS executes the steps shown in Code 5.1 which presents the pseudo-code of HLDS. Code 5.1 explains flowchart of Figure 5.4.

```
algorithm HLDS(input New hacker location information,
                        List of packets to be monitored PL,
                  output: monitoring results Result)
   1. Begin
   2. The HLDB is created/imported into the HLDB server if we used HLDS for the first
      time. Otherwise, HLDB is updated periodically using the HLDB Update Module.
      HLDB Update Module uses the input "New hacker location information" for this
      purpose.
   3. Packets are passed to HLDS for monitoring, Input "List of packets to be monitored
      PL" is used for this purpose.
   4. For each packet P in PL, HLDS executes the following steps:
      4.1Traceroute is invoked to supply HLDS with routers that P passed through from
           source to destination. Traceroute generate a list of routers RL.
      4.2 For each router R in RL
          4.2.1 HLDS Monitor Module compares location of R with hacker routers locations
                stored in HLDB.
          4.2.2 If Location of R == any hacker router location in HLDB then
                HLDS generates an alarm that packet P has been hacked.
                Result: "Packet has been hacked"
          4.2.3 Else if location of R != all hacker routers locations in HLDB then HLDS
                generates a message stating that packet P passed through safe routers.
                Result" "Packet passed through a safe route"
5. End algorithm
```

Code 5.1: HLDS pseudo-code

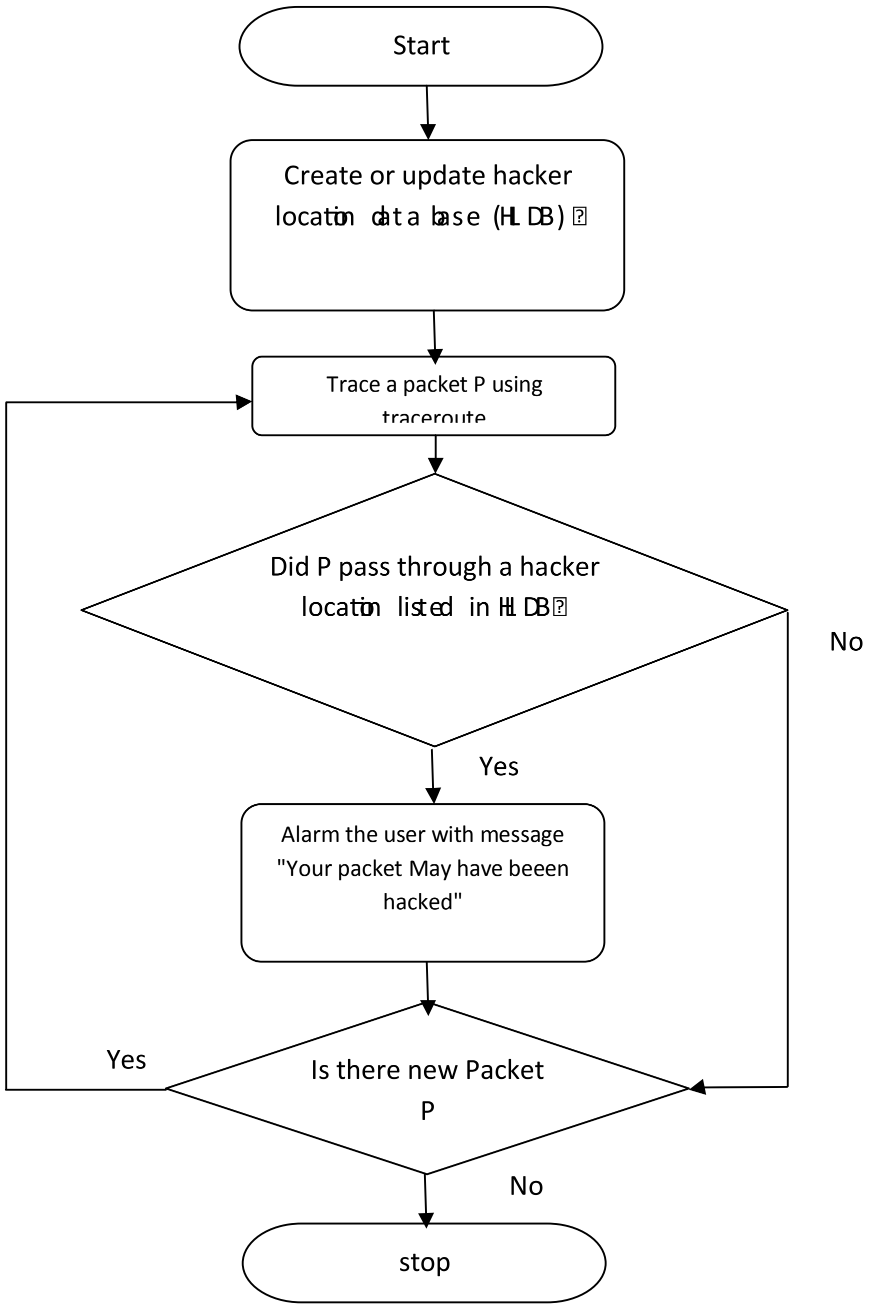

Start
Create or update hacker location data base (HLDB)
Trace a packet P using traceroute
Did P pass through a hacker location listed in HLDB
No
Yes
Alarm the user with message "Your packet May have beeen hacked"
Yes
Is there new Packet P
No
stop

Figure 5.4: HLDS flowchart

5.3 HLDS Test bed Development

The environment required for HLDS is strictly based on devices composed of machines running Ubuntu operating system. These machines will be used to demonstrate the scope and capabilities of HLDS.

HLDS uses the Ubuntu distribution of Linux operating system with Scapy installed on the host OS with packet tracer CISCO academy tool to simulate Cisco routers and switches. The following sections will explain the HLDS required environment in more detail.

5.3.1 Scapy Installation

The operating system used for HLDS is Ubuntu release 12.04.1 LTS. Ubuntu as the universal operating system has a package management system to provide packages in binary format. Scapy is one of those packages which are available through Ubuntu package management system. The installation of Scapy using Ubuntu package management system is simply to install python-Scapy with graphical user interface. Then we need to fetch "python-Scapy" binary package from the official Ubuntu package repository and install it.

Python-Scapy depends on other packages that need to be installed on the system before the package is to be installed.

We need to ensure that python 2.6 is installed as well as other required packages. An alternative to installing Scapy using a pre-build binary package is downloading the package from the maintainer's website (http://www.secdev.org/projects/scapy/). This will provide the most up-to-date release of Scapy.

HLDS users need to learn the basics of Scapy in order to develop tests and experiments using the tool. Scapy provides users with capabilities to generate, craft, modify, capture of packets through writing the appropriate code. It enables users to modify any field in the packet. A good starting point to learn Scapy is reference [85].

Note: Scapy requires more privileges than a regular user and must be ran as root.

5.3.2 Packet Tracer Installation and Setup

Packet Tracer is a graphical network simulator that allows simulation of complex networks [80]. Simply stated, Packet Tracer will simulate most popular network devices such as Cisco Switches and Routers by allowing the network administrator to try different network topologies. The simulator is a Graphical User Interface (GUI) that facilitates the Cisco IOSs. It simulates both desktops and servers computers with different architectures. The simulator will facilitate all network connections between all devices used. The installation of Packet Tracer is simply downloading the binary package from Packet Tracer web site of ref. [86]. Once downloaded, it is extracted to a location on the file system in Ubuntu package system and could be installed using the Ubuntu package management system. We will not provide detailed instruction on how to install Packet Tracer. In order to implement HLDS we installed and configured Packet Tracer properly. The version of Packet Tracer used for this thesis is 5.3.0.0088. Note that Packet Tracer uses official Cisco IOS images.

5.3.3 HLDS Environment Setup

At this point, all applications required for HLDS are installed and configured properly. This section provides detailed information about the network topology used to test, validate and experiment HLDS. The topology will be created using Packet Tracer to simulate end nodes such as Windows 7 client and Linux. Figure 5.5 shows the first test bed network topology. Figure 5.5 was developed using packet tracer. The topology consists of an Internet cloud which connects routers used in the test bed. The red points that connect network devices mean that devices are not configured yet. A router in the topology is the router-on-a-stick that will be responsible for routing internal traffic between the various VLANs and external traffic by sending frames to the next hop which is the tap on the host machine. Traffic from one VLAN must be routed using routers before reaching its final destination.

Gaza router in Figure 5.5 has two interfaces:

- The first interface is Fa0/0 with global IPv6 address of 2001:BDA:0:1::2/64 (local link address FE80::2) which is connected to the layer two switch that connects the Hacker Location DataBase (HLDB) server and the victim.
- The second interface is Fa0/1 with global IPv6 address of 2001:BDA:0:2::22/64 (local link address FE80::22) which is connected to Internet cloud tap.

Lamaca router has three interfaces:

- Fa0/0 with global IPv6 address of 2001:BDA:0:4::4/64 (local link address FE80::4) which is connected to the cloud0 tap.
- Fa0/1 with global IPv6 address of 2001:BDA:0:3::5/64 (local link address FE80::5) which is connected to hacker PC.
- Fa1/0 with global IPv6 address of 2001:BDA:0:5::6/64 (local link address FE80::6) which is connected to the cloud1 tap.

London router has two interfaces:

- Fa0/0 with global IPv6 address of 2001:BDA:0:6::7/64 (local link address FE80::7) which is connected to the cloud1 tap.
- Fa0/1 with global IPv6 address of 2001:BDA:0:7::8/64 (local link address FE80::8) which is connected to destination PC.

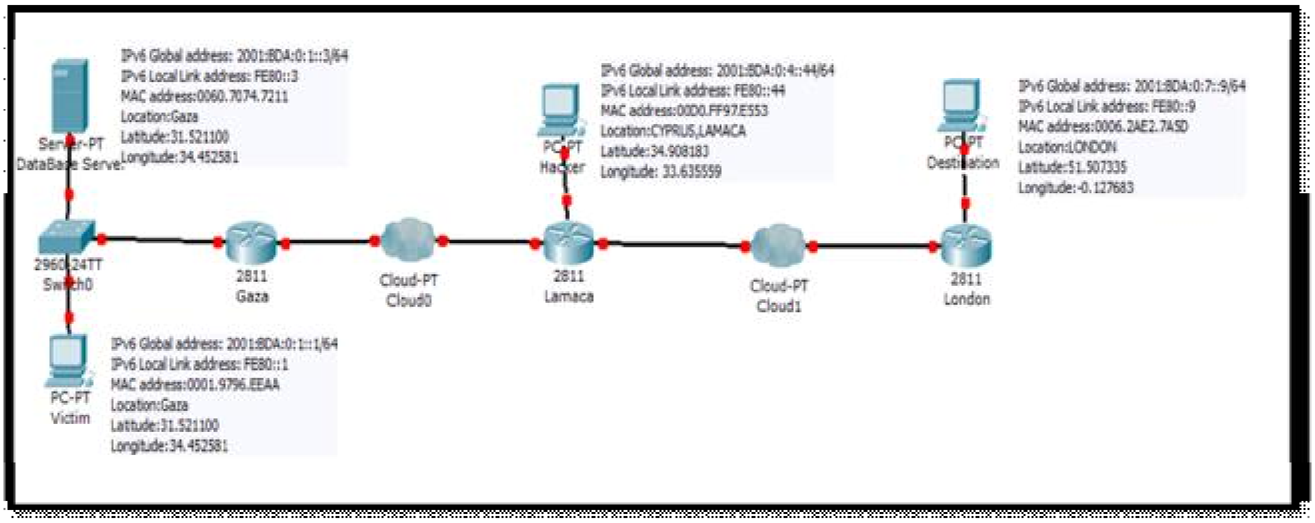
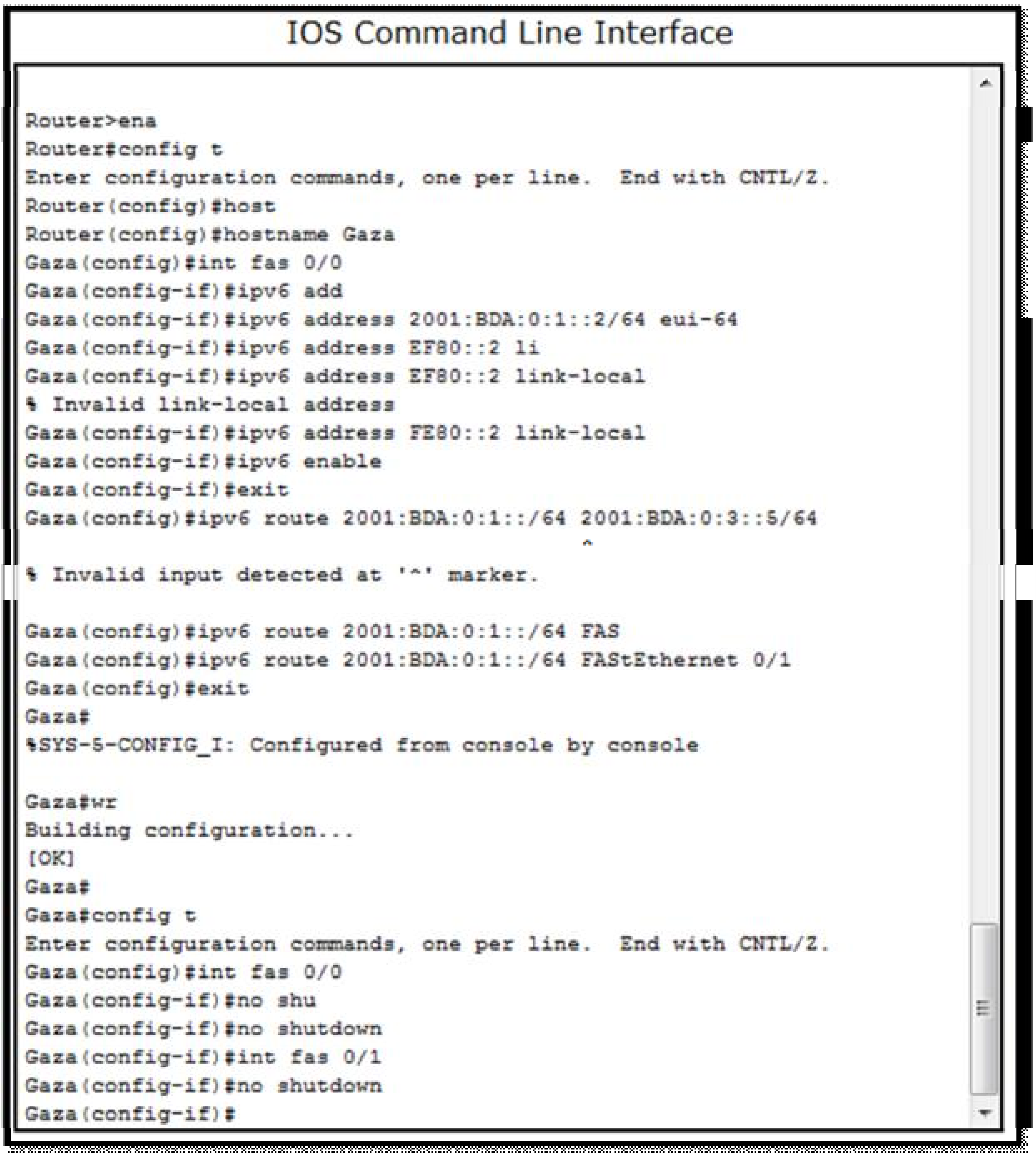

IPv6 Global address: 2001:BDA:0:1::3/64
IPv6 Local Link address: FE80::3
MAC address:0060.7074.7211
Location:Gaza
Latitude:31.521100
Longitude:34.452581
Server-PT
DataBase Server
IPv6 Global address: 2001:BDA:0:4::44/64
IPv6 Local Link address: FE80::44
MAC address:00D0.FF97.E553
Location:CYPRUS,LAMACA
Latitude:34.908183
Longitude: 33.635559
PC-PT
Hacker
IPv6 Global address: 2001:BDA:0:7::9/64
IPv6 Local Link address: FE80::9
MAC address:0006.2AE2.7A5D
Location:LONDON
Latitude:51.507335
Longitude:-0.127683
PC-PT
Destination
2960-24TT
Switch0
2811
Gaza
Cloud-PT
Cloud0
2811
Lamaca
Cloud-PT
Cloud1
2811
London
IPv6 Global address: 2001:BDA:0:1::1/64
IPv6 Local Link address: FE80::1
MAC address:0001.9796.EEAA
Location:Gaza
Latitude:31.521100
Longitude:34.452581
PC-PT
Victim

Figure 5.5: First test bed topology of HLDS

In Figure 5.5 we configured the network devices as follows:

- The HLDB server was configured with global IPv6 address of 2001:BDA:0:1::3/64 (local link address FE80::3). We installed on it the Hacker Location Database (HLDB) which defines the locations of known hackers (or Hacker Homes). Each entry in the HLDB includes the 4 tuple (Hacker IP address, Hacker MAC address, Hacker's location –latitude, Hacker's location – longitude). IP and MAC addresses of the hacker may be changed by the hacker. Thus, HLDS does not depend only on these addresses but also on its geographic location. Geographic location of the hacker cannot be spoofed by the hacker. The HLDB server contains virtual database.
- The victim (sender of the packet) PC with global IPv6 address of 2001:BDA:0:1::1/64 (local link address FE80::1).
- The Cisco Router model 2811 is used to build Gaza router. Figure 5.6 shows the configuration of Gaza router.
- The Cisco Router model 2811 is used to build a host with the name LAMACA. Figure 5.7 shows the configuration of LAMACA router.
- The Cisco Router model 2811 is used to build a host with the name London router. Figure 5.8 shows the configuration of London router.
- The hacker PC is configured with IPv6 address of global IPv6 address of 2001:BDA:0:4::44/64 (local link address FE80::44).
- The destination PC is configured with global IPv6 address of 2001:BDA:0:7::9/64 (local link address FE80::9)

After the configuration our test bed as shown above, network topology is configured properly. Then every red connected port in Figure 5.5 is changed to green as shown in Figure 5.9.

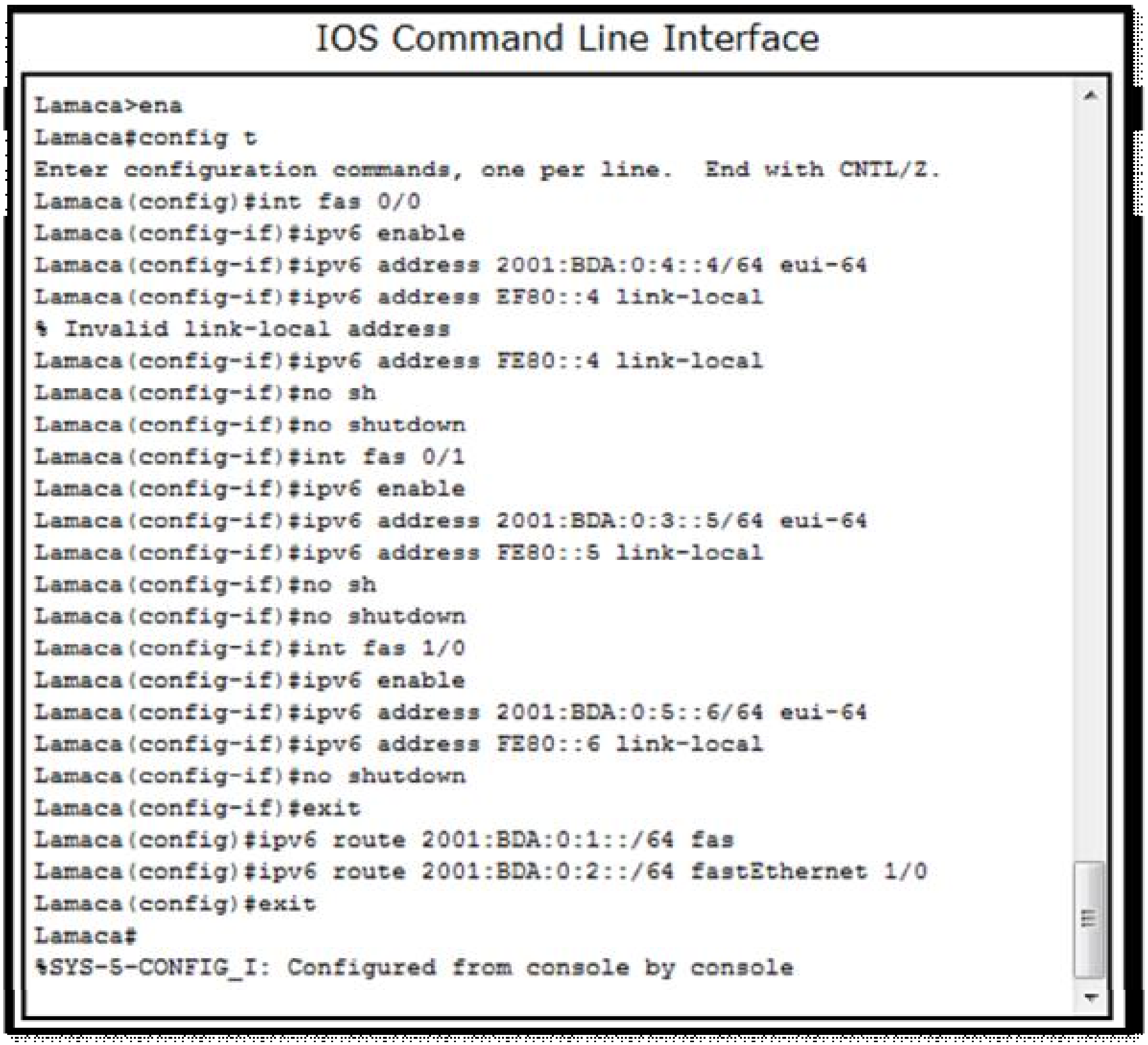


```
Router>ena
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#host
Router(config)#hostname Gaza
Gaza(config)#int fas 0/0
Gaza(config-if)#ipv6 add
Gaza(config-if)#ipv6 address 2001:BDA:0:1::2/64 eui-64
Gaza(config-if)#ipv6 address EF80::2 li
Gaza(config-if)#ipv6 address EF80::2 link-local
% Invalid link-local address
Gaza(config-if)#ipv6 address FE80::2 link-local
Gaza(config-if)#ipv6 enable
Gaza(config-if)#exit
Gaza(config)#ipv6 route 2001:BDA:0:1::/64 2001:BDA:0:3::5/64
                                              ^
% Invalid input detected at '^' marker.

Gaza(config)#ipv6 route 2001:BDA:0:1::/64 FAS
Gaza(config)#ipv6 route 2001:BDA:0:1::/64 FAStEthernet 0/1
Gaza(config)#exit
Gaza#
%SYS-5-CONFIG_I: Configured from console by console

Gaza#wr
Building configuration...
[OK]
Gaza#
Gaza#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Gaza(config)#int fas 0/0
Gaza(config-if)#no shu
Gaza(config-if)#no shutdown
Gaza(config-if)#int fas 0/1
Gaza(config-if)#no shutdown
Gaza(config-if)#
```

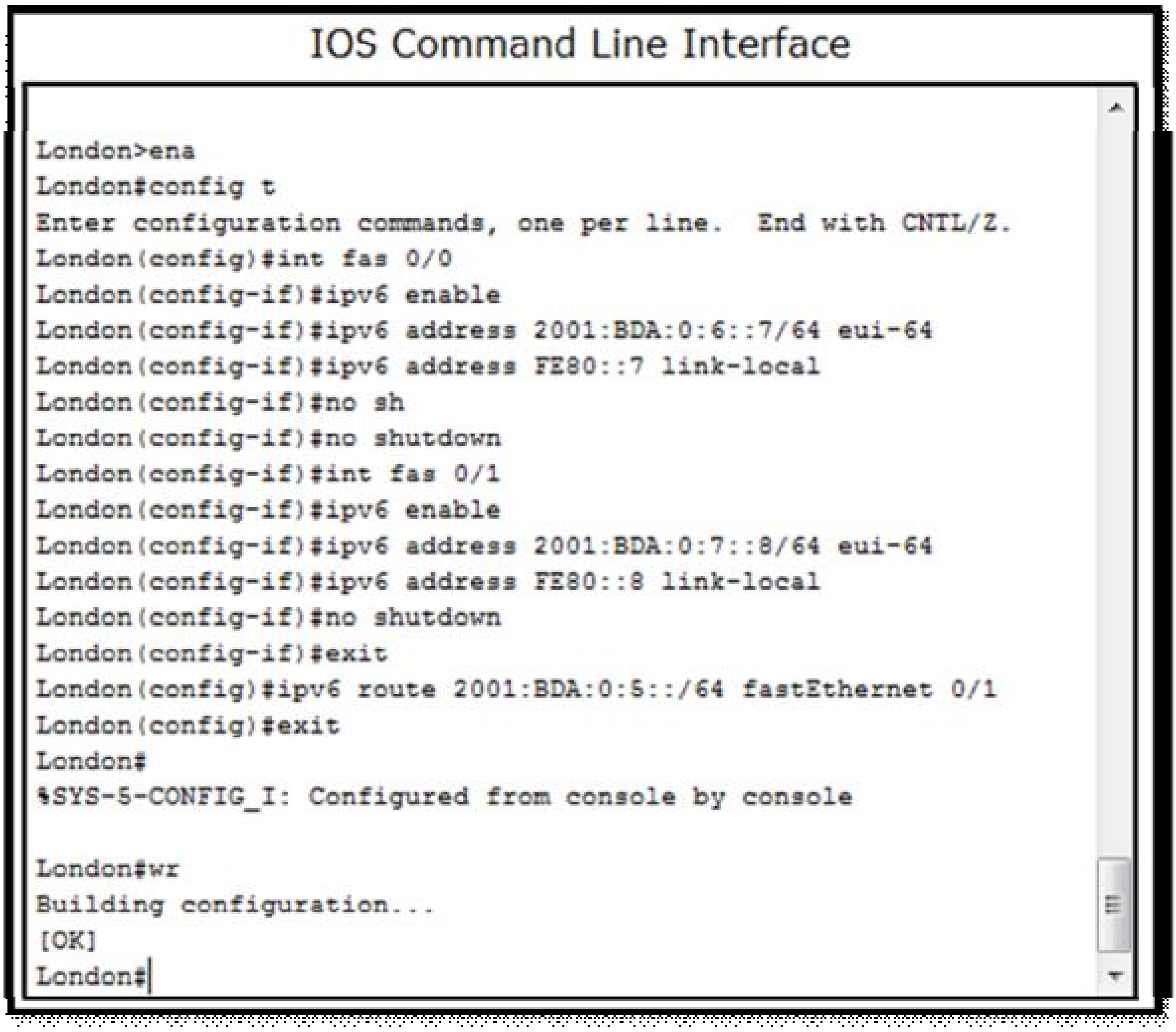
Figure 5.6: Gaza router configuration



```
Lamaca>ena
Lamaca#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Lamaca(config)#int fas 0/0
Lamaca(config-if)#ipv6 enable
Lamaca(config-if)#ipv6 address 2001:BDA:0:4::4/64 eui-64
Lamaca(config-if)#ipv6 address EF80::4 link-local
% Invalid link-local address
Lamaca(config-if)#ipv6 address FE80::4 link-local
Lamaca(config-if)#no sh
Lamaca(config-if)#no shutdown
Lamaca(config-if)#int fas 0/1
Lamaca(config-if)#ipv6 enable
Lamaca(config-if)#ipv6 address 2001:BDA:0:3::5/64 eui-64
Lamaca(config-if)#ipv6 address FE80::5 link-local
Lamaca(config-if)#no sh
Lamaca(config-if)#no shutdown
Lamaca(config-if)#int fas 1/0
Lamaca(config-if)#ipv6 enable
Lamaca(config-if)#ipv6 address 2001:BDA:0:5::6/64 eui-64
Lamaca(config-if)#ipv6 address FE80::6 link-local
Lamaca(config-if)#no shutdown
Lamaca(config-if)#exit
Lamaca(config)#ipv6 route 2001:BDA:0:1::/64 fas
Lamaca(config)#ipv6 route 2001:BDA:0:2::/64 fastEthernet 1/0
Lamaca(config)#exit
Lamaca#
%SYS-5-CONFIG_I: Configured from console by console
```

-

Figure 5.7: Lamaca router configuration



```
London>ena
London#config t
Enter configuration commands, one per line.  End with CNTL/Z.
London(config)#int fas 0/0
London(config-if)#ipv6 enable
London(config-if)#ipv6 address 2001:BDA:0:6::7/64 eui-64
London(config-if)#ipv6 address FE80::7 link-local
London(config-if)#no sh
London(config-if)#no shutdown
London(config-if)#int fas 0/1
London(config-if)#ipv6 enable
London(config-if)#ipv6 address 2001:BDA:0:7::8/64 eui-64
London(config-if)#ipv6 address FE80::8 link-local
London(config-if)#no shutdown
London(config-if)#exit
London(config)#ipv6 route 2001:BDA:0:5::/64 fastEthernet 0/1
London(config)#exit
London#
%SYS-5-CONFIG_I: Configured from console by console

London#wr
Building configuration...
[OK]
London#
```

Figure 5.8: London router configuration

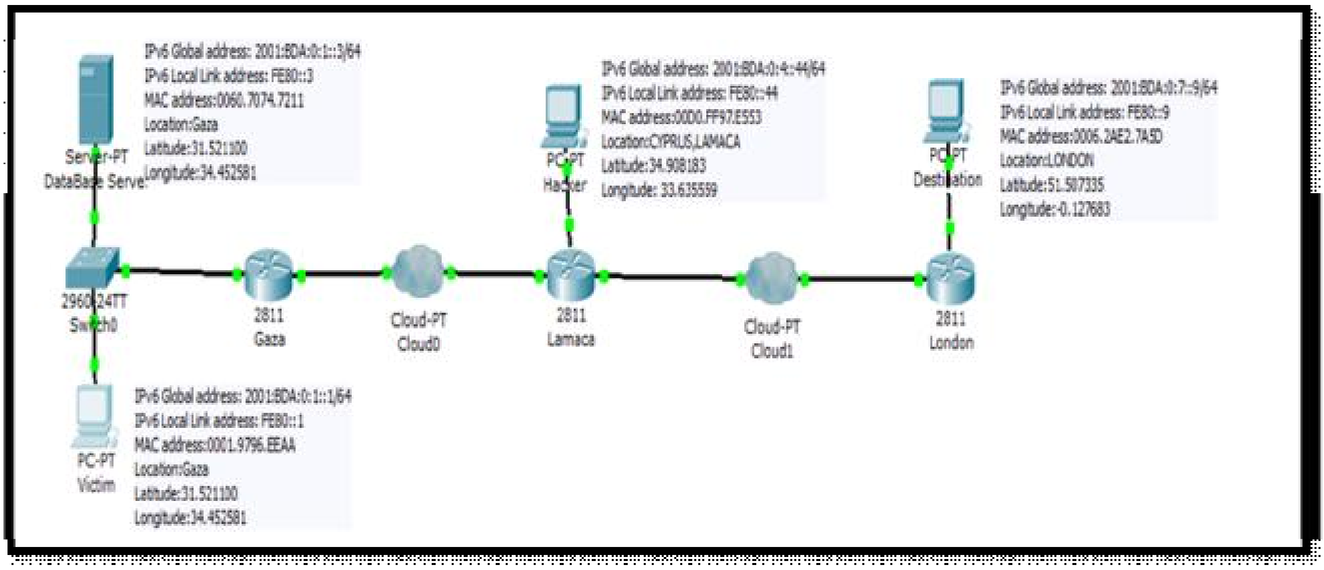

IPv6 Global address: 2001:BDA:0:1::3/64
IPv6 Local Link address: FE80::3
MAC address:0060.7074.7211
Location:Gaza
Latitude:31.521100
Longitude:34.452581
Server-PT
DataBase Server
IPv6 Global address: 2001:BDA:0:4::44/64
IPv6 Local Link address: FE80::44
MAC address:00D0.FF97.E553
Location:CYPRUS,LAMACA
Latitude:34.908183
Longitude: 33.635559
PC-PT
Hacker
IPv6 Global address: 2001:BDA:0:7::9/64
IPv6 Local Link address: FE80::9
MAC address:0006.2AE2.7A5D
Location:LONDON
Latitude:51.507335
Longitude:-0.127683
PC-PT
Destination
2960-24TT
Switch0
2811
Gaza
Cloud-PT
Cloud0
2811
Lamaca
Cloud-PT
Cloud1
2811
London
IPv6 Global address: 2001:BDA:0:1::1/64
IPv6 Local Link address: FE80::1
MAC address:0001.9796.EEAA
Location:Gaza
Latitude:31.521100
Longitude:34.452581
PC-PT
Victim

Figure 5.9: Test bed topology of Figure 5.5 (first test bed of HLDS) after being configured

Chapter 6
Hacker Location Detection System (HLDS) Validation

6.1 Introduction

In this chapter we validate the proposed HLDS. We develop experiments to measure the performance of HLDS. We used some performance metrics to measure the performance of HLDS. The performance metrics used are:

- **Accuracy:** Accuracy measures the rate of generating correct results. If there are N packets being monitored by HLDS and there are x of these packets for which HLDS predicts correctly if the path of these packets is hacked or not then:
  accuracy (%) = (x/N)*100%
  Accuracy is also called Detection Rate (DR) which is defined as the number of intrusion instances detected by the system (True Positive) divided by the total number of intrusion instances present in the test set.
- **False positive rate:** This is the rate at which HLDS generates an alarm that a packet has been hacked while in reality the packet has not been hacked. If HLDS tests N packets for possible hacking. In addition, HLDS generates N1 positive alarms that N1 packets have been hacked out of N packets. If x of these N1 alarms are false, then:
  False positive rate (%) = (x/N1)*100 %
- **False negative rate:** This is the rate at which HLDS dos not generates an alarm that a packet has been hacked while in reality the packet has been hacked. If HLDS tests N packets for possible hacking. In addition, HLDS generates N1 negative alarms that N1 packets have been hacked out of N packets. If x of these N1 alarms are false, then:
  False negative rate (%) = (x/N1)*100 %

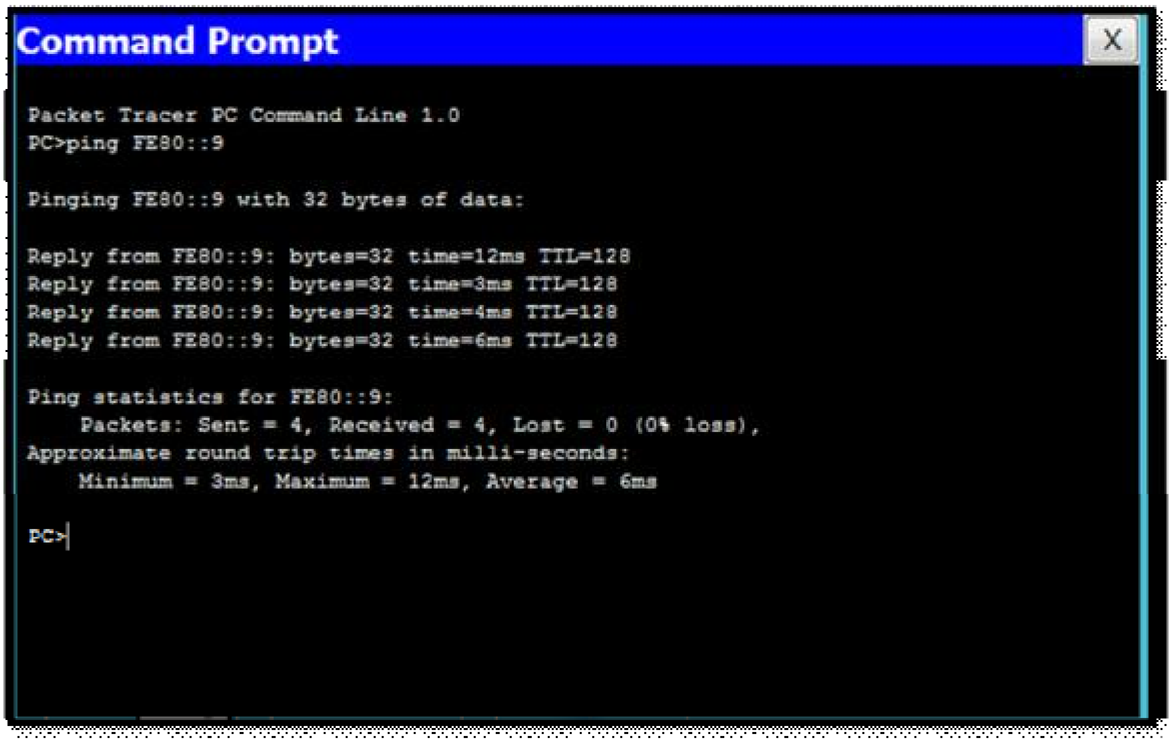
In order to validate HLDS we compare the performance results of HLDS with performance of similar tools in the literature. The comparison shows how HLDS performs compared to other similar tools.

To test HLDS, we develop test beds. These test beds are network topologies that connect two nodes, the source node and the destination node. The source and node in the test bed topology have multiple routers between them. These routers represent the path connecting the source and destinations.

6.2 HLDS Results of First Test Bed

The first test bed we developed was presented in Section 5.3.3. It connects the source node which is located in Gaza and the destination node which is located in London.

We used ping command to test the health of the path connecting source and destination of the first test bed which is shown in Figure 5.5. The result of the ping is shown in Figure 6.1. Figure 6.1 shows that there is a network connection between Gaza (source node) and London (destination node).



```
Packet Tracer PC Command Line 1.0
PC>ping FE80::9

Pinging FE80::9 with 32 bytes of data:

Reply from FE80::9: bytes=32 time=12ms TTL=128
Reply from FE80::9: bytes=32 time=3ms TTL=128
Reply from FE80::9: bytes=32 time=4ms TTL=128
Reply from FE80::9: bytes=32 time=6ms TTL=128

Ping statistics for FE80::9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 12ms, Average = 6ms

PC>
```

Figure 6.1: ping from Gaza node to London node of the test bed of Figure 5.5 (first test bed)

In order to test HLDS we create two paths. The first path is a healthy path whose routers between source and destination are safe and not hacked, i.e., no router's location of these routers is in the HLDB. The second path is a hacked path where one of the routers in the path between source and destination is hacked, i.e., its location is in the HLDB.

Then, we craft packets using Scapy. These packets will be tested for hacking through passing through both the healthy path and hacked path. We develop code using Scapy for this purpose. The result of HLDS Monitoring Module could be one of the following:

- HLDS does not generate an alarm when the packet passes through the healthy path. In this case HLDS works correctly and accurately.

- HLDS generates an alarm when the packet passes through the healthy path. In this case HLDS works incorrectly and produces a wrong result. This case is called false positive since HLDS should not generate an alarm but it does. HLDS produces a false positive result when a safe router in the path has its location stored in HLDB. This is due to this safe router being too close geographically to a hacked router whose location is stored in HLDB. The case of false positive could also be that a hacked router does not anymore use its location that has been previously stored in the HLDB.
- HLDS generates an alarm when the packet passes through the hacked path. In this case HLDS works correctly and accurately.
- HLDS does not generate an alarm when the packet passes through the hacked path. In this case HLDS works incorrectly and produces a wrong result. This case is called false negative since HLDS should generate an alarm but it does not. HLDS produces a false negative result when a hacker router in the path does not have its location stored in HLDB. This is due to this hacked router being new such that it is not identified yet as a hacked router and hence its location is not yet stored in the HLDB. The case of false negative could also be that a hacked router whose location that is stored in the HLDB is an old location and this hacked router does not anymore use this location.

The next step is to run HLDS Code shown in Code 6.1. In Code 6.1, HLDS runs traceroute program between source node and destination node. Code 6.1 shows the HLDS Scapy program needed to implement this task. In Code 6.1 HLDS invokes traceroute. Code 6.1 works for the test bed case of Figure 5.5. This test bed was presented in Section 5.3.3. Code 6.1 does the following:

- HLDS crafts a packet at the source node for being tested.
- HLDS creates a healthy path that connects source and destination that does not include a hacked router whose location is in the HLDB.
- HLDS creates a hacked path that connects source and destination through a sequence of routers, where one of these routers is a hacked router whose location is stored in the HLDB.
- HLDS invokes traceroute to trace the crafted packet. Traceroute will generate a list IP addresses of routers that the packet passed through.
- HLDS converts the IP addresses of the routers generated by traceroute into geographical locations.

- HLDS checks whether HLDB contains the location of any router in the path between source and destination. If HLDB contains the location of any router in the path between source and destination, then, HLDS generates an alarm. Otherwise, HLDS does not generate an alarm.

```
### Create Victim (source node) IPv6 Packet
          i=IPv6()
          i.src="2001:bda:0:1::1"
          i.dst="2001:bda:0:7::9"
          q=ICMPv6EchoRequest()
          p=(i/q)
          sr1(p)
### Create Victim (source node) IPv6 Packet path without passing through hacker location
          h1=IPv6ExtHdrRouting()
          h1.addresses=["2001:bda:0:1::2","2001:bda:0:2::2","2001:bda:0:3::5","2001:bda:0:5::6","2001:bda:0:6
          ::7","2001:bda:0:7::8"]
### Create Victim (source node) IPv6 Packet path that passes  through hacker’s router
          h2=IPv6ExtHdrRouting()
          h2.addresses=["2001:bda:0:1::2","2001:bda:0:2::2","2001:bda:0:3::5","2001:bda:0:4::4","2001:bda:0:5
          ::6","2001:bda:0:6::7","2001:bda:0:7::8"]
### Traceroute Victim (source node) IPv6 Packet
          waypoint = "2001:bda:0:1::2"
          target = "2001:bda:0:7::9"
          t1= traceroute6(waypoint, minttl=15 , maxttl=34, l4=h1 (addresses=[target])
        /ICMPv6EchoRequest(data=RandString(7)))
          t2= traceroute6(waypoint, minttl=15 , maxttl=34, l4=h1 (addresses=[target])
        /ICMPv6EchoRequest(data=RandString(7)))
### Convert traced IPv6 addresses to locations
          L1=Convtolocation(t1)
          L2=Convtolocation(t2)
### check for hacking
        for i in range(2):
                    for addr in Li:
                              If addr=hackerLocation
                                        print('Your packet may have been hacked')
                              els continue
                    end for
        end for
```

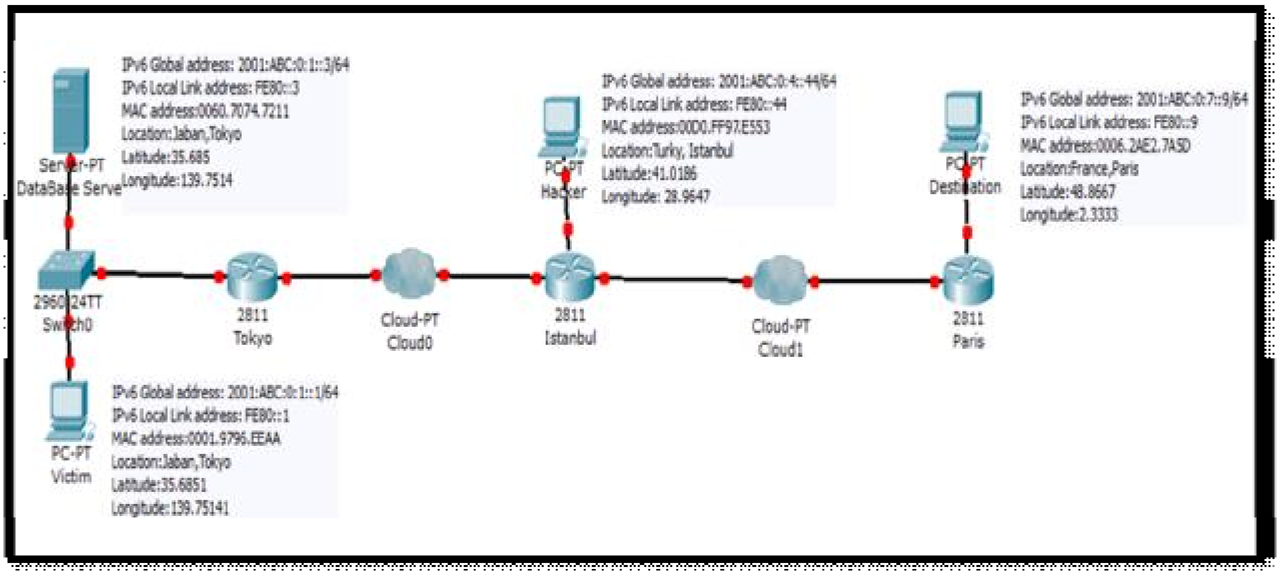
Code 6.1: HLDS Scapy program that tests test bed topology of Figure 5.5 (first test bed)

When running HLDS program of Code 6.1 and force the crafted packet to pass through the hacked path, HLDS generates the alarm message `'Your packet may have been hacked'` and sends this message to the source node. When running HLDS program of Code 6.1 and force the crafted packet to pass through the safe path, HLDS does not generate any alarm message. This means that HLDS works correctly for the first test bed topology.

6.3 HLDS Results of Second Test Bed

In this section we present a second test bed. Here we examine HLDS with another test bed to see if it works correctly for a different topology than that which was presented in the first topology of Figure 5.5.

In the second test bed topology the victim node (source node) is located in Tokyo in Japan with latitude of 35.6851 and longitude of 139.75141. The hacker's router is located in Istanbul in Turkey with latitude of 41.0186 and longitude of 28.9647. The destination node is located in Paris in France with latitude of 48.8667 and longitude of 2.3333. This topology is shown in Figure 6.2. Figure 6.2 was developed using packet tracer.



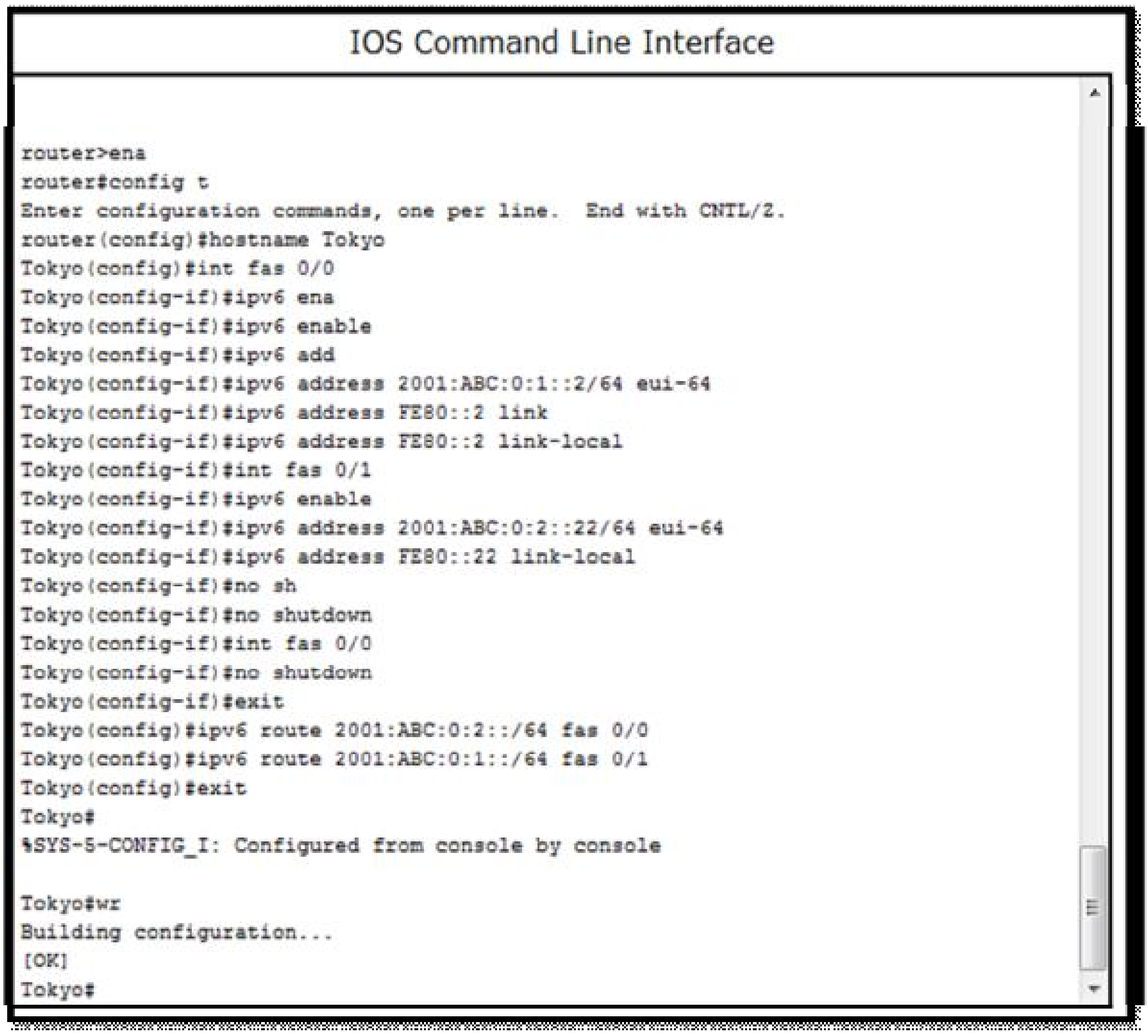
IPv6 Global address: 2001:ABC:0:1::3/64
IPv6 Local Link address: FE80::3
MAC address:0060.7074.7211
Location:Jaban,Tokyo
Latitude:35.685
Longitude:139.7514
Server-PT
DataBase Serve
IPv6 Global address: 2001:ABC:0:4::44/64
IPv6 Local Link address: FE80::44
MAC address:00D0.FF97.E553
Location:Turky, Istanbul
Latitude:41.0186
Longitude: 28.9647
PC-PT
Hacker
IPv6 Global address: 2001:ABC:0:7::9/64
IPv6 Local Link address: FE80::9
MAC address:0006.2AE2.7A5D
Location:France,Paris
Latitude:48.8667
Longitude:2.3333
PC-PT
Destination
2960-24TT
Switch0
2811
Tokyo
Cloud-PT
Cloud0
2811
Istanbul
Cloud-PT
Cloud1
2811
Paris
PC-PT
Victim
IPv6 Global address: 2001:ABC:0:1::1/64
IPv6 Local Link address: FE80::1
MAC address:0001.9796.EEAA
Location:Jaban,Tokyo
Latitude:35.6851
Longitude:139.75141

Figure 6.2: Second test bed topology of HLDS

The topology of Figure 6.2 consists of an Internet cloud which connects routers used in the second test bed. The red points that connect network devices mean that devices are not configured yet. So we configure each device as follows:

1. The HLDB server is configured with the global IPv6 address of 2001:ABC:0:1::3/64 (local link address FE80::3). We install on this server the HLDB. HLDB define the locations (or Hacker Homes), their IPs and MAC addresses (Hackers may be change their IP address or MAC Address frequently but they can not change their locations).
2. The source node PC is configured with the global IPv6 address of 2001:ABC:0:1::1/64 and a local link address of FE80::1.
3. The Cisco Router model 2811 is used to build the source node that is located Tokyo. Figure 6.3 shows the configuration of Tokyo router.
4. The Cisco Router model 2811 is used to build Istanbul router. Figure 6.4 shows the configuration of Istanbul router.
5. The Cisco Router model 2811 is used to build Paris router. Figure 6.5 shows the configuration of Istanbul router.
6. The hacker's router is configured with the global IPv6 address of 2001:ABC:0:4::44/64 and local link address of FE80::44.
7. The destination node PC is configured with the global IPv6 address of 2001:ABC:7::9/64 and local link address of FE80::9.

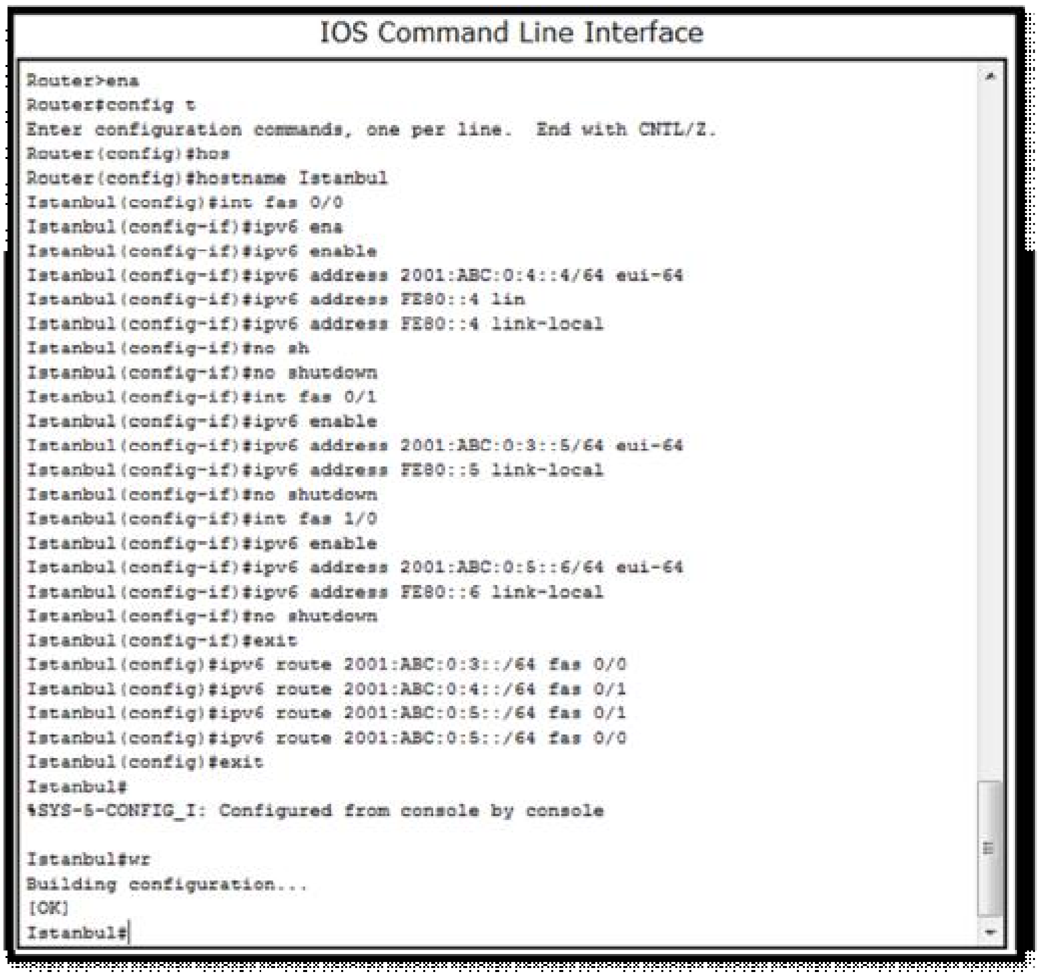
After the configuration our second test bed as shown above, network topology is configured properly. Then every red connected port in Figure 6.2 is changed to green as shown in Figure 6.6.



```
router>ena
router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
router(config)#hostname Tokyo
Tokyo(config)#int fas 0/0
Tokyo(config-if)#ipv6 ena
Tokyo(config-if)#ipv6 enable
Tokyo(config-if)#ipv6 add
Tokyo(config-if)#ipv6 address 2001:ABC:0:1::2/64 eui-64
Tokyo(config-if)#ipv6 address FE80::2 link
Tokyo(config-if)#ipv6 address FE80::2 link-local
Tokyo(config-if)#int fas 0/1
Tokyo(config-if)#ipv6 enable
Tokyo(config-if)#ipv6 address 2001:ABC:0:2::22/64 eui-64
Tokyo(config-if)#ipv6 address FE80::22 link-local
Tokyo(config-if)#no sh
Tokyo(config-if)#no shutdown
Tokyo(config-if)#int fas 0/0
Tokyo(config-if)#no shutdown
Tokyo(config-if)#exit
Tokyo(config)#ipv6 route 2001:ABC:0:2::/64 fas 0/0
Tokyo(config)#ipv6 route 2001:ABC:0:1::/64 fas 0/1
Tokyo(config)#exit
Tokyo#
%SYS-5-CONFIG_I: Configured from console by console

Tokyo#wr
Building configuration...
[OK]
Tokyo#
```

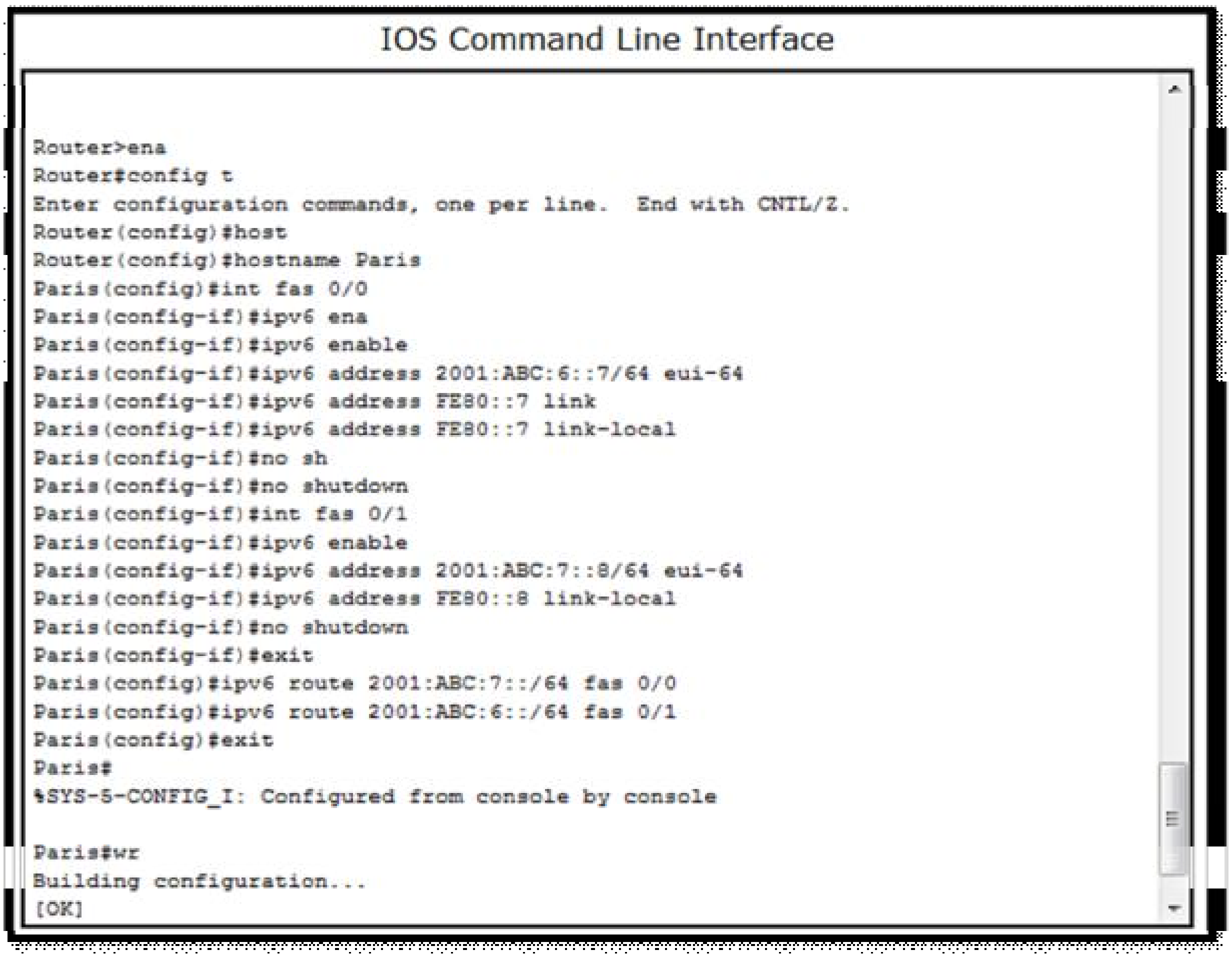
Figure 6.3: Tokyo router configuration



```
Router>ena
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hos
Router(config)#hostname Istanbul
Istanbul(config)#int fas 0/0
Istanbul(config-if)#ipv6 ena
Istanbul(config-if)#ipv6 enable
Istanbul(config-if)#ipv6 address 2001:ABC:0:4::4/64 eui-64
Istanbul(config-if)#ipv6 address FE80::4 lin
Istanbul(config-if)#ipv6 address FE80::4 link-local
Istanbul(config-if)#no sh
Istanbul(config-if)#no shutdown
Istanbul(config-if)#int fas 0/1
Istanbul(config-if)#ipv6 enable
Istanbul(config-if)#ipv6 address 2001:ABC:0:3::5/64 eui-64
Istanbul(config-if)#ipv6 address FE80::5 link-local
Istanbul(config-if)#no shutdown
Istanbul(config-if)#int fas 1/0
Istanbul(config-if)#ipv6 enable
Istanbul(config-if)#ipv6 address 2001:ABC:0:5::6/64 eui-64
Istanbul(config-if)#ipv6 address FE80::6 link-local
Istanbul(config-if)#no shutdown
Istanbul(config-if)#exit
Istanbul(config)#ipv6 route 2001:ABC:0:3::/64 fas 0/0
Istanbul(config)#ipv6 route 2001:ABC:0:4::/64 fas 0/1
Istanbul(config)#ipv6 route 2001:ABC:0:5::/64 fas 0/1
Istanbul(config)#ipv6 route 2001:ABC:0:6::/64 fas 0/0
Istanbul(config)#exit
Istanbul#
%SYS-5-CONFIG_I: Configured from console by console

Istanbul#wr
Building configuration...
[OK]
Istanbul#
```

Figure 6.4: Istanbul router configuration



```
Router>ena
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#host
Router(config)#hostname Paris
Paris(config)#int fas 0/0
Paris(config-if)#ipv6 ena
Paris(config-if)#ipv6 enable
Paris(config-if)#ipv6 address 2001:ABC:6::7/64 eui-64
Paris(config-if)#ipv6 address FE80::7 link
Paris(config-if)#ipv6 address FE80::7 link-local
Paris(config-if)#no sh
Paris(config-if)#no shutdown
Paris(config-if)#int fas 0/1
Paris(config-if)#ipv6 enable
Paris(config-if)#ipv6 address 2001:ABC:7::8/64 eui-64
Paris(config-if)#ipv6 address FE80::8 link-local
Paris(config-if)#no shutdown
Paris(config-if)#exit
Paris(config)#ipv6 route 2001:ABC:7::/64 fas 0/0
Paris(config)#ipv6 route 2001:ABC:6::/64 fas 0/1
Paris(config)#exit
Paris#
%SYS-5-CONFIG_I: Configured from console by console

Paris#wr
Building configuration...
[OK]
```

Figure 6.5: Paris router configuration

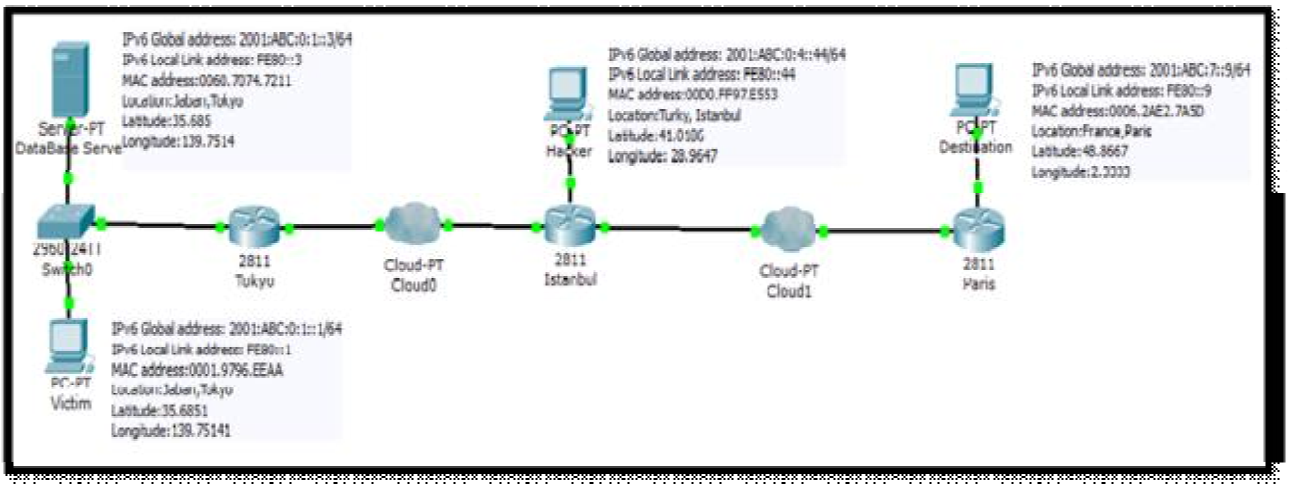
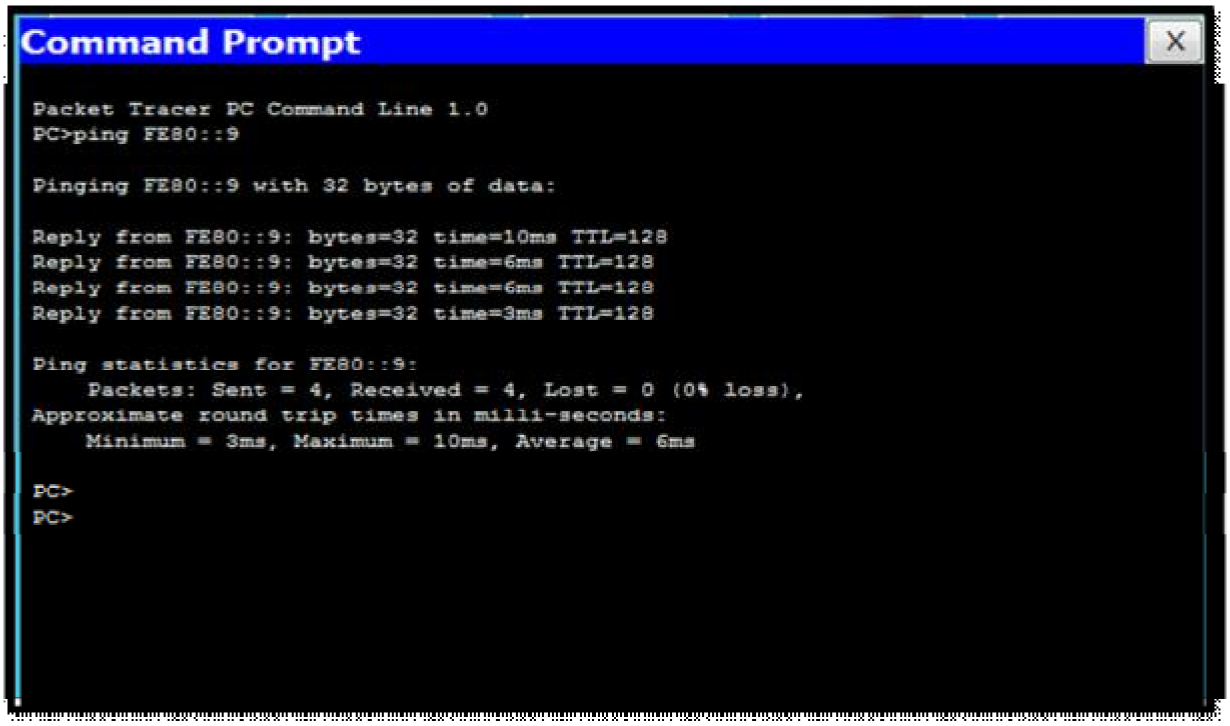

IPv6 Global address: 2001:ABC:0:1::3/64
IPv6 Local Link address: FE80::3
MAC address:0060.7074.7211
Location:Jaban,Tokyo
Latitude:35.685
Longitude:139.7514
Server-PT
DataBase Server
2960 24TT
Switch0
2811
Tokyo
Cloud-PT
Cloud0
IPv6 Global address: 2001:ABC:0:4::44/64
IPv6 Local Link address: FE80::44
MAC address:00D0.FF97.E553
Location:Turky, Istanbul
Latitude:41.0106
Longitude: 28.9647
PC-PT
Hacker
2811
Istanbul
Cloud-PT
Cloud1
IPv6 Global address: 2001:ABC:7::9/64
IPv6 Local Link address: FE80::9
MAC address:0006.2AE2.7A5D
Location:France,Paris
Latitude:48.8667
Longitude:2.3333
PC-PT
Destination
2811
Paris
IPv6 Global address: 2001:ABC:0:1::1/64
IPv6 Local Link address: FE80::1
MAC address:0001.9796.EEAA
Location:Jaban,Tokyo
Latitude:35.6851
Longitude:139.75141
PC-PT
Victim

Figure 6.6: Test bed topology of Figure 6.2 (second test bed of HLDS) after being configured

We used ping command to test the health of the path connecting source and destination of the second test bed which is shown in Figure 6.2. The result of the ping is shown in Figure 6.7. Figure 6.7 shows that there is a network connection between Tokyo (source node) and Paris (destination node).



```
Command Prompt

Packet Tracer PC Command Line 1.0
PC>ping FE80::9

Pinging FE80::9 with 32 bytes of data:

Reply from FE80::9: bytes=32 time=10ms TTL=128
Reply from FE80::9: bytes=32 time=6ms TTL=128
Reply from FE80::9: bytes=32 time=6ms TTL=128
Reply from FE80::9: bytes=32 time=3ms TTL=128

Ping statistics for FE80::9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 10ms, Average = 6ms

PC>
PC>
```

Figure 6.7: ping from Tokyo node to Paris node of the test bed of Figure 6.2 (second test bed)

The next step is to run HLDS Code shown in Code 6.5. In Code 6.5, HLDS runs traceroute program between source node and destination node. Code 6.5 shows the HLDS Scapy program needed to implement this task. In Code 6.5 HLDS invokes traceroute. Code 6.5 works for the test bed case of Figure 6.2. Code 6.5 is similar to the HLDS code 6.1 used in the first test bed. Hence, same explanation of Code 6.1 applies to Code 6.5.

```
### Create Victim (source node) IPv6 Packet
        i=IPv6()
        i.src="2001:ABC:0:1::1"
        i.dst="2001:ABC:0:7::9"
        q=ICMPv6EchoRequest()
        p=(i/q)
        sr1(p)
### Create Victim (source node) IPv6 Packet path without
### passing through hacker’s location
        h1=IPv6ExtHdrRouting()
        h1.addresses=["2001:ABC:0:1::2","2001:ABC:0:2::2","2001:ABC:0:3::5","
        2001:ABC:0:5::6","2001:ABC:0:6::7","2001:ABC:7::8"]
### Create Victim IPv6 Packet path that passes through hacker’s location
        h2=IPv6ExtHdrRouting()
        h2.addresses=["2001:ABC:0:1::2","2001:ABC:0:2::2","2001:ABC:0:3::5","
        2001:ABC:0:4::4","2001:ABC:0:5::6","2001:ABC:0:6::7","2001:ABC:7::8"]
### Invoke Traceroute for Victim (source node) IPv6 Packet
        waypoint = "2001:ABC:0:1::2"
        target = "2001:ABC:7::9"
        t1= traceroute6(waypoint, minttl=15 , maxttl=34, l4=h1
      (addresses=[target]) /ICMPv6EchoRequest(data=RandString(7)))
        t2= traceroute6(waypoint, minttl=15 , maxttl=34, l4=h1
      (addresses=[target]) /ICMPv6EchoRequest(data=RandString(7)))
### Convert traced IPv6 addresses to locations
        L1=Convtolocation(t1)
        L2=Convtolocation(t2)
### check for hacking
      for i in range(2):
               for addr in Li:
                       If addr=hackerLocation
                               print('Your packet may be hacked')
                       els continue
               end for
      end for
```

Code 6.5: HLDS Scapy program that tests test bed topology of Figure 6.2 (second test bed)

When running HLDS program of Code 6.5 and force the crafted packet to pass through the hacked path, HLDS generates the alarm message `'Your packet may have been` `hacked'` and sends this message to the source node. When running HLDS program of Code 6.5 and force the crafted packet to pass through the safe path, HLDS does not generate any alarm message. This means that HLDS works correctly for the second test bed topology.

6.4 HLDS Results of Rest of Test Beds

In order to validate HLDS more thoroughly, we developed 300 test bed topologies that are similar to first and second test bed topologies presented in Section 5.3.3 and 6.3 respectively. HLDS is run for each of these 300 test bed topologies in both Packet Tracer simulator and Scapy environment.

For every test bed topology of these 300, we configured two paths between source and destination: healthy path that does not include any hacked router, and hacked path that includes a hacked router. We configured every test case topology including source node, destination node, intermediate nodes, and hacked router (in case of hacked router). For every router in the path we configured its IPv6 address, and location. For every test bed topology of these 300 test bed topologies, and for every path (healthy and hacked), we measured whether HLDS generates an alarm or not. HLDS performance results for the first 20 test bed topologies of the 300 test bed topologies are summarized in Table 6.1. Note that HLDS performance results of all 300 test bed topologies are computed and measured but only the performance results of the first 20 test bed topologies is shown for simplicity. For every test bed topology we of the 300 test bed topologies we measured (for both healthy and hacked path) Detection Rate (DR)/Accuracy, False Positive Rate (FPR), True Positive Rate (TPR), False Negative Rate (FNR), and True Negative Rate (TNR).

Results of running HLDS using these 300 test bed topologies are as follows:

Detection Rate (DR)	= (571/600)*100 %	= 95.16%
False Positive Rate (FPR)	= (4/300)*100 %	= 1.3%
True Positive Rate (TPR)	= (287 / 300)*100 %	= 95.6 %
False Negative Rate (FNR)	= (25/300)*100 %	= 8.3%
True Negative Rate (TNR)	= (284 / 300)*100 %	= 94.6%

Table 6.1: NLDS performance results (accuracy) for 1st 20 test bed topologies out of 300 test bed topologies

Test bed No.	Hacker's Location	Hacker's IPv6 Address	Hacker's Latitude	Hacker Longitude	Hacked path?	HLDS gives an alarm	Result
1.	Cyprus Lamaca	2001:BDA:0:4::44/64 FE80::44	34.9081	33.6355	Yes	Yes	True Positive
					No	No	True Negative
2.	Turkey Istanbul	2001:ABC:0:4::44/64 FE80::44	41.0186	28.9647	Yes	Yes	True Positive
					No	No	True Negative
3.	Cote D'Ivoire,	2001:AB:29CA:8000	8	-5	Yes	Yes	True Positive
					No	No	True Negative
4.	Nigeria, Kachia	2001:A1:50FF:2870	10	8	Yes	Yes	True Positive
					No	No	True Negative
5.	UK, Leeds	2001:B1:538A:A728	54	-2	Yes	Yes	True Positive
					No	No	True Negative
6.	Nigeria, Port Harcourt	2001:5A:29CD:A000	4.7774	7.0134	Yes	No	False Positive
					No	No	True Negative
7.	Switzerland, Willisau	2001:2C:C32C:A800	47	8	Yes	Yes	True Positive
					No	No	True Negative
8.	Canada, Manitoba	2001:EF01:D046:0000	60	-95	Yes	Yes	True Positive
					No	Yes	False Positive
9.	Austria, Falchau	2001:A234:D992:03B0	47.3333	13.3333	Yes	Yes	True Positive
					No	No	True Negative
10.	Ukraine, Kamiaka	2001:A78:4D78:0000	49	32	Yes	Yes	True Positive
					No	No	True Negative

Table 6.1: NLDS performance results (accuracy) for 1st 20 test bed topologies out of 300 test bed topologies (Continued)

11.	**Russian, Moscow**	**2001:A92:4F78:0000**	**55.7522**	**37.6156**	**Yes**	**Yes**	**True Positive**
					No	No	True Negative
12.	Romania, Rupea	2001:E12:524C:0000	46	25	Yes	Yes	True Positive
					No	No	True Negative
13.	Russian Kaliningrad	2001:F567:53DB:8100	55.9142	37.8256	Yes	Yes	True Positive
					No	No	True Negative
14.	Bulgaria, Blagoevgrad	2001:8965:550E:2300	42.0167	23.1	Yes	Yes	True Positive
					No	No	True Negative
15.	Romania, Constanta	2001:ABDC:567F:1300	44.1833	28.65	Yes	No	False Negative
					No	Yes	False Positive
16.	China, Zhengzhou	2001:FCA:01C0:0000	34.6836	113.5325	Yes	Yes	True Positive
					No	No	True Negative
17.	India, Risod	2001:DCA:0E60:0000	20	77	Yes	Yes	True Positive
					No	No	True Negative
18.	Indonesia, Bontocani	2001:EEA:3132:0400	-5	120	Yes	Yes	True Positive
					No	No	True Negative
19.	Japan, Kaya	2001:1A23:1B32:6000	36	138	Yes	Yes	True Positive
					No	No	True Negative
20.	Korea, Seojeongri	2001:23A9:1B73:8000	37	127.5	Yes	Yes	True Positive
					No	No	True Negative

6.4 Performance Comparison between HLDS and Other Similar Tools

In this section we measure the performance of HLDS and other similar tools to prove that HLDS is more accurate than most of the current similar tools in the literature.

To compare performance of HLDS with similar tools from the literature, we used Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree [80]. [87] proposed a hierarchical Self Organized Map (SOM) for intrusion detection. They utilized the classification capability of the SOM on selected dimensions and specific attention was given to the hierarchical development of abstractions. The reported results showed that there was an increase in attack detection rate. [88] demonstrated that the ensemble of di erent learning paradigms by assigning proper weight to the individual classifiers. They have also observed that there was an improvement on attack detection and significant reduction on false alarm. Several hybrid Intrusion Detection System (IDS) have been proposed recently to deal with the complexity of the intrusion detection problem by combining di erent machine learning algorithms. [89] were developed a hybrid intelligent IDS by incorporating a Hierarchical Clustering and Support Vector Machines (SVM). The SVM theory was slightly modified in this research in order to be used with standard network intrusions dataset that contains labels. [90] designed IDS by combining the supervised tree classifiers and unsupervised Bayesian clustering to detect intrusions. [91] proposed a new framework of unsupervised anomaly Network IDS based on the outlier detection technique in random forests algorithm. The framework builds the patterns of network services over datasets labeled by the services. With the built in patterns, the framework detected attacks in the datasets using the outlier detection algorithm. This approach reduced the time complexity and cost of memory to a larger extent. [92] took a slightly di erent approach. Their anomaly IDS was based on modular multiple classifier system where each module was designed for each group of protocols and services. The reported results showed that this approach provides a better trade-o between generalization abilities and false alarm generation than that provided by an individual classifier trained on the overall feature set. [93] Have demonstrated a new ensemble boosted decision tree for intrusion detection system. The underlying idea of this approach is to combine simple rules to form an ensemble such that the performance of the single ensemble is improved. [94] Have constructed a classifier by using a decision tree as its base learner. The classification accuracy of this algorithm was little better than SOM algorithms. [95] Have

proposed an Adaboost based algorithm for network intrusion detection which used decision stump as a weak learner. The decision rules are provided for both categorical and continuous features and some provision was made for handling the overfitting.
The performance metrics used for the comparison are detection rate (accuracy), false positive and false negative rates. The detection rate and false positive rate of HLDS are compared to Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree. Results are shown in Table 6.2. As the table shows, HLDS is more accurate (has a higher Detection Rate – DR) than Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree. In addition, HLDS has the lowest False Positive Rate (FPR) than Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree. This shows that HLDS has the best performance that existing similar Internet security tools present in the literature.

Table 6.2: Performance comparison between HLDS and other similar tools

Tool	**Detection Rate –DR (%)**	**False Positive Rate – FPR (%)**
HLDS	95.16	1.3
Hierarchical SOM	90.04	2.19
IDS using SVM	91.2	6.12
Adaboost with Decision tree	92.12	3.26

Detection Rate (DR) for HLDS, Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree is shown in Figure 6.8. HLDS has the highest DR.
False Positive Rate (FPR) for HLDS, Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree is shown in Figure 6.9. HLDS has the lowest FPR.
It is important for a network security tool/IDS to have low False Positive Rate (FPR) and high accuracy/detection Rate (DR).

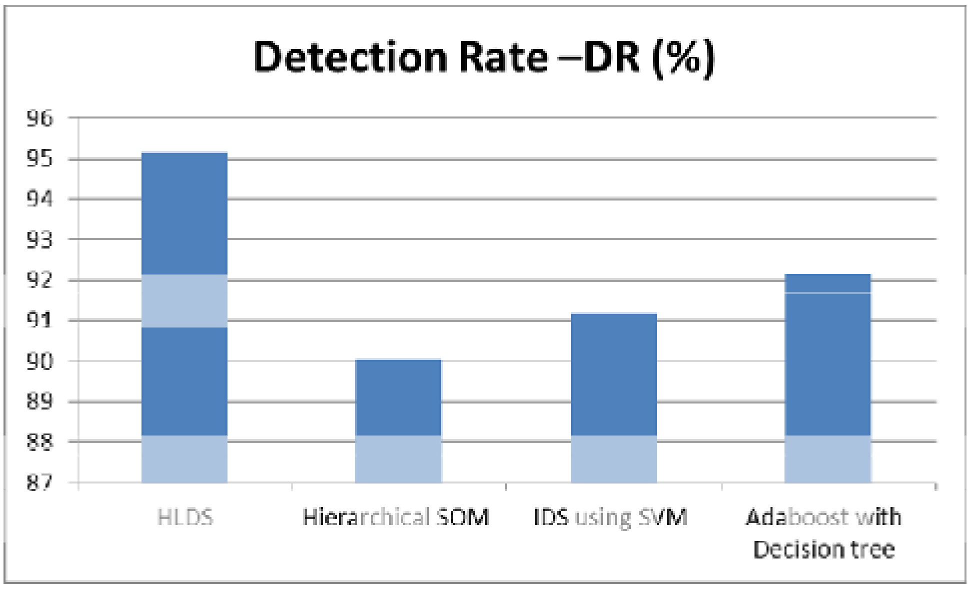


Figure 6.8: Detection Rate (DR) for HLDS, Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree

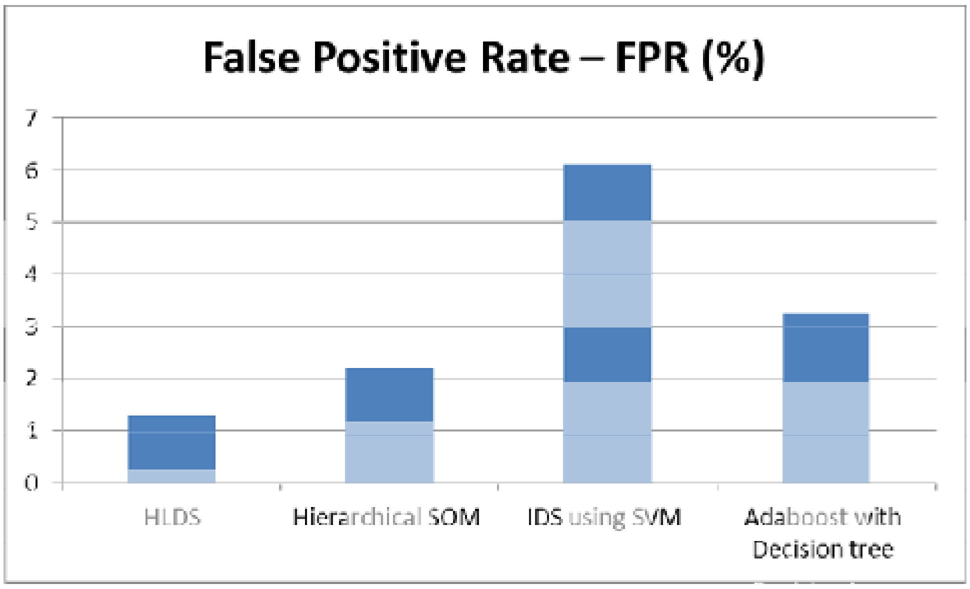


Figure 6.9: False Positive Rate (FPR) for HLDS, Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree

Chapter Seven Conclusion and Future work

In this Master thesis we proposed a new methodology to detect the hackers depending on their location. This methodology is called Hacker Location Detection System (HLDS). HLDS monitors packing while they pass through the network and sends an alarm message to the involved communicating parties about the status of the path between them. HLDS uses Scapy, Packet Tracer simulator, and traceroute to fulfill its objectives.

HLDS enables a packet crafter of Scapy to pull packets off the wire or create ones as required. Once a packet set is assembled, Scapy will enable the Crafter to change any fields in any header of any packet, as well as altering the payload of the packet to the Crafters own desires. A fully assembled packet set must be played or replayed onto the network as many times and at any speed required as stipulated by the testing case at hand; Scapy is able to replay such packets. Packets launched onto the network will most generally cause a response from the target device. Such response must be captured and analyzed to further understand the problem at hand or to confirm the results; Scapy is able to decode packets.

HLDS uses Open Source tools that are readily available to accomplish the task of Packet Assembly, Packet Editing, Packet tracing and Packet decoding; all together constitute Packet Crafting. Two of the well documented industry standard protocols, and one proprietary protocol were used to demonstrate the ease and power of Scapy as it adheres to the Packet Monitoring methodology.

By using HLDS SRP we are able to identify a lot of network vulnerabilities such as Man in the middle, sniffing IP, spoofing IP and hacker honeypot.

By introducing such a tool HLDS and demonstrating the danger of such a tool when used by the wrong hands; users need to see such potential danger and be prepared for it. For the time being it can be said that training is essential because the engineers who grew up using IPv4 need a fresh education because IPv6 quite different than IPv4, and it uses different tools. Also the security policies between IPv4 and IPv6 aren’t consistent. In addition, we recommend that people be ready to run IPv6 to get their systems prepared for this new technology. People also need to try to protect themselves against the known network weaknesses.

The contribution of this thesis is the design of an original network monitoring system that automatically detects network vulnerabilities, weaknesses, and shortcomings of IPv6-based networks. The designed system is called Hacker Location Detection System (HLDS) which is a Software Router Plug-in (SRP) that could be deployed as advanced and flexible tool to IPv6 next generation routers to increase the network security. HLDS is capable of both detection of hackers and identification of their location based on their IP address, MAC address, and geographic location. HLDS is used in IPv6 routers to provide Internet security. Since hackers are able to change IP and MAC addresses of their computers and NIC respectively, our proposed system does not depend only on IP or MAC addresses but also on location of hackers. Hence, even if hackers change these IP and/or MAC addresses our proposed system is capable of tracing these hackers.

HLDS has four main modules: HLDB Server Module which contains the Hacker Location Database (HLDB) that includes the known locations of the hackers, HLDB Update Module which updates the HLDB based on the New hacker location information received, Traceroute Module which is used to trace the route of the packet that is being monitored and it lists all routers that the packet passed through from source to destination, and finally HLDS Monitor Module which monitors the packets paths and generates an alarm if any packet that is being monitored has passed through a hacker router.

We validate the proposed HLDS by developing experiments to measure its performance. We used some performance metrics to measure the performance of HLDS which include accuracy, False positive rate, and False negative rate. We also compare the performance results of HLDS with performance of similar tools in the literature, namely Hierarchical SOM, IDS using SVM, and Adaboost with Decision tree. Performance results show that HLDS has a better accuracy, less false positive rate, and less false negative rate than these programs.

Future work:

- Could be to try to connect HLDS with GPS and GIS to improve modeling hacker location positions more accurately.
- IDS companies adopt this idea and create a set of database servers that contain blacklists of hacker's locations and publish these databases around the world and update them continuously to easily identify the hacker's locations.

References

[1] Raffaele Bolla, Roberto Bruschi, Guerino Lamanna and Andrea Ranieri, " DROP: An Open- Source Project towards Distributed SW Router Architectures" This paper appears in: Performance Evaluation of Computer and Telecommunication Systems (SPECTS), International Symposium on July 2010.

[2] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification.” RFC 2460 (Draft Standard), Dec. 1998. Updated by RFCs 5095, 5722, 5871.

[3] J. Postel, “Internet Protocol.” RFC 791 (Standard), Sept. 1981. Updated by RFC 1349.

[4] Hurricane Electric Internet Services, “Hurricane electric ipv4 exhaustion counters.” http://ipv6.he.net/statistics, April 2011.

[5] N Ranjith Kumar," Seminar Report on IPv6: Next Generation IP" School of Information Technology Indian Institute of Technology-Kharagpur, May2005.

[6] Martin Sch¨utte "Design and Implementation of an IPv6 Plugin for the Snort Intrusion Detection System" Potsdam University Institute for Computer Science September 1, 2011.

[7] IAB and IESG, “IAB/IESG Recommendations on IPv6 Address Allocations to Sites.” RFC 3177 (Informational), Sept. 2001. Obsoleted by RFC 6177.

[8] S. Chozos, “Implementation and analysis of a threat model for ipv6 host autoconfiguration,” Master’s thesis, Naval Postgraduate School, Monterey, CA, 2006.

[9] S. Hogg and E. Vyncke, IPv6 Security. Cisco Press, 1st ed., 2008.

[10] R. Hinden and S. Deering, “Internet Protocol Version 6 (IPv6) Addressing Architecture.” RFC 3513 (Proposed Standard), Apr. 2003. Obsoleted by RFC 4291.

[11] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification.” RFC 2460(Draft Standard), Dec. 1998. Updated by RFCs 5095, 5722, 5871.

[12] S. Chozos, “Implementation and analysis of a threat model for ipv6 host autoconfiguration,”Master’s thesis, Naval Postgraduate School, Monterey, CA, 2006.

[13] J. Postel, “Internet Control Message Protocol.” RFC 792 (Standard), Sept. 1981. Updated by RFCs 950, 4884.

[14] A. Conta, S. Deering, and M. Gupta, “Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification.” RFC 4443 (Draft Standard), Mar. 2006. Updated by RFC 4884.

[15] Gehrke, Keith A., "The Unexplored Impact of IPv6 on Intrusion Detection Systems", April 2012.

[16] Free download of ISO/IEC 27000:2009 from ISO, via their ITTF web site

[17] S. Convery and D. Miller, “Ipv6 and ipv4 threat comparison and best-practice evaluation (v1.0),” www.seanconvery.comv6-v4-threats.pdf, March 2004.

[18] D. Zagar and K. Grgic, “Ipv6 security threats and possible solutions,” World Automation Congress, 2006.

[19]SourceForge, “Tcpdump.” http://www.tcpdump.org, Feb. 2011.

[20] S. Bellovin, A. Keromytis, and B. Cheswick, “Worm propagation strategies in an ipv6 internet,” USENIX ;login, vol. 31, pp. 70–76, February 2006.

[21] S. Staniford, V. Paxson, and N. Weaver, “How to own the internet in your spare time,” in Proceedings of the 11th USENIX Security Symposium, (Berkeley, CA, USA), pp. 149–167, USENIX Association, 2002.

[22] Emmett Dulaney, "CompTIA security+ deluxe study guide " 1st ed, 2009

[23] C. Caicedo, J. Joshi, and S. Tuladhar, “Ipv6 security challenges,” Computer, vol. 42, pp. 36 –42, feb. 2009.

[24] E. Nerakis, “Ipv6 host fingerprint,” Master’s thesis, Naval Postgraduate School, Monterey, CA, 2006.

[25] R. Hinden and S. Deering, “IPv6 Multicast Address Assignments.” RFC 2375 (Informational), July 1998.

[26] P. Ferguson and D. Senie, “Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.” RFC 2827 (Best Current Practice), May 2000. Updated by RFC 3704.

[27] J. Arkko, J. Kempf, B. Zill, and P. Nikander, “SEcure Neighbor Discovery (SEND).” RFC 3971 (Proposed Standard), Mar. 2005.

[28] M. Pohl, “Experimentation and evaluation of ipv6 secure neighbor discovery protocol,” Master’s thesis, Naval Postgraduate School, Monterey, CA, 2007.

[29] A. Conta and S. Deering, “Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification.” RFC 2463 (Draft Standard), Dec. 1998. Obsoleted by RFC 4443.

[30] S. Bellovin, A. Keromytis, and B. Cheswick, “Worm propagation strategies in an ipv6 internet,” USENIX ;login, vol. 31, pp. 70–76, February 2006.

[31] Hurricane Electric Internet Services, “Hurricane electric homepage.” http://ipv6.he.net/, April 2011.

[32] C. Aoun and E. Davies, “Reasons to Move the Network Address Translator – Protocol Translator (NAT-PT) to Historic Status.” RFC 4966 (Informational), July 2007.

[33] Oracle, “Virtualbox homepage.” https://www.virtualbox.org/, Sept 2011.

[34] D. Morr, “Thoughts on ipv6 security, take two.” http://www.personal.psu.edu/dvm105/blogs/ipv6/2009/05/thoughts-on-ipv6-securitytake. html, May 2009.

[35] http://www.ntia.doc.gov/legacy/ntiahome/ntiageneral/ipv6/final/IPv6final3.htm#ftn171.

[36] http://www.secdev.org/projects/scapy/.

[37] http://www.secdev.org/projects/scapy/demo.html.

[38] Adam Maxwell, "The Very Unofficial Dummies Guide to Scapy", May 2012.

[39] Philippe BIONDI, "Scapy Documentation", Oct 2006.

[40] azem M. El-Bakry, Nikos MastorakisA, “Real-Time Intrusion Detection Algorithm for Network Security,WSEAS Transactions on communications, Issue 12, Volume 7, December 2008.

[41] Debar.H, Dacier.M and Wespi.A, “A Revised Taxonomy of Intrusion-Detection Systems” Annales des Telecommunications 55(7–8) (2000) 361–378 .

[42]Allen.J, Christie.A, Fithen.W, McHugh.J, Pickel.J, Stoner.E, “State of the practice of intrusion detection technologies” Technical Report CMU/SEI-99TR- 028, Carnegie-Mellon University - Software Engineering Institute (2000).

[43] Roesch.M, “Snort - Lightweight Intrusion Detection for Networks” 13th USENIX Conference on System Administration, USENIX Association (1999) 229–238 .

[44] Sourcefire: Snort Network Intrusion Detection System web site (1999) URL http://www.snort.org.

[45] Wang. K and Stolfo.S.J, “Anomalous Payload-Based Network Intrusion Detection” 7th Symposium on Recent Advances in Intrusion Detection, Volume 3224 of LNCS., Springer-Verlag (2004) 203–222.

[46] Bolzoni.D, Zambon.E., Etalle.S, Hartel.P, “POSEIDON: a 2-tier Anomaly based Network Intrusion Detection System”IEEE International Workshop on Information Assurance, IEEE Computer Society Press (2006) 144–156.

[47]V. Jyothsna, V. V. Rama Prasad, and K. Munivara Prasad," A Review of Anomaly based Intrusion Detection Systems", International Journal of Computer Applications (0975 – 8887(Volume 28– No.7, August 2011.

[48]Y. Freund, Schapire.R. , "Experiments with a new boosting algorithm" Thirteenth International Conference on Machine Learning, Italy, 1996.

[49]B.Pfahringer, "Winning the KDD99 Classification Cup: Bagged Boosting," in SIGKDD Explorations, 2000.

[50]I. Levin, "KDD-99 Classifier Learning Contest: LLSoft s Results Overview" SIGKDD Explorations, 2000.

[51]V. Miheev, Vopilov.A and Shabalin.I., "The MP13 Approach to the KDD 99 Classifier Learning Contest" SIGKDD Explorations, 2000.

[52]Q. Yang, Li, F., "Support Vector Machine for Intrusion Detection Based on LSI Feature Selection," Intelligent Control and Automation, WCICA, 2006.

[53] J. C. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines" Advances in Kernel Method: Support Vector Learning, 1998.

[54]F. E. Osuna, R., Girosi, F., "Improved training algorithm for support vector machines," IEEE NNSP 97, 1997.

[55]Y. Yao, Wei, Y., Gao, F.X., Yu, G. , "Anomaly Intrusion Detection Approach Using Hybrid MLP/CNN Neural Network," Sixth International Conference on Intelligent Systems Design and Applications (ISDA'06) Washington, DC, USA 2006.

[56]D. F. Specht, "Probabilistic Neural Network," International Journal of Neural Networks, vol. 3, pp. 109-118, 1990.

[57]A. Zaknich, "Introduction to the modified probabilistic neural network for general signal processing applications" IEEE Transactions on Signal Processing, vol. 46, 1998.

[58]D. Barbar´a, C. Domeniconi and J. Rogers, “Detecting outliers using transduction and statistical testing” ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), Philadelphia, PA,Aug. 2003.

[59]J. Ma and S. Perkins, “Online novelty detection on temporal sequences” ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), Washington, DC, Aug. 2003.

[60]A. Ihler, J. Hutchins, and P. Smyth, “Adaptive event detection with time-varying Poisson processes” ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD), Philadelphia, PA, Aug. 2006.

[61]D. Dasgupta, “Artificial Immune Systems and Their Applications” Springer, 1999

[62]S. A. Hofmeyr, S. Forrest, “Architecture for an artificial immune system” IEEE Trans. on Evolutionary Computation, vol. 8, N4, 2000, pp. 443-473.

[63]E. Hart, P. Ross, J. Nelson, “Producing robust schedules via an artificial immune system” IEEE International Conference on Evolutionary Computing, May 1998, pp. 464-469.

[64]A. Gardner, A. Krieger, G. Vachtsevanos, and B. Litt, “One-class novelty detection for seizure analysis from intracranial EEG,” J. Machine Learning Research (JMLR), vol. 7, pp. 1025–1044, Jun. 2006.

[65] "NIST – Guide to Intrusion Detection and Prevention Systems (IDPS)". 2007-02. Retrieved 2010-06-25.

[66] Robert C. Newman ,"Computer Security: Protecting Digital Resources". Jones & Bartlett Learning. pp. 273–. ISBN 978-0-7637-5994-0. Retrieved 25 June 2010.

[67]Michael E. Whitman; Herbert J. Mattord."Principles of Information Security". Cengage Learning EMEA. pp. 289–. ISBN 978-1-4239-0177-8. Retrieved 25 June 2010.

[68]Tim Boyles ."CCNA Security Study Guide": Exam 640-553. John Wiley and Sons. pp. 249–. ISBN 978-0-470-52767-2. Retrieved 29 June 2010.

[69]Harold F. Tipton; Micki Krause."Information Security Management Handbook". CRC Press. pp. 1000–. ISBN 978-1-4200-1358-0. Retrieved 29 June 2010.

[70] Matt Carlson and Andrew Scharlott. Intrusion detection and prevention systems, (2006).

[71] Wei, L., 2007. Evaluation of Intrusion Detection Systems. University of Auckland, Department of Coputer Science, pp.1-10.

[72] Martins Sapats, Nauris Paulins" Evaluation Methods Of Network Intrusion Detection Systems " International Conference on Applied Information and Communication Technologies (AICT2012), 26.-27. April, 2012, Jelgava, Latvia.

[73] Atay, S. & Masera, M., 2011. Challenges for the security analysis of Next Generation Networks. Information Security Technical Report, 16(1), pp.3-11. Available at: http://linkinghub.elsevier.com/retrieve/pii/S136341271000035X [Accessed October 10, 2011].

[74] Convery, Sean, and Miller, Darrin. IPv6 and IPv4 threat comparison and best-practice evaluation (v1.0). in IPv6 Eprints Server [database online]. 2006 Available from http://www.6journal.org/archive/00000202/01/v6-v4-threats.pdf (accessed 12/21/ 2006).

[75] NTT/Verio Comments at 13. See also Microsoft Comments at 11 (IPv6 is a “new, more secure protocol” that could help make North America a “Safe Cyber Zone”).

[76] BellSouth Comments at 3. http://www.ntia.doc.gov/legacy/ntiahome/ ntiageneral/ipv6/ final/IPv6final 3.htm.

[77] http://www.webopedia.com/DidYouKnow/Computer_Science/2005/intrusion_detection_prevention.asp.

[78] Scarfone, K. and P. Mell. 2007. Guide to Intrusion Detection and Prevention Systems. National Institute of Standards and Technology. Special Publication 800-94. http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf.

[79] Buyer's Guide For Intrusion Prevention Systems (IPS) (http://www.forum-intrusion.com/Buyers_Guide.pdf).

[80] Natesan P ،Balasubramanie P, and Gowrison G." PERFORMANCE COMPARISON OF ADABOOST BASED WEAK CLASSIFIERS IN NETWORK INTRUSION DETECTION"Journal of Information Systems and Communication ISSN: 0976-8742 & E-ISSN: 0976-8750, Volume 3, Issue 1, 2012, pp.-295-299.

[81] http://www.mcafee.com

[82] http://whatismyipaddress.com/ip/213.244.80.135.

[83] http://whatismyipaddress.com/traceroute-tool.

[84] http://www.exit109.com/~jeremy/news/providers/traceroute.html.

[85] (http://www.secdev.org/projects/scapy/doc/).

[86] http://www.cisco.com/web/learning/netacad/index.html, Cisco Network Academy Packet Tracer.

[87] H. G. Kayacik, A. NZincir-Heywood, M. I. Heywood, On the capability of an SOM based intrusion detection systems, in Proc. Int. Joint Conference in Neural Networks. Vol. 3, 2003, 1808-1813.

[88] Anazida Zainal, Mohd Aizaini Maarof, Siti Mariyam Shasuddin, Ensemble classifiers for network intrusion cetection system. Journal of Information Assurance and Security, Vol. 4, 2009, 217-225.

[89] Shi-Jinn Horng, Ming-Yang Su, Yuan-Hsin Chen, Tzong-Wann Kao, Rong-Jian Chen, Jui-Lin Lai, Citra Dwi Perkasa, A novel intrusion detection system based on hierarchical clustering and support vector machines, Journal of Expert systems with Applications, Vol. 38, 2011, 306-313.

[90] Cheng Xiang, Png Chin Yong, Lim Swee Meng, Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees, Journal of Pattern Recognition Letters, Vol. 29, 2008, 918-924.

[91] Jiong Zhang, Mohammad Zulkernine, Anomaly based network intrusion detection with unsupervised outlier detection, Proc. IEEE Communication Society, 2006, 2388-2393.

[92] Giorgio Giacinto, Fabio Roli, Luca Didaci, Fusion of multiple classifiers for intrusion detection in computer networks, Proc. IEEE Conference in Network Security, 2007, 23-32.

[93] Mrudula Gudahe, Prakash Prasad, Kapil Wankhade, A new data mining based network intrusion detection model, International Conference on Computer & Communication Technology, 2010, 731- 736.

[94] Yongiin Liu, Na Li, Leina Shi, Fangping Li, An intrusion detection method based on decision tree, International Conference on E-Health Networking, Digital Ecosystems and Technologies, 2010, 232-235.

[95] Weiming Hu, Wei Hu, Steve Maybank, AdaBoost-based algorithm for network intrusion detection, IEEE Transactions on Systems, Man and Cybernetics, Vol. 38, April-2008, 577-583.